

An Exponential Improvement for Diagonal Ramsey

Lawrence C. Paulson

6 February 2026

Abstract

The (diagonal) Ramsey number $R(k)$ denotes the minimum size of a complete graph such that every red-blue colouring of its edges contains a monochromatic subgraph of size k . In 1935, Erdős and Szekeres found an upper bound, proving that $R(k) \leq 4^k$. Somewhat later, a lower bound of $\sqrt{2}^k$ was established. In subsequent improvements to the upper bound, the base of the exponent stubbornly remained at 4 until March 2023, when Campos et al. [1] sensationally showed that $R(k) \leq (4 - \epsilon)^k$ for a particular small positive ϵ .

The Isabelle/HOL formalisation of the result presented here is largely independent of the prior formalisation (in Lean) by Bhavik Mehta.

Contents

1	Background material: the neighbours of vertices	5
1.1	Preliminaries on graphs	5
1.2	Neighbours of a vertex	7
1.3	Density: for calculating the parameter p	8
1.4	Lemma 9.2 preliminaries	14
2	The book algorithm	19
2.1	Locales for the parameters of the construction	19
2.2	State invariants	28
2.3	Degree regularisation	29
2.4	Big blue steps: code	31
2.5	The central vertex	32
2.6	Red step	33
2.7	Density-boost step	35
2.8	Execution steps 2–5 as a function	36
2.9	The classes of execution steps	40
2.10	Termination proof	44
3	Big Blue Steps: theorems	48
3.1	Preliminaries	48
3.2	Preliminaries: Fact D1	51
4	Red Steps: theorems	69
4.1	Density-boost steps	70
4.1.1	Observation 5.5	70
4.1.2	Lemma 5.6	71
4.2	Lemma 5.4	74
4.3	Lemma 5.1	80
4.4	Lemma 5.3	88
5	Bounding the Size of Y	89
5.1	The following results together are Lemma 6.4	90
5.2	Towards Lemmas 6.3	92
5.3	Lemma 6.5	94
5.4	Lemma 6.2	97
5.5	Lemma 6.1	103
6	Bounding the Size of X	107
6.1	Preliminaries	107
6.2	Lemma 7.2	111
6.3	Lemma 7.3	113
6.4	Lemma 7.5	117
6.5	Lemma 7.4	121

6.6	Observation 7.7	124
6.7	Lemma 7.8	125
6.8	Lemma 7.9	126
6.9	Lemma 7.10	129
6.10	Lemma 7.11	131
6.11	Lemma 7.12	136
6.12	Lemma 7.6	138
6.13	Lemma 7.1	141
7	The Zigzag Lemma	143
7.1	Lemma 8.1 (the actual Zigzag Lemma)	143
7.2	Lemma 8.5	153
7.3	Lemma 8.6	155
8	An exponential improvement far from the diagonal	157
8.1	An asymptotic form for binomial coefficients via Stirling's formula	157
8.2	Fact D.3 from the Appendix	158
8.3	Fact D.2	160
8.4	Lemma 9.3	163
8.5	Lemma 9.5	172
8.6	Lemma 9.2	175
8.7	Theorem 9.1	182
9	An exponential improvement closer to the diagonal	194
9.1	Lemma 10.2	194
9.2	Theorem 10.1	199
10	From diagonal to off-diagonal	213
10.1	Lemma 11.2	213
10.2	Lemma 11.3	219
10.3	Theorem 11.1	221
11	The Proof of Theorem 1.1	224
11.1	The bounding functions	224
11.2	The monster calculation from appendix A	236
11.2.1	Observation A.1	236
11.2.2	Claims A.2–A.4	238
11.3	Concluding the proof	243

Acknowledgements Many thanks to Mantas Bakšys, Chelsea Edmonds, Simon Griffiths, Bhavik Mehta, Fedor Petrov and Andrew Thomason for their help with aspects of the proofs. The author was supported by the ERC

Advanced Grant ALEXANDRIA (Project 742178), funded by the European Research Council.

1 Background material: the neighbours of vertices

Preliminaries for the Book Algorithm

theory *Neighbours* **imports** *Ramsey-Bounds.Ramsey-Bounds*

begin

abbreviation *set-difference* :: [*'a set, 'a set*] \Rightarrow *'a set* (**infixl** $\langle \setminus \rangle$ 65)
where $A \setminus B \equiv A - B$

lemma *setdif-eq-iff*: $\llbracket A \subseteq C; B \subseteq C \rrbracket \Longrightarrow C \setminus A = C \setminus B \longleftrightarrow A = B$
by *auto*

1.1 Preliminaries on graphs

context *ulgraph*

begin

The set of *undirected* edges between two sets

definition *all-edges-betw-un* :: [*'a set*] \Rightarrow [*'a set*] \Rightarrow [*'a set set*] **where**
 $all-edges-betw-un\ X\ Y \equiv \{\{x, y\} \mid x\ y.\ x \in X \wedge y \in Y\} \cap E$

lemma *all-edges-betw-un-commute1*: $all-edges-betw-un\ X\ Y \subseteq all-edges-betw-un\ Y\ X$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-commute*: $all-edges-betw-un\ X\ Y = all-edges-betw-un\ Y\ X$
by (*simp add: all-edges-betw-un-commute1 subset-antisym*)

lemma *all-edges-betw-un-iff-mk-edge*: $all-edges-betw-un\ X\ Y = mk-edge\ 'all-edges-between\ X\ Y$
by (*auto simp: all-edges-between-set all-edges-betw-un-def*)

lemma *all-uedges-betw-subset*: $all-edges-betw-un\ X\ Y \subseteq E$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-uedges-betw-I*: $x \in X \Longrightarrow y \in Y \Longrightarrow \{x, y\} \in E \Longrightarrow \{x, y\} \in all-edges-betw-un\ X\ Y$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-subset*: $all-edges-betw-un\ X\ Y \subseteq Pow\ (X \cup Y)$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-empty* [*simp*]:
 $all-edges-betw-un\ \{\}\ Z = \{\}\ all-edges-betw-un\ Z\ \{\} = \{\}$
by (*auto simp: all-edges-betw-un-def*)

lemma *card-all-uedges-betw-le*:

assumes *finite X finite Y*
shows $\text{card } (\text{all-edges-betw-un } X Y) \leq \text{card } (\text{all-edges-between } X Y)$
by (*simp add: all-edges-betw-un-iff-mk-edge assms card-image-le finite-all-edges-between*)

lemma *all-edges-betw-un-le:*
assumes *finite X finite Y*
shows $\text{card } (\text{all-edges-betw-un } X Y) \leq \text{card } X * \text{card } Y$
by (*meson assms card-all-uedges-betw-le max-all-edges-between order-trans*)

lemma *all-edges-betw-un-insert1:*
 $\text{all-edges-betw-un } (\text{insert } v X) Y = (\{\{v, y\} \mid y. y \in Y\} \cap E) \cup \text{all-edges-betw-un } X Y$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-insert2:*
 $\text{all-edges-betw-un } X (\text{insert } v Y) = (\{\{x, v\} \mid x. x \in X\} \cap E) \cup \text{all-edges-betw-un } X Y$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-Un1:*
 $\text{all-edges-betw-un } (X \cup Y) Z = \text{all-edges-betw-un } X Z \cup \text{all-edges-betw-un } Y Z$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-Un2:*
 $\text{all-edges-betw-un } X (Y \cup Z) = \text{all-edges-betw-un } X Y \cup \text{all-edges-betw-un } X Z$
by (*auto simp: all-edges-betw-un-def*)

lemma *finite-all-edges-betw-un:*
assumes *finite X finite Y*
shows *finite (all-edges-betw-un X Y)*
by (*simp add: all-edges-betw-un-iff-mk-edge assms finite-all-edges-between*)

lemma *all-edges-betw-un-Union1:*
 $\text{all-edges-betw-un } (\text{Union } \mathcal{X}) Y = (\bigcup X \in \mathcal{X}. \text{all-edges-betw-un } X Y)$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-Union2:*
 $\text{all-edges-betw-un } X (\text{Union } \mathcal{Y}) = (\bigcup Y \in \mathcal{Y}. \text{all-edges-betw-un } X Y)$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-mono1:*
 $Y \subseteq Z \implies \text{all-edges-betw-un } Y X \subseteq \text{all-edges-betw-un } Z X$
by (*auto simp: all-edges-betw-un-def*)

lemma *all-edges-betw-un-mono2:*
 $Y \subseteq Z \implies \text{all-edges-betw-un } X Y \subseteq \text{all-edges-betw-un } X Z$
by (*auto simp: all-edges-betw-un-def*)

lemma *disjnt-all-edges-betw-un:*

assumes *disjnt* $X Y$ *disjnt* $X Z$
shows *disjnt* (*all-edges-betw-un* $X Z$) (*all-edges-betw-un* $Y Z$)
using *assms* **by** (*auto simp: all-edges-betw-un-def disjnt-iff doubleton-eq-iff*)

end

1.2 Neighbours of a vertex

definition *Neighbours* :: 'a set set \Rightarrow 'a \Rightarrow 'a set **where**
Neighbours $\equiv \lambda E x. \{y. \{x,y\} \in E\}$

lemma *in-Neighbours-iff*: $y \in \text{Neighbours } E x \iff \{x,y\} \in E$
by (*simp add: Neighbours-def*)

lemma *finite-Neighbours*:

assumes *finite* E

shows *finite* (*Neighbours* $E x$)

proof –

have *Neighbours* $E x \subseteq \text{Neighbours } \{X \in E. \text{finite } X\} x$

by (*auto simp: Neighbours-def*)

also have $\dots \subseteq (\bigcup \{X \in E. \text{finite } X\})$

by (*meson Union-iff in-Neighbours-iff insert-iff subset-iff*)

finally show *?thesis*

using *assms finite-subset* **by** *fastforce*

qed

lemma (**in** *fin-sgraph*) *not-own-Neighbour*: $E' \subseteq E \implies x \notin \text{Neighbours } E' x$
by (*force simp: Neighbours-def singleton-not-edge*)

context *fin-sgraph*

begin

declare *singleton-not-edge* [*simp*]

"A graph on vertex set $S \cup T$ that contains all edges incident to S "
 (page 3). In fact, S is a clique and every vertex in T has an edge into S .

definition *book* :: 'a set \Rightarrow 'a set \Rightarrow 'a set set \Rightarrow bool **where**
book $\equiv \lambda S T F. \text{disjnt } S T \wedge \text{all-edges-betw-un } S (S \cup T) \subseteq F$

Cliques of a given number of vertices; the definition of clique from Ramsey is used

definition *size-clique* :: nat \Rightarrow 'a set \Rightarrow 'a set set \Rightarrow bool **where**
size-clique $p K F \equiv \text{card } K = p \wedge \text{clique } K F \wedge K \subseteq V$

lemma *size-clique-smaller*: $\llbracket \text{size-clique } p K F; p' < p \rrbracket \implies \exists K'. \text{size-clique } p' K' F$

unfolding *size-clique-def*

by (*meson card-Ex-subset order.trans less-imp-le-nat smaller-clique*)

1.3 Density: for calculating the parameter \mathbf{p}

definition $edge-card \equiv \lambda C X Y. card (C \cap all-edges-betw-un X Y)$

definition $gen-density \equiv \lambda C X Y. edge-card C X Y / (card X * card Y)$

lemma $edge-card-empty$ [*simp*]: $edge-card C \{\} X = 0$ $edge-card C X \{\} = 0$
by (*auto simp: edge-card-def*)

lemma $edge-card-commute$: $edge-card C X Y = edge-card C Y X$
using $all-edges-betw-un-commute$ $edge-card-def$ **by** *presburger*

lemma $edge-card-le$:

assumes $finite X$ $finite Y$

shows $edge-card C X Y \leq card X * card Y$

proof –

have $edge-card C X Y \leq card (all-edges-betw-un X Y)$

by (*simp add: assms card-mono edge-card-def finite-all-edges-betw-un*)

then show *?thesis*

by (*meson all-edges-betw-un-le assms le-trans*)

qed

the assumption that Z is disjoint from X (or Y) is necessary

lemma $edge-card-Un$:

assumes $disjnt X Y$ $disjnt X Z$ $finite X$ $finite Y$

shows $edge-card C (X \cup Y) Z = edge-card C X Z + edge-card C Y Z$

proof –

have [*simp*]: $finite (all-edges-betw-un U Z)$ **for** U

by (*meson all-uedges-betw-subset fin-edges finite-subset*)

have $disjnt (C \cap all-edges-betw-un X Z) (C \cap all-edges-betw-un Y Z)$

using *assms* **by** (*meson Int-iff disjnt-all-edges-betw-un disjnt-iff*)

then show *?thesis*

by (*simp add: edge-card-def card-Un-disjnt all-edges-betw-un-Un1 Int-Un-distrib*)

qed

lemma $edge-card-diff$:

assumes $Y \subseteq X$ $disjnt X Z$ $finite X$

shows $edge-card C (X \setminus Y) Z = edge-card C X Z - edge-card C Y Z$

proof –

have $(X \setminus Y) \cup Y = X$ $disjnt (X \setminus Y) Y$

by (*auto simp: Un-absorb2 assms disjnt-iff*)

then show *?thesis*

by (*metis add-diff-cancel-right' assms disjnt-Un1 edge-card-Un finite-Diff finite-subset*)

qed

lemma $edge-card-mono$:

assumes $Y \subseteq X$ **shows** $edge-card C Y Z \leq edge-card C X Z$

unfolding $edge-card-def$

proof (*intro card-mono*)

show $finite (C \cap all-edges-betw-un X Z)$

by (meson all-uedges-betw-subset fin-edges finite-Int finite-subset)
 show $C \cap \text{all-edges-betw-un } Y Z \subseteq C \cap \text{all-edges-betw-un } X Z$
 by (meson Int-mono all-edges-betw-un-mono1 assms subset-refl)
 qed

lemma *edge-card-eq-sum-Neighbours:*

assumes $C \subseteq E$ and B : finite B disjnt $A B$
 shows $\text{edge-card } C A B = (\sum i \in B. \text{card } (\text{Neighbours } C i \cap A))$
 using B
proof (induction B)
 case empty
 then show ?case
 by (auto simp: edge-card-def)
next
 case (insert $b B$)
 have finite C
 using assms(1) fin-edges finite-subset by blast
 have $\text{bij: bij-betw } (\lambda e. \text{the-elem}(e - \{b\})) (C \cap \{\{x, b\} \mid x. x \in A\}) (\text{Neighbours } C b \cap A)$
 unfolding *bij-betw-def*
proof
 have [simp]: $\text{the-elem } (\{x, b\} - \{b\}) = x$ if $x \in A$ for x
 using *insert.prem*s by (simp add: *disjnt-iff insert-Diff-if that*)
 show *inj-on* $(\lambda e. \text{the-elem } (e - \{b\})) (C \cap \{\{x, b\} \mid x. x \in A\})$
 by (auto simp: *inj-on-def*)
 show $(\lambda e. \text{the-elem } (e - \{b\})) \text{ ` } (C \cap \{\{x, b\} \mid x. x \in A\}) = \text{Neighbours } C b \cap A$
 by (fastforce simp: *Neighbours-def insert-commute image-iff Bex-def*)
qed
 have $(C \cap \text{all-edges-betw-un } A (\text{insert } b B)) = (C \cap (\{\{x, b\} \mid x. x \in A\} \cup \text{all-edges-betw-un } A B))$
 using $\langle C \subseteq E \rangle$ by (auto simp: *all-edges-betw-un-insert2*)
 then have $\text{edge-card } C A (\text{insert } b B) = \text{card } ((C \cap (\{\{x, b\} \mid x. x \in A\}) \cup (C \cap \text{all-edges-betw-un } A B))$
 by (simp add: *edge-card-def Int-Un-distrib*)
 also have $\dots = \text{card } (C \cap \{\{x, b\} \mid x. x \in A\}) + \text{card } (C \cap \text{all-edges-betw-un } A B)$
proof (rule *card-Un-disjnt*)
 show *disjnt* $(C \cap \{\{x, b\} \mid x. x \in A\}) (C \cap \text{all-edges-betw-un } A B)$
 using *insert* by (auto simp: *disjnt-iff all-edges-betw-un-def doubleton-eq-iff*)
qed (use $\langle \text{finite } C \rangle$ in auto)
 also have $\dots = \text{card } (\text{Neighbours } C b \cap A) + \text{card } (C \cap \text{all-edges-betw-un } A B)$
 using *bij-betw-same-card [OF bij]* by simp
 also have $\dots = (\sum i \in \text{insert } b B. \text{card } (\text{Neighbours } C i \cap A))$
 using *insert* by (simp add: *edge-card-def*)
 finally show ?case .
qed

lemma *sum-eq-card:* finite $A \implies (\sum x \in A. \text{if } x \in B \text{ then } 1 \text{ else } 0) = \text{card } (A \cap B)$

by (*metis (no-types, lifting) card-eq-sum sum.cong sum.inter-restrict*)

lemma *sum-eq-card-Neighbours*:
assumes $x \in V \ C \subseteq E$
shows $(\sum y \in V \setminus \{x\}. \text{if } \{x,y\} \in C \text{ then } 1 \text{ else } 0) = \text{card } (\text{Neighbours } C \ x)$
proof –
have $\text{Neighbours } C \ x = (V \setminus \{x\}) \cap \{y. \{x, y\} \in C\}$
using *assms wellformed* **by** (*auto simp: Neighbours-def*)
with *finV sum-eq-card [of - {y. {x,y} ∈ C}]* **show** *?thesis* **by** *simp*
qed

lemma *Neighbours-insert-NO-MATCH*: $\text{NO-MATCH } \{ \} \ C \implies \text{Neighbours } (\text{insert } e \ C) \ x = \text{Neighbours } \{e\} \ x \cup \text{Neighbours } C \ x$
by (*auto simp: Neighbours-def*)

lemma *Neighbours-sing-2*:
assumes $e \in E$
shows $(\sum x \in V. \text{card } (\text{Neighbours } \{e\} \ x)) = 2$
proof –
obtain $u \ v$ **where** $uv: e = \{u,v\} \ u \neq v$
by (*meson assms card-2-iff two-edges*)
then have $u \in V \ v \in V$
using *assms wellformed* **by** *blast+*
have $*$: $\text{Neighbours } \{e\} \ x = (\text{if } x=u \text{ then } \{v\} \ \text{else if } x=v \text{ then } \{u\} \ \text{else } \{ \})$ **for**
 x
by (*auto simp: Neighbours-def uv doubleton-eq-iff*)
show *?thesis*
using $\langle u \neq v \rangle$
by (*simp add: * if-distrib [of card] finV sum.delta-remove <u ∈ V> <v ∈ V>*
cong: if-cong)
qed

lemma *sum-Neighbours-eq-card*:
assumes *finite C C ⊆ E*
shows $(\sum i \in V. \text{card } (\text{Neighbours } C \ i)) = \text{card } C * 2$
using *assms*
proof (*induction C*)
case empty
then show *?case*
by (*auto simp: Neighbours-def*)
next
case (*insert e C*)
then have [*simp*]: $\text{Neighbours } \{e\} \ x \cap \text{Neighbours } C \ x = \{ \}$ **for** x
by (*auto simp: Neighbours-def*)
with *insert* **show** *?case*
by (*auto simp: card-Un-disjoint finite-Neighbours Neighbours-insert-NO-MATCH sum.distrib Neighbours-sing-2*)
qed

lemma *gen-density-empty* [simp]: $\text{gen-density } C \ \{\} \ X = 0$ $\text{gen-density } C \ X \ \{\} = 0$

by (*auto simp: gen-density-def*)

lemma *gen-density-commute*: $\text{gen-density } C \ X \ Y = \text{gen-density } C \ Y \ X$

by (*simp add: edge-card-commute gen-density-def*)

lemma *gen-density-ge0*: $\text{gen-density } C \ X \ Y \geq 0$

by (*auto simp: gen-density-def*)

lemma *gen-density-gt0*:

assumes *finite X finite Y* $\{x,y\} \in C$ $x \in X$ $y \in Y$ $C \subseteq E$

shows $\text{gen-density } C \ X \ Y > 0$

proof –

have *xy*: $\{x,y\} \in \text{all-edges-betw-un } X \ Y$

using *assms* **by** (*force simp: all-edges-betw-un-def*)

moreover **have** *finite* (*all-edges-betw-un X Y*)

by (*simp add: assms finite-all-edges-betw-un*)

ultimately **have** $\text{edge-card } C \ X \ Y > 0$

by (*metis IntI assms(3) card-0-eq edge-card-def emptyE finite-Int gr0I*)

with *xy* **show** *?thesis*

using *assms gen-density-def less-eq-real-def* **by** *fastforce*

qed

lemma *gen-density-le1*: $\text{gen-density } C \ X \ Y \leq 1$

unfolding *gen-density-def*

apply (*simp add: divide-simps mult-less-0-iff zero-less-mult-iff*)

by (*metis card-ge-0-finite edge-card-le of-nat-mono of-nat-mult*)

lemma *gen-density-le-1-minus*:

shows $\text{gen-density } C \ X \ Y \leq 1 - \text{gen-density } (E \setminus C) \ X \ Y$

proof (*cases finite X ^ finite Y*)

case *True*

have $C \cap \text{all-edges-betw-un } X \ Y \cup (E \setminus C) \cap \text{all-edges-betw-un } X \ Y = \text{all-edges-betw-un } X \ Y$

by (*auto simp: all-edges-betw-un-def*)

with *True* **have** $(\text{edge-card } C \ X \ Y) + (\text{edge-card } (E \setminus C) \ X \ Y) \leq \text{card } (\text{all-edges-betw-un } X \ Y)$

unfolding *edge-card-def*

by (*metis Diff-Int-distrib2 Diff-disjoint card-Un-disjoint card-Un-le finite-Int finite-all-edges-betw-un*)

with *True* **show** *?thesis*

apply (*simp add: gen-density-def divide-simps*)

by (*smt (verit) all-edges-betw-un-le of-nat-add of-nat-mono of-nat-mult*)

qed (*auto simp: gen-density-def*)

lemma *gen-density-lt1*:

assumes $\{x,y\} \in E \setminus C$ $x \in X$ $y \in Y$ $C \subseteq E$

shows $\text{gen-density } C \ X \ Y < 1$

proof (*cases finite X ∧ finite Y*)
case *True*
then have $0 < \text{gen-density } (E \setminus C) X Y$
using *assms gen-density-gt0* **by** *auto*
have $\text{gen-density } C X Y \leq 1 - \text{gen-density } (E \setminus C) X Y$
by (*intro gen-density-le-1-minus*)
then show *?thesis*
using $\langle 0 < \text{gen-density } (E \setminus C) X Y \rangle$ **by** *linarith*
qed (*auto simp: gen-density-def*)

lemma *gen-density-le-iff*:
assumes *disjnt X Z finite X Y ⊆ X Y ≠ {} finite Z*
shows $\text{gen-density } C X Z \leq \text{gen-density } C Y Z \iff$
 $\text{edge-card } C X Z / \text{card } X \leq \text{edge-card } C Y Z / \text{card } Y$
using *assms* **by** (*simp add: gen-density-def divide-simps mult-less-0-iff zero-less-mult-iff*)

"Removing vertices whose degree is less than the average can only increase the density from the remaining set" (page 17)

lemma *gen-density-below-avg-ge*:
assumes *disjnt X Z finite X Y ⊆ X finite Z*
and *genY: gen-density C Y Z ≤ gen-density C X Z*
shows $\text{gen-density } C (X \setminus Y) Z \geq \text{gen-density } C X Z$
proof –
have $\text{real } (\text{edge-card } C Y Z) / \text{card } Y \leq \text{real } (\text{edge-card } C X Z) / \text{card } X$
using *assms*
by (*force simp: gen-density-def divide-simps zero-less-mult-iff split: if-split-asm*)
have $\text{card } Y < \text{card } X$
by (*simp add: assms psubset-card-mono*)
have $*$: $\text{finite } Y Y \subseteq X X \neq \{\}$
using *assms finite-subset* **by** *blast+*
then
have $\text{card } X * \text{edge-card } C Y Z \leq \text{card } Y * \text{edge-card } C X Z$
using *genY assms*
by (*simp add: gen-density-def field-split-simps card-eq-0-iff flip: of-nat-mult split: if-split-asm*)
with *assms * <card Y < card X>* **show** *?thesis*
by (*simp add: gen-density-le-iff field-split-simps edge-card-diff card-Diff-subset edge-card-mono flip: of-nat-mult*)
qed

lemma *edge-card-insert*:
assumes *NO-MATCH {} F and e ∉ F*
shows $\text{edge-card } (\text{insert } e F) X Y = \text{edge-card } \{e\} X Y + \text{edge-card } F X Y$
proof –
have *fin: finite (all-edges-betw-un X Y)*
by (*meson all-uedges-betw-subset fin-edges finite-subset*)
have $\text{insert } e F \cap \text{all-edges-betw-un } X Y$
 $= \{e\} \cap \text{all-edges-betw-un } X Y \cup F \cap \text{all-edges-betw-un } X Y$
by *auto*

with $\langle e \notin F \rangle$ **show** *?thesis*
by (*auto simp: edge-card-def card-Un-disjoint disjoint-iff fin*)
qed

lemma *edge-card-sing*:
assumes $e \in E$
shows $\text{edge-card } \{e\} \ U \ U = (\text{if } e \subseteq U \text{ then } 1 \text{ else } 0)$
proof (*cases* $e \subseteq U$)
case *True*
obtain $x \ y$ **where** $xy: e = \{x, y\} \ x \neq y$
using *assms* **by** (*metis card-2-iff two-edges*)
with *True assms* **have** $\{e\} \cap \text{all-edges-betw-un } U \ U = \{e\}$
by (*auto simp: all-edges-betw-un-def*)
with *True* **show** *?thesis*
by (*simp add: edge-card-def*)
qed (*auto simp: edge-card-def all-edges-betw-un-def*)

lemma *sum-edge-card-choose*:
assumes $2 \leq k \ C \subseteq E$
shows $(\sum U \in [V]^k. \text{edge-card } C \ U \ U) = (\text{card } V - 2 \ \text{choose } (k-2)) * \text{card } C$
proof –
have $*$: $\text{card } \{A \in [V]^k. e \subseteq A\} = \text{card } V - 2 \ \text{choose } (k-2)$ **if** $e: e \in C$ **for** e
proof –
have $e \subseteq V$
using $\langle C \subseteq E \rangle$ ***e* wellformed** **by** *force*
obtain $x \ y$ **where** $xy: e = \{x, y\} \ x \neq y$
using $\langle C \subseteq E \rangle$ ***e* by** (*metis in-mono card-2-iff two-edges*)
define \mathcal{A} **where** $\mathcal{A} \equiv \{A \in [V]^k. e \subseteq A\}$
have $\bigwedge A. A \in \mathcal{A} \implies A = e \cup (A \setminus e) \wedge A \setminus e \in [V \setminus e]^{(k-2)}$
by (*auto simp: A-def nsets-def xy*)
moreover **have** $\bigwedge xa. \llbracket xa \in [V \setminus e]^{(k-2)} \rrbracket \implies e \cup xa \in \mathcal{A}$
using $\langle e \subseteq V \rangle$ *assms*
by (*auto simp: A-def nsets-def xy card-insert-if*)
ultimately **have** $\mathcal{A} = (\cup) e \cdot [V \setminus e]^{(k-2)}$
by *auto*
moreover **have** *inj-on* $((\cup) \ e) \ ([V \setminus e]^{(k-2)})$
by (*auto simp: inj-on-def nsets-def*)
moreover **have** $\text{card } (V \setminus e) = \text{card } V - 2$
by (*metis* $\langle C \subseteq E \rangle \ \langle e \in C \rangle$ *subsetD card-Diff-subset finV finite-subset two-edges wellformed*)
ultimately **show** *?thesis*
using *assms* **by** (*simp add: card-image A-def*)
qed
have $(\sum U \in [V]^k. \text{edge-card } R \ U \ U) = ((\text{card } V - 2) \ \text{choose } (k-2)) * \text{card } R$
if *finite* $R \ R \subseteq C$ **for** R
using *that*
proof (*induction* R)
case *empty*
then **show** *?case*

```

    by (simp add: edge-card-def)
next
case (insert e R)
with assms have e ∈ E by blast
with insert show ?case
by (simp add: edge-card-insert * sum.distrib edge-card-sing Ramsey.finite-imp-finite-nsets

      finV flip: sum.inter-filter)
qed
then show ?thesis
by (meson <C ⊆ E> fin-edges finite-subset set-eq-subset)
qed

```

```

lemma sum-nsets-Compl:
  assumes finite A k ≤ card A
  shows (∑ U ∈ [A]k. f (A \ U)) = (∑ U ∈ [A](card A - k). f U)
proof -
  have B ∈ ( \ ) A ‘ [A]k if B ∈ [A](card A - k) for B
  proof -
    have card (A \ B) = k
      using assms that by (simp add: nsets-def card-Diff-subset)
    moreover have B = A \ (A \ B)
      using that by (auto simp: nsets-def)
    ultimately show ?thesis
      using assms unfolding nsets-def image-iff by blast
  qed
  then have bij-betw (λ U. A \ U) ([A]k) ([A](card A - k))
    using assms by (auto simp: nsets-def bij-betw-def inj-on-def card-Diff-subset)
  then show ?thesis
    using sum.reindex-bij-betw by blast
qed

```

1.4 Lemma 9.2 preliminaries

Equation (45) in the text, page 30, is seemingly a huge gap. The development below relies on binomial coefficient identities.

definition *graph-density* $\equiv \lambda C. \text{card } C / \text{card } E$

```

lemma graph-density-Un:
  assumes disjnt C D C ⊆ E D ⊆ E
  shows graph-density (C ∪ D) = graph-density C + graph-density D
proof (cases card E > 0)
case True
with assms obtain finite C finite D
by (metis card-ge-0-finite finite-subset)
with assms show ?thesis
by (auto simp: graph-density-def card-Un-disjnt divide-simps)
qed (auto simp: graph-density-def)

```

Could be generalised to any complete graph

lemma *density-eq-average*:
assumes $C \subseteq E$ **and** *complete*: $E = \text{all-edges } V$
shows *graph-density* $C =$
 $\text{real } (\sum x \in V. \sum y \in V \setminus \{x\}. \text{if } \{x, y\} \in C \text{ then } 1 \text{ else } 0) / (\text{card } V * (\text{card } V - 1))$
proof –
have *cardE*: $\text{card } E = \text{card } V$ **choose** 2
using *card-all-edges complete finV* **by** *blast*
have *finite C*
using *assms fin-edges finite-subset* **by** *blast*
then have *: $(\sum x \in V. \sum y \in V \setminus \{x\}. \text{if } \{x, y\} \in C \text{ then } 1 \text{ else } 0) = \text{card } C * 2$
using *assms* **by** (*simp add: sum-eq-card-Neighbours sum-Neighbours-eq-card*)
show *?thesis*
by (*auto simp: graph-density-def divide-simps cardE choose-two-real **)
qed

lemma *edge-card-V-V*:
assumes $C \subseteq E$ **and** *complete*: $E = \text{all-edges } V$
shows *edge-card* $C V V = \text{card } C$
proof –
have $C \subseteq \text{all-edges-betw-un } V V$
using *assms clique-iff complete subset-refl*
by (*metis all-uedges-betw-I all-uedges-betw-subset clique-def*)
then show *?thesis*
by (*metis Int-absorb2 edge-card-def*)
qed

Bhavik's statement; own proof

proposition *density-eq-average-partition*:
assumes $k: 0 < k < \text{card } V$ **and** $C \subseteq E$ **and** *complete*: $E = \text{all-edges } V$
shows *graph-density* $C = (\sum U \in [V]^k. \text{gen-density } C U (V \setminus U)) / (\text{card } V \text{ choose } k)$
proof (*cases k=1 ∨ gorder = Suc k*)
case *True*
then have [*simp*]: $\text{gorder choose } k = \text{gorder}$ **by** *auto*
have $V \neq \{\}$
using *assms* **by** *force*
then have *nontriv*: $E \neq \{\}$
using *assms card-all-edges finV* **by** *force*
have *eq*: $(C \cap (\{\{x, y\} \mid y. y \in V \wedge y \neq x\} \cap E))$
 $= (\lambda y. \{x, y\}) ' \{y. \{x, y\} \in C\}$ **for** x
using $\langle C \subseteq E \rangle$ *wellformed* **by** *fastforce*
have $(\sum U \in [V]^k. \text{gen-density } C U (V \setminus U)) = (\sum x \in V. \text{gen-density } C \{x\} (V \setminus \{x\}))$
using *True*
proof
assume $k = 1$
then show *?thesis*

by (*simp add: sum-nsets-one*)
 next
 assume §: *gorder = Suc k*
 have $[V]^k \subseteq (\lambda x. V \setminus \{x\}) \text{ ' } V$
 unfolding *inj-on-def bij-betw-def nsets-def*
 proof *clarify*
 fix *A*
 assume *A: k = card A finite A A ⊆ V*
 then obtain *x* where $x \in V \setminus A$
 by (*metis assms(2) order-less-le psubset-imp-ex-mem*)
 then have $A = V \setminus \{x\}$
 using § *A*
 by (*metis Diff-insert-absorb card.insert card-subset-eq insert-subset subsetI*
finV Diff-iff)
 then show $A \in (\lambda x. V \setminus \{x\}) \text{ ' } V$
 using $\langle x \in V \setminus A \rangle$ by *blast*
 qed
 then have *bij: bij-betw* $(\lambda x. V \setminus \{x\}) V ([V]^k)$
 using *finV §* by (*auto simp add: inj-on-def bij-betw-def nsets-def*)
 moreover have $V \setminus (V \setminus \{x\}) = \{x\}$ if $x \in V$ for *x*
 using *that* by *auto*
 ultimately show *?thesis*
 using *sum.reindex-bij-betw [OF bij] gen-density-commute*
 by (*metis (no-types, lifting) sum.cong*)
 qed
 also have $\dots = (\sum x \in V. \text{real } (\text{edge-card } C \{x\} (V \setminus \{x\}))) / (\text{gorder} - 1)$
 by (*simp add: <C ⊆ E> gen-density-def flip: sum-divide-distrib*)
 also have $\dots = (\sum i \in V. \text{card } (\text{Neighbours } C i)) / (\text{gorder} - 1)$
 unfolding *edge-card-def Neighbours-def all-edges-betw-un-def*
 by (*simp add: eq card-image inj-on-def doubleton-eq-iff*)
 also have $\dots = \text{graph-density } C * \text{gorder}$
 using *assms density-eq-average [OF <C ⊆ E> complete]*
 by (*simp add: sum-eq-card-Neighbours*)
 finally show *?thesis*
 using *k* by *simp*
 next
 case *False*
 then have *K: gorder > Suc k k ≥ 2*
 using *assms* by *auto*
 then have $\text{gorder} - \text{Suc } (\text{Suc } (\text{gorder} - \text{Suc } (\text{Suc } k))) = k$
 using *assms* by *auto*
 then have [*simp*]: $\text{gorder} - 2 \text{ choose } (\text{gorder} - \text{Suc } (\text{Suc } k)) = (\text{gorder} - 2 \text{ choose } k)$
 using *binomial-symmetric [of (gorder - Suc (Suc k))]*
 by *simp*
 have *cardE: card E = card V choose 2*
 using *card-all-edges complete finV* by *blast*
 have *card E > 0*
 using *k cardE* by *auto*

have *in-E-iff* [*iff*]: $\{v,w\} \in E \iff v \in V \wedge w \in V \wedge v \neq w$ **for** $v w$
by (*auto simp: complete all-edges-alt doubleton-eq-iff*)

have B : $\text{edge-card } C \ V \ V = \text{edge-card } C \ U \ U + \text{edge-card } C \ U \ (V \setminus U) +$
 $\text{edge-card } C \ (V \setminus U) \ (V \setminus U)$
(is $?L = ?R$)
if $U \subseteq V$ **for** U
proof –
have *fin*: *finite* (*all-edges-betw-un* $U \ U'$) **for** U'
by (*meson all-uedges-betw-subset fin-edges finite-subset*)
have *dis*: *all-edges-betw-un* $U \ U \cap$ *all-edges-betw-un* $U \ (V \setminus U) = \{\}$
by (*auto simp: all-edges-betw-un-def doubleton-eq-iff*)
have *all-edges-betw-un* $V \ V =$ *all-edges-betw-un* $U \ U \cup$ *all-edges-betw-un* U
 $(V \setminus U) \cup$ *all-edges-betw-un* $(V \setminus U) \ (V \setminus U)$
by (*smt* (*verit*) *that Diff-partition Un-absorb Un-assoc all-edges-betw-un-Un2*
all-edges-betw-un-commute)
with *that* **have** $?L = \text{card } (C \cap$ *all-edges-betw-un* $U \ U \cup C \cap$ *all-edges-betw-un*
 $U \ (V \setminus U)$
 $\cup C \cap$ *all-edges-betw-un* $(V \setminus U) \ (V \setminus U))$
by (*simp add: edge-card-def Int-Un-distrib*)
also **have** $\dots = ?R$
using *fin dis* $\langle C \subseteq E \rangle$ *fin-edges finite-subset*
by (*(subst card-Un-disjoint)?, fastforce simp: edge-card-def all-edges-betw-un-def*
doubleton-eq-iff)
finally **show** *?thesis* .
qed
have C : $(\sum U \in [V]^k. \text{real } (\text{edge-card } C \ U \ (V \setminus U)))$
 $= (\text{card } V \ \text{choose } k) * \text{card } C - \text{real}(\sum U \in [V]^k. \text{edge-card } C \ U \ U + \text{edge-card}$
 $C \ (V \setminus U) \ (V \setminus U))$
(is $?L = ?R$)
proof –
have $?L = (\sum U \in [V]^k. \text{edge-card } C \ V \ V - \text{real } (\text{edge-card } C \ U \ U + \text{edge-card}$
 $C \ (V \setminus U) \ (V \setminus U)))$
unfolding *nsets-def* **by** (*rule sum.cong*) (*auto simp: B*)
also **have** $\dots = ?R$
using $\langle C \subseteq E \rangle$ *complete edge-card-V-V*
by (*simp add:* $\langle C \subseteq E \rangle$ *sum-subtractf edge-card-V-V*)
finally **show** *?thesis* .
qed

have $(\text{gorder-2 choose } k) + (\text{gorder-2 choose } (k-2)) + 2 * (\text{gorder-2 choose}$
 $(k-1)) = (\text{gorder choose } k)$
using *assms K* **by** (*auto simp: choose-reduce-nat [of gorder] choose-reduce-nat*
[of gorder-Suc 0] eval-nat-numeral)
moreover
have $(\text{gorder} - 1) * (\text{gorder-2 choose } (k-1)) = (\text{gorder} - k) * (\text{gorder-1 choose}$
 $(k-1))$
by (*metis Suc-1 Suc-diff-1 binomial-absorb-comp diff-Suc-eq-diff-pred* $\langle k > 0 \rangle$)
ultimately **have** F : $(\text{gorder} - 1) * (\text{gorder-2 choose } k) + (\text{gorder} - 1) *$

$(\text{gorder}-2 \text{ choose } (k-2)) + 2 * (\text{gorder}-k) * (\text{gorder}-1 \text{ choose } (k-1))$
 $= (\text{gorder} - 1) * (\text{gorder choose } k)$
by (*smt (verit) add-mult-distrib2 mult.assoc mult.left-commute*)

have $(\sum U \in [V]^k. \text{edge-card } C \ U \ (V \setminus U) / (\text{real } (\text{card } U) * \text{card } (V \setminus U)))$
 $= (\sum U \in [V]^k. \text{edge-card } C \ U \ (V \setminus U) / (\text{real } k * (\text{card } V - k)))$
using *card-Diff-subset* **by** (*intro sum.cong*) (*auto simp: nsets-def*)
also have $\dots = (\sum U \in [V]^k. \text{edge-card } C \ U \ (V \setminus U) / (k * (\text{card } V - k)))$
by (*simp add: sum-divide-distrib*)
finally have $*$: $(\sum U \in [V]^k. \text{edge-card } C \ U \ (V \setminus U) / (\text{real } (\text{card } U) * \text{card } (V \setminus U)))$
 $= (\sum U \in [V]^k. \text{edge-card } C \ U \ (V \setminus U) / (k * (\text{card } V - k)))$.

have *choose-m1*: $\text{gorder} * (\text{gorder} - 1 \text{ choose } (k - 1)) = k * (\text{gorder choose } k)$
using $\langle k > 0 \rangle$ *times-binomial-minus1-eq* **by** *presburger*
have $**$: $(\text{real } k * (\text{real } \text{gorder} - \text{real } k) * \text{real } (\text{gorder choose } k)) =$
 $(\text{real } (\text{gorder choose } k) - (\text{real } (\text{gorder} - 2 \text{ choose } (k - 2)) + \text{real } (\text{gorder} - 2 \text{ choose } k))) *$
 $\text{real } (\text{gorder choose } 2)$
using *assms K arg-cong [OF F, of $\lambda u. \text{real } \text{gorder} * \text{real } u$] arg-cong [OF choose-m1, of real]*
apply (*simp add: choose-two-real ring-distrib*)
by (*smt (verit) distrib-right mult.assoc mult-2-right mult-of-nat-commute*)
have *eq*: $(\sum U \in [V]^k. \text{real } (\text{edge-card } C \ (V \setminus U) \ (V \setminus U)))$
 $= (\sum U \in [V]^{(\text{gorder}-k)}. \text{real } (\text{edge-card } C \ U \ U))$
using *K finV* **by** (*subst sum-nsets-Compl, simp-all*)
show *?thesis*
unfolding *graph-density-def gen-density-def*
using *K $\langle \text{card } E > 0 \rangle \langle C \subseteq E \rangle$*
apply (*simp add: eq divide-simps B C sum.distrib **)
apply (*simp add: ** sum-edge-card-choose cardE flip: of-nat-sum*)
by *argo*

qed

lemma *exists-density-edge-density*:
assumes *k*: $0 < k < \text{card } V$ **and** $C \subseteq E$ **and** *complete*: $E = \text{all-edges } V$
obtains *U* **where** $\text{card } U = k$ $U \subseteq V$ *graph-density* $C \leq \text{gen-density } C \ U \ (V \setminus U)$
proof –
have *False* **if** $\bigwedge U. U \in [V]^k \implies \text{graph-density } C > \text{gen-density } C \ U \ (V \setminus U)$
proof –
have $\text{card}([V]^k) > 0$
using *assms* **by** *auto*
then have $(\sum U \in [V]^k. \text{gen-density } C \ U \ (V \setminus U)) < \text{card}([V]^k) * \text{graph-density } C$
 C
by (*meson sum-bounded-above-strict that*)
with *density-eq-average-partition assms* **show** *False* **by** *force*
qed
with *that* **show** *thesis*
unfolding *nsets-def* **by** *fastforce*

qed

end

end

2 The book algorithm

theory *Book* imports

Neighbours

HOL-Library.Disjoint-Sets HOL-Decision-Procs.Approximation

HOL-Real-Asymp.Real-Asymp

begin

hide-const *Bseq*

2.1 Locales for the parameters of the construction

type-synonym *'a config* = *'a set* × *'a set* × *'a set* × *'a set*

locale *P0-min* =

fixes *p0-min* :: *real*

assumes *p0-min*: $0 < p0-min$ $p0-min < 1$

locale *Book-Basis* = *fin-sgraph* + *P0-min* + — building on finite simple graphs
(no loops)

assumes *complete*: $E = \text{all-edges } V$

assumes *infinite-UNIV*: *infinite* (*UNIV*::*'a set*)

begin

abbreviation $nV \equiv \text{card } V$

lemma *graph-size*: $\text{graph-size} = (nV \text{ choose } 2)$

using *card-all-edges complete finV* **by** *blast*

lemma *in-E-iff* [*iff*]: $\{v, w\} \in E \longleftrightarrow v \in V \wedge w \in V \wedge v \neq w$

by (*auto simp: complete all-edges-alt doubleton-eq-iff*)

lemma *all-edges-betw-un-iff-clique*: $K \subseteq V \implies \text{all-edges-betw-un } K \iff K \subseteq F \longleftrightarrow \text{clique } K \iff F$

unfolding *clique-def all-edges-betw-un-def doubleton-eq-iff subset-iff*

by *blast*

lemma *clique-Un*:

assumes *clique A F clique B F all-edges-betw-un A B* $\subseteq F$ $A \subseteq V$ $B \subseteq V$

shows *clique (A ∪ B) F*

using *assms* **by** (*simp add: all-edges-betw-un clique-Un subset-iff*)

lemma *clique-insert*:
assumes *clique* A F *all-edges-betw-un* $\{x\}$ $A \subseteq F$ $A \subseteq V$ $x \in V$
shows *clique* $(\text{insert } x \ A)$ F
using *assms*
by (*metis Un-subset-iff clique-def insert-is-Un insert-subset clique-Un singletonD*)

lemma *less-RN-Red-Blue*:
fixes l k
assumes nV : $nV < RN$ k l
obtains Red $Blue$:: 'a set set
where $Red \subseteq E$ $Blue = E \setminus Red$ $\neg (\exists K. \text{size-clique } k \ K \ Red) \neg (\exists K. \text{size-clique } l \ K \ Blue)$
proof –
have \neg *is-Ramsey-number* k l nV
using *RN-le assms leD* **by** *blast*
then obtain f **where** $f: f \in \text{nsets } \{..<nV\} \ 2 \rightarrow \{..<2\}$
and *noclique*: $\bigwedge i. i < 2 \implies \neg \text{monochromatic } \{..<nV\} ([k,l] ! i) \ 2 \ f \ i$
by (*auto simp: partn-lst-def eval-nat-numeral*)
obtain φ **where** $\varphi: \text{bij-betw } \varphi \ \{..<nV\} \ V$
using *bij-betw-from-nat-into-finite finV* **by** *blast*
define ϑ **where** $\vartheta \equiv \text{inv-into } \{..<nV\} \ \varphi$
have $\vartheta: \text{bij-betw } \vartheta \ V \ \{..<nV\}$
using φ ϑ -*def* *bij-betw-inv-into* **by** *blast*
have *emap*: $\text{bij-betw } (\lambda e. \varphi' e) \ (\text{nsets } \{..<nV\} \ 2) \ E$
by (*metis* φ *bij-betw-nsets complete nsets2-eq-all-edges*)
define Red **where** $Red \equiv (\lambda e. \varphi' e) \ ' ((f \ -' \ \{0\}) \cap \text{nsets } \{..<nV\} \ 2)$
define $Blue$ **where** $Blue \equiv (\lambda e. \varphi' e) \ ' ((f \ -' \ \{1\}) \cap \text{nsets } \{..<nV\} \ 2)$
have $f0: f \ (\vartheta' e) = 0$ **if** $e \in Red$ **for** e
using *that* φ **by** (*auto simp add: Red-def image-iff* ϑ -*def* *bij-betw-def nsets-def*)
have $f1: f \ (\vartheta' e) = 1$ **if** $e \in Blue$ **for** e
using *that* φ **by** (*auto simp add: Blue-def image-iff* ϑ -*def* *bij-betw-def nsets-def*)
have $Red \subseteq E$
using *bij-betw-imp-surj-on[OF emap]* **by** (*auto simp: Red-def*)
have $Blue = E - Red$
using *emap* f
by (*auto simp: Red-def Blue-def bij-betw-def inj-on-eq-iff image-iff Pi-iff*)
have *no-Red-K*: *False* **if** *size-clique* k K Red **for** K
proof –
have *clique* K Red **and** $Kk: \text{card } K = k$ **and** $K \subseteq V$
using *that* **by** (*auto simp: size-clique-def*)
with $f0$ **have** $f \ ' [\vartheta' K]^2 \subseteq \{0\}$
unfolding *clique-def image-subset-iff*
by (*force simp: elim!: nsets2-E*)
moreover **have** $\vartheta' K \in [\{..<nV\}]^{\text{card } K}$
using $\langle K \subseteq V \rangle \ \vartheta$
by (*auto simp add: nsets-def bij-betw-def card-image finite-nat-iff-bounded inj-on-subset*)
ultimately show *False*
using *noclique [of 0]* Kk **by** (*simp add: size-clique-def monochromatic-def*)

qed
have *no-Blue-K*: *False* **if** *size-clique l K Blue* **for** *K*
proof –
have *clique K Blue* **and** *Kl: card K = l* **and** $K \subseteq V$
using *that* **by** (*auto simp: size-clique-def*)
with *f1* **have** $f' [\vartheta'K]^2 \subseteq \{1\}$
by (*force simp: clique-def nsets-def card-2-iff*)
moreover **have** $\vartheta'K \in [\{..<nV\}]^{\text{card } K}$
using *bij-betw-nsets [OF \vartheta] \langle K \subseteq V \rangle* *bij-betwE finV infinite-super nsets-def*
by *fastforce*
ultimately **show** *False*
using *noclique [of 1] Kl* **by** (*simp add: size-clique-def monochromatic-def*)
qed
show *thesis*
using $\langle \text{Blue} = E \setminus \text{Red} \rangle \langle \text{Red} \subseteq E \rangle$ *no-Blue-K no-Red-K* **that** **by** *presburger*
qed

end

locale *No-Cliques* = *Book-Basis* +
fixes *Red Blue* :: 'a set set
assumes *Red-E*: $\text{Red} \subseteq E$
assumes *Blue-def*: $\text{Blue} = E - \text{Red}$
— the following are local to the program
fixes *l::nat* — blue limit
fixes *k::nat* — red limit
assumes *l-le-k*: $l \leq k$ — they should be "sufficiently large"
assumes *no-Red-clique*: $\neg (\exists K. \text{size-clique } k K \text{ Red})$
assumes *no-Blue-clique*: $\neg (\exists K. \text{size-clique } l K \text{ Blue})$

locale *Book* = *Book-Basis* + *No-Cliques* +
fixes $\mu::\text{real}$ — governs the big blue steps
assumes $\mu 01$: $0 < \mu \mu < 1$
fixes *X0* :: 'a set **and** *Y0* :: 'a set — initial values
assumes *XY0*: $\text{disjnt } X0 Y0 X0 \subseteq V Y0 \subseteq V$
assumes *density-ge-p0-min*: $\text{gen-density } \text{Red } X0 Y0 \geq p0\text{-min}$

locale *Book'* = *Book-Basis* + *No-Cliques* +
fixes $\gamma::\text{real}$ — governs the big blue steps
assumes $\gamma\text{-def}$: $\gamma = \text{real } l / (\text{real } k + \text{real } l)$
fixes *X0* :: 'a set **and** *Y0* :: 'a set — initial values
assumes *XY0*: $\text{disjnt } X0 Y0 X0 \subseteq V Y0 \subseteq V$
assumes *density-ge-p0-min*: $\text{gen-density } \text{Red } X0 Y0 \geq p0\text{-min}$

definition $\text{eps} \equiv \lambda k. \text{real } k \text{ powr } (-1/4)$

definition $\text{qfun-base} :: [\text{nat}, \text{nat}] \Rightarrow \text{real}$
where $\text{qfun-base} \equiv \lambda k h. ((1 + \text{eps } k) ^ h - 1) / k$

definition *hgt-maximum* $\equiv \lambda k. 2 * \ln (\text{real } k) / \text{eps } k$

The first of many "bigness assumptions"

definition *Big-height-upper-bound* $\equiv \lambda k. \text{qfun-base } k (\text{nat } \lfloor \text{hgt-maximum } k \rfloor) > 1$

lemma *Big-height-upper-bound*:

shows $\forall^\infty k. \text{Big-height-upper-bound } k$

unfolding *Big-height-upper-bound-def hgt-maximum-def eps-def qfun-base-def*
by *real-asymp*

context *No-Cliques*

begin

abbreviation $\varepsilon \equiv \text{eps } k$

lemma *eps-eq-sqrt*: $\varepsilon = 1 / \text{sqrt } (\text{sqrt } (\text{real } k))$

by (*simp add: eps-def powr-minus-divide powr-powr flip: powr-half-sqrt*)

lemma *eps-ge0*: $\varepsilon \geq 0$

by (*simp add: eps-def*)

lemma *ln0*: $l > 0$

using *no-Blue-clique* **by** (*force simp: size-clique-def clique-def*)

lemma *kn0*: $k > 0$

using *l-le-k ln0* **by** *auto*

lemma *eps-gt0*: $\varepsilon > 0$

by (*simp add: eps-def kn0*)

lemma *eps-le1*: $\varepsilon \leq 1$

using *kn0 ge-one-powr-ge-zero*

by (*simp add: eps-def powr-minus powr-mono2 divide-simps*)

lemma *eps-less1*:

assumes $k > 1$ **shows** $\varepsilon < 1$

using *assms powr-less-one* **by** (*auto simp: eps-def*)

lemma *Blue-E*: $\text{Blue} \subseteq E$

by (*simp add: Blue-def*)

lemma *disjnt-Red-Blue*: $\text{disjnt } \text{Red } \text{Blue}$

by (*simp add: Blue-def disjnt-def*)

lemma *Red-Blue-all*: $\text{Red} \cup \text{Blue} = \text{all-edges } V$

using *Blue-def Red-E complete* **by** *blast*

lemma *Blue-eq*: $\text{Blue} = \text{all-edges } V - \text{Red}$

using *Blue-def complete* **by** *auto*

lemma *Red-eq*: $Red = all_edges\ V - Blue$
using *Blue-eq Red-Blue-all* **by** *blast*

lemma *disjnt-Red-Blue-Neighbours*: $disjnt\ (Neighbours\ Red\ x\ \cap\ X)\ (Neighbours\ Blue\ x\ \cap\ X')$
using *disjnt-Red-Blue* **by** (*auto simp: disjnt-def Neighbours-def*)

lemma *indep-Red-iff-clique-Blue*: $K \subseteq V \implies indep\ K\ Red \longleftrightarrow clique\ K\ Blue$
using *Blue-eq* **by** *auto*

lemma *Red-Blue-RN*:
fixes $X :: 'a\ set$
assumes $card\ X \geq RN\ m\ n\ X \subseteq V$
shows $\exists K \subseteq X. size_clique\ m\ K\ Red \vee size_clique\ n\ K\ Blue$
using *partn-lst-imp-is-clique-RN [OF is-Ramsey-number-RN [of m n]]* *assms*
indep-Red-iff-clique-Blue
unfolding *is-clique-RN-def size-clique-def clique-indep-def*
by (*metis finV finite-subset subset-eq*)

end

context *Book*
begin

lemma *Red-edges-XY0*: $Red \cap all_edges_betw_un\ X0\ Y0 \neq \{\}$
using *density-ge-p0-min p0-min*
by (*auto simp: gen-density-def edge-card-def*)

lemma *finite-X0*: $finite\ X0$ **and** *finite-Y0*: $finite\ Y0$
using *XY0 finV finite-subset* **by** *blast+*

lemma *Red-nonempty*: $Red \neq \{\}$
using *Red-edges-XY0* **by** *blast*

lemma *gorder-ge2*: $gorder \geq 2$
using *Red-nonempty*
by (*metis Red-E card-mono equalsOI finV subset-empty two-edges wellformed*)

lemma *nontriv*: $E \neq \{\}$
using *Red-E Red-nonempty* **by** *force*

lemma *no-singleton-Blue* [*simp*]: $\{a\} \notin Blue$
using *Blue-E* **by** *auto*

lemma *no-singleton-Red* [*simp*]: $\{a\} \notin Red$
using *Red-E* **by** *auto*

lemma *not-Red-Neighbour* [*simp*]: $x \notin Neighbours\ Red\ x$ **and** *not-Blue-Neighbour*

[simp]: $x \notin \text{Neighbours Blue } x$
using *Red-E Blue-E not-own-Neighbour* **by** *auto*

lemma *Neighbours-RB*:
assumes $a \in V \ X \subseteq V$
shows $\text{Neighbours Red } a \cap X \cup \text{Neighbours Blue } a \cap X = X - \{a\}$
using *assms Red-Blue-all complete singleton-not-edge*
by (*fastforce simp: Neighbours-def*)

lemma *Neighbours-Red-Blue*:
assumes $x \in V$
shows $\text{Neighbours Red } x = V - \text{insert } x (\text{Neighbours Blue } x)$
using *Red-E assms* **by** (*auto simp: Blue-eq Neighbours-def complete all-edges-def*)

abbreviation $\text{red-density } X \ Y \equiv \text{gen-density Red } X \ Y$
abbreviation $\text{blue-density } X \ Y \equiv \text{gen-density Blue } X \ Y$

definition $\text{Weight} :: ['a \ \text{set}, 'a \ \text{set}, 'a, 'a] \Rightarrow \text{real}$ **where**
 $\text{Weight} \equiv \lambda X \ Y \ x \ y. \text{inverse } (\text{card } Y) * (\text{card } (\text{Neighbours Red } x \cap \text{Neighbours Red } y \cap Y) - \text{red-density } X \ Y * \text{card } (\text{Neighbours Red } x \cap Y))$

definition $\text{weight} :: 'a \ \text{set} \Rightarrow 'a \ \text{set} \Rightarrow 'a \Rightarrow \text{real}$ **where**
 $\text{weight} \equiv \lambda X \ Y \ x. \sum y \in X \setminus \{x\}. \text{Weight } X \ Y \ x \ y$

definition $p0 :: \text{real}$
where $p0 \equiv \text{red-density } X0 \ Y0$

definition $\text{qfun} :: \text{nat} \Rightarrow \text{real}$
where $\text{qfun} \equiv \lambda h. p0 + \text{qfun-base } k \ h$

lemma *qfun-eq*: $\text{qfun} \equiv \lambda h. p0 + ((1 + \varepsilon)^h - 1) / k$
by (*simp add: qfun-def qfun-base-def eps-def*)

definition $\text{hgt} :: \text{real} \Rightarrow \text{nat}$
where $\text{hgt} \equiv \lambda p. \text{LEAST } h. p \leq \text{qfun } h \wedge h > 0$

lemma *qfun0* [simp]: $\text{qfun } 0 = p0$
by (*simp add: qfun-eq*)

lemma *p0-ge*: $p0 \geq p0\text{-min}$
using *density-ge-p0-min* **by** (*simp add: p0-def*)

lemma *card-XY0*: $\text{card } X0 > 0 \ \text{card } Y0 > 0$
using *Red-edges-XY0 finite-X0 finite-Y0* **by** *force+*

lemma *finite-Red* [simp]: finite Red
by (*metis Red-Blue-all complete fin-edges finite-Un*)

lemma *finite-Blue* [*simp*]: *finite Blue*
using *Blue-E fin-edges finite-subset* **by** *blast*

lemma *Red-edges-nonzero*: *edge-card Red X0 Y0 > 0*
using *Red-edges-XY0*
using *Red-E edge-card-def fin-edges finite-subset* **by** *fastforce*

lemma *p0-01*: $0 < p0 \ p0 \leq 1$
proof –
show $0 < p0$
using *Red-edges-nonzero card-XY0*
by (*auto simp: p0-def gen-density-def divide-simps mult-less-0-iff*)
show $p0 \leq 1$
by (*simp add: gen-density-le1 p0-def*)
qed

lemma *qfun-strict-mono*: $h' < h \implies \text{qfun } h' < \text{qfun } h$
by (*simp add: divide-strict-right-mono eps-gt0 kn0 qfun-eq*)

lemma *qfun-mono*: $h' \leq h \implies \text{qfun } h' \leq \text{qfun } h$
by (*metis less-eq-real-def nat-less-le qfun-strict-mono*)

lemma *q-Suc-diff*: $\text{qfun } (\text{Suc } h) - \text{qfun } h = \varepsilon * (1 + \varepsilon)^h / k$
by (*simp add: qfun-eq field-split-simps*)

lemma *height-exists'*:
obtains *h* **where** $p \leq \text{qfun-base } k \ h \wedge h > 0$
proof –
have $1: 1 + \varepsilon \geq 1$
by (*auto simp: eps-def*)
have $\forall^\infty h. p \leq \text{real } h * \varepsilon / \text{real } k$
using *p0-01 kn0 unfolding eps-def* **by** *real-asymp*
then obtain *h* **where** $k * p \leq \text{real } h * \varepsilon$
using *kn0* **by** (*force simp add: eventually-sequentially field-simps*)
also have $\dots \leq ((1 + \varepsilon)^h - 1)$
using *linear-plus-1-le-power* [*of* $\varepsilon \ h$]
by (*auto simp: eps-def add-ac*)
also have $\dots \leq ((1 + \varepsilon)^{\text{Suc } h} - 1)$
using *power-increasing* [*OF le-SucI* [*OF order-refl*] 1] **by** *simp*
finally have $p \leq \text{qfun-base } k \ (\text{Suc } h)$
using *kn0* **by** (*simp add: qfun-base-def eps-def field-simps*)
then show *thesis*
using *that* **by** *blast*
qed

lemma *height-exists*:
obtains *h* **where** $p \leq \text{qfun } h \ h > 0$
using *height-exists'* [*of* p] *p0-01*

by (*metis add.commute add-increasing2 less-eq-real-def qfun-def*)

lemma *hgt-gt0*: $hgt\ p > 0$
unfolding *hgt-def height-exists kn0*
by (*metis (no-types, lifting) LeastI-ex height-exists*)

lemma *hgt-works*: $p \leq qfun\ (hgt\ p)$
by (*metis (no-types, lifting) LeastI height-exists hgt-def*)

lemma *hgt-Least*:
assumes $0 < h\ p \leq qfun\ h$
shows $hgt\ p \leq h$
by (*simp add: Suc-leI assms hgt-def Least-le*)

lemma *real-hgt-Least*:
assumes $real\ h \leq r\ 0 < h\ p \leq qfun\ h$
shows $real\ (hgt\ p) \leq r$
using *assms* **by** (*meson assms order.trans hgt-Least of-nat-mono*)

lemma *hgt-greater*:
assumes $p > qfun\ h$
shows $hgt\ p > h$
by (*meson assms hgt-works kn0 not-less order.trans qfun-mono*)

lemma *hgt-less-imp-qfun-less*:
assumes $0 < h\ h < hgt\ p$
shows $p > qfun\ h$
by (*metis assms hgt-Least not-le*)

lemma *hgt-le-imp-qfun-ge*:
assumes $hgt\ p \leq h$
shows $p \leq qfun\ h$
by (*meson assms hgt-greater not-less*)

This gives us an upper bound for heights, namely *hgt 1*, but it's not explicit.

lemma *hgt-mono*:
assumes $p \leq q$
shows $hgt\ p \leq hgt\ q$
by (*meson assms order.trans hgt-Least hgt-gt0 hgt-works*)

lemma *hgt-mono'*:
assumes $hgt\ p < hgt\ q$
shows $p < q$
by (*smt (verit) assms hgt-mono leD*)

The upper bound of the height $h(p)$ appears just below (5) on page 9. Although we can bound all Heights by monotonicity (since $p \leq 1$), we need to exhibit a specific $o(k)$ function.

lemma *height-upper-bound*:

assumes $p \leq 1$ **and** *big*: *Big-height-upper-bound* k

shows $\text{hgt } p \leq 2 * \ln k / \varepsilon$

proof (*intro* *real-hgt-Least* [**where** $h = \text{nat } (\text{floor } (\text{hgt-maximum } k))$])

show $\text{real } (\text{nat } \lfloor \text{hgt-maximum } k \rfloor) \leq 2 * \ln (\text{real } k) / \varepsilon$

by (*simp* *add*: *hgt-maximum-def* *eps-def* *floor-divide-lower* *divide-simps*)

show $0 < \text{nat } \lfloor \text{hgt-maximum } k \rfloor$

using *big* *qfun0* *not-le* **by** (*force* *simp*: *Big-height-upper-bound-def* *qfun-def*)

show $p \leq \text{qfun } (\text{nat } \lfloor \text{hgt-maximum } k \rfloor)$

using *assms* *p0-01* **by** (*auto* *simp*: *Big-height-upper-bound-def* *qfun-def*)

qed

definition *alpha* :: $\text{nat} \Rightarrow \text{real}$ **where** $\text{alpha} \equiv \lambda h. \text{qfun } h - \text{qfun } (h-1)$

lemma *alpha-ge0*: $\text{alpha } h \geq 0$

by (*simp* *add*: *alpha-def* *qfun-eq* *divide-le-cancel* *eps-gt0*)

lemma *alpha-Suc-ge*: $\text{alpha } (\text{Suc } h) \geq \varepsilon / k$

proof –

have $(1 + \varepsilon) ^ h \geq 1$

by (*simp* *add*: *eps-def*)

then show *?thesis*

by (*simp* *add*: *alpha-def* *qfun-eq* *eps-gt0* *field-split-simps*)

qed

lemma *alpha-ge*: $h > 0 \implies \text{alpha } h \geq \varepsilon / k$

by (*metis* *Suc-pred* *alpha-Suc-ge*)

lemma *alpha-gt0*: $h > 0 \implies \text{alpha } h > 0$

by (*meson* *alpha-ge* *less-le-trans* *divide-pos-pos* *eps-gt0* *kn0* *of-nat-0-less-iff*)

lemma *alpha-Suc-eq*: $\text{alpha } (\text{Suc } h) = \varepsilon * (1 + \varepsilon) ^ h / k$

by (*simp* *add*: *alpha-def* *q-Suc-diff*)

lemma *alpha-eq*:

assumes $h > 0$ **shows** $\text{alpha } h = \varepsilon * (1 + \varepsilon) ^ (h-1) / k$

by (*metis* *Suc-pred'* *alpha-Suc-eq* *assms*)

lemma *alpha-hgt-eq*: $\text{alpha } (\text{hgt } p) = \varepsilon * (1 + \varepsilon) ^ (\text{hgt } p - 1) / k$

using *alpha-eq* *hgt-gt0* **by** *presburger*

lemma *alpha-mono*: $\llbracket h' \leq h; 0 < h \rrbracket \implies \text{alpha } h' \leq \text{alpha } h$

by (*simp* *add*: *alpha-eq* *eps-ge0* *divide-right-mono* *mult-left-mono* *power-increasing*)

definition *all-incident-edges* :: 'a set \Rightarrow 'a set set **where**

$\text{all-incident-edges} \equiv \lambda A. \bigcup v \in A. \text{incident-edges } v$

lemma *all-incident-edges-Un* [*simp*]: $\text{all-incident-edges } (A \cup B) = \text{all-incident-edges } A \cup \text{all-incident-edges } B$

by (auto simp: all-incident-edges-def)

end

context Book

begin

2.2 State invariants

definition $V\text{-state} \equiv \lambda(X, Y, A, B). X \subseteq V \wedge Y \subseteq V \wedge A \subseteq V \wedge B \subseteq V$

definition $\text{disjoint-state} \equiv \lambda(X, Y, A, B). \text{disjnt } X \ Y \wedge \text{disjnt } X \ A \wedge \text{disjnt } X \ B \wedge \text{disjnt } Y \ A \wedge \text{disjnt } Y \ B \wedge \text{disjnt } A \ B$

previously had all edges incident to A, B

definition $RB\text{-state} \equiv \lambda(X, Y, A, B). \text{all-edges-betw-un } A \ A \subseteq \text{Red} \wedge \text{all-edges-betw-un } A \ (X \cup Y) \subseteq \text{Red} \\ \wedge \text{all-edges-betw-un } B \ (B \cup X) \subseteq \text{Blue}$

definition $\text{valid-state} \equiv \lambda U. V\text{-state } U \wedge \text{disjoint-state } U \wedge RB\text{-state } U$

definition $\text{termination-condition} \equiv \lambda X \ Y. \text{card } X \leq RN \ k \ (\text{nat } \lceil \text{real } l \ \text{powr } (3/4) \rceil) \vee \text{red-density } X \ Y \leq 1/k$

lemma

assumes $V\text{-state}(X, Y, A, B)$

shows $\text{fin}X: \text{finite } X$ and $\text{fin}Y: \text{finite } Y$ and $\text{fin}A: \text{finite } A$ and $\text{fin}B: \text{finite } B$

using $V\text{-state-def}$ assms $\text{fin } V$ finite-subset by auto

lemma

assumes $\text{valid-state}(X, Y, A, B)$

shows $A\text{-Red-clique}: \text{clique } A \ \text{Red}$ and $B\text{-Blue-clique}: \text{clique } B \ \text{Blue}$

using assms

by (auto simp: valid-state-def $V\text{-state-def}$ $RB\text{-state-def}$ $\text{all-edges-betw-un-iff-clique}$ $\text{all-edges-betw-un-Un2}$)

lemma $A\text{-less-}k$:

assumes $\text{valid}: \text{valid-state}(X, Y, A, B)$

shows $\text{card } A < k$

using assms $A\text{-Red-clique}[OF \ \text{valid}]$ no-Red-clique **unfolding** valid-state-def $V\text{-state-def}$

by (metis $\text{nat-neq-iff prod.case size-clique-def size-clique-smaller}$)

lemma $B\text{-less-}l$:

assumes $\text{valid}: \text{valid-state}(X, Y, A, B)$

shows $\text{card } B < l$

using assms $B\text{-Blue-clique}[OF \ \text{valid}]$ no-Blue-clique **unfolding** valid-state-def $V\text{-state-def}$

by (metis $\text{nat-neq-iff prod.case size-clique-def size-clique-smaller}$)

2.3 Degree regularisation

definition *red-dense* $\equiv \lambda Y p x. \text{card} (\text{Neighbours Red } x \cap Y) \geq (p - \varepsilon \text{ powr } (-1/2) * \text{alpha} (\text{hgt } p)) * \text{card } Y$

definition *X-degree-reg* $\equiv \lambda X Y. \{x \in X. \text{red-dense } Y (\text{red-density } X Y) x\}$

definition *degree-reg* $\equiv \lambda(X, Y, A, B). (X\text{-degree-reg } X Y, Y, A, B)$

lemma *X-degree-reg-subset*: $X\text{-degree-reg } X Y \subseteq X$
by (*auto simp*: *X-degree-reg-def*)

lemma *degree-reg-V-state*: $V\text{-state } U \implies V\text{-state } (\text{degree-reg } U)$
by (*auto simp*: *degree-reg-def X-degree-reg-def V-state-def*)

lemma *degree-reg-disjoint-state*: $\text{disjoint-state } U \implies \text{disjoint-state } (\text{degree-reg } U)$
by (*auto simp*: *degree-reg-def X-degree-reg-def disjoint-state-def disjnt-iff*)

lemma *degree-reg-RB-state*: $RB\text{-state } U \implies RB\text{-state } (\text{degree-reg } U)$
using *all-edges-betw-un-mono2* [*OF X-degree-reg-subset*]
by (*force simp add*: *degree-reg-def RB-state-def all-edges-betw-un-Un2*)

lemma *degree-reg-valid-state*: $\text{valid-state } U \implies \text{valid-state } (\text{degree-reg } U)$
by (*simp add*: *degree-reg-RB-state degree-reg-V-state degree-reg-disjoint-state valid-state-def*)

lemma *not-red-dense-sum-less*:

assumes $\bigwedge x. x \in X \implies \neg \text{red-dense } Y p x$ **and** $X \neq \{\}$ *finite* X
shows $(\sum_{x \in X}. \text{card} (\text{Neighbours Red } x \cap Y)) < p * \text{real} (\text{card } Y) * \text{card } X$

proof –

have $\bigwedge x. x \in X \implies \text{card} (\text{Neighbours Red } x \cap Y) < p * \text{real} (\text{card } Y)$

using *assms*

unfolding *red-dense-def*

by (*smt (verit) alpha-ge0 mult-right-mono of-nat-0-le-iff powr-ge-zero zero-le-mult-iff*)

with $\langle X \neq \{\} \rangle$ **show** *?thesis*

by (*smt (verit) <finite X> of-nat-sum sum-strict-mono mult-of-nat-commute sum-constant*)

qed

lemma *red-density-X-degree-reg-ge*:

assumes *disjnt* $X Y$

shows $\text{red-density } (X\text{-degree-reg } X Y) Y \geq \text{red-density } X Y$

proof (*cases* $X = \{\} \vee \text{infinite } X \vee \text{infinite } Y$)

case *True*

then show *?thesis*

by (*force simp*: *gen-density-def X-degree-reg-def*)

next

case *False*

then have *finite* X *finite* Y

by *auto*

{ assume $\bigwedge x. x \in X \implies \neg \text{red-dense } Y (\text{red-density } X Y) x$

```

    with False have  $(\sum_{x \in X}. \text{card} (\text{Neighbours Red } x \cap Y)) < \text{red-density } X Y$ 
  * real (card Y) * card X
    using <finite X> not-red-dense-sum-less by blast
    with Red-E have edge-card Red Y X < (red-density X Y * real (card Y)) *
  card X
    by (metis False assms disjnt-sym edge-card-eq-sum-Neighbours)
    then have False
    by (simp add: gen-density-def edge-card-commute split: if-split-asm)
  }
  then obtain x where x:  $x \in X$  red-dense Y (red-density X Y) x
    by blast
  define X' where  $X' \equiv \{x \in X. \neg \text{red-dense } Y (\text{red-density } X Y) x\}$ 
  have X': finite X' disjnt Y X'
    using assms <finite X> by (auto simp: X'-def disjnt-iff)
  have eq: X-degree-reg X Y = X - X'
    by (auto simp: X-degree-reg-def X'-def)
  show ?thesis
  proof (cases X' = {})
    case True
    then show ?thesis
      by (simp add: eq)
  next
  case False
  show ?thesis
    unfolding eq
  proof (rule gen-density-below-avg-ge)
    have  $(\sum_{x \in X'}. \text{card} (\text{Neighbours Red } x \cap Y)) < \text{red-density } X Y * \text{real} (\text{card } Y) * \text{card } X'$ 
  proof (intro not-red-dense-sum-less)
    fix x
    assume  $x \in X'$ 
    show  $\neg \text{red-dense } Y (\text{red-density } X Y) x$ 
      using < $x \in X'$ > by (simp add: X'-def)
  qed (use False X' in auto)
  then have card X *  $(\sum_{x \in X'}. \text{card} (\text{Neighbours Red } x \cap Y)) < \text{card } X' * \text{edge-card Red Y X}$ 
  by (simp add: gen-density-def mult.commute divide-simps edge-card-commute flip: of-nat-sum of-nat-mult split: if-split-asm)
  then have card X *  $(\sum_{x \in X'}. \text{card} (\text{Neighbours Red } x \cap Y)) \leq \text{card } X' * (\sum_{x \in X}. \text{card} (\text{Neighbours Red } x \cap Y))$ 
  using assms Red-E
  by (metis <finite X> disjnt-sym edge-card-eq-sum-Neighbours nless-le)
  then have red-density Y X'  $\leq \text{red-density } Y X$ 
  using assms X' False <finite X>
  apply (simp add: gen-density-def edge-card-eq-sum-Neighbours disjnt-commute Red-E)
  apply (simp add: X'-def field-split-simps flip: of-nat-sum of-nat-mult)
  done
  then show red-density X' Y  $\leq \text{red-density } X Y$ 

```

by (*simp add: X'-def gen-density-commute*)
qed (*use assms x <finite X> <finite Y> X'-def in auto*)
qed
qed

2.4 Big blue steps: code

definition *bluish* :: [*'a set, 'a*] \Rightarrow *bool* **where**
bluish $\equiv \lambda X x. \text{card } (\text{Neighbours Blue } x \cap X) \geq \mu * \text{real } (\text{card } X)$

definition *many-bluish* :: *'a set* \Rightarrow *bool* **where**
many-bluish $\equiv \lambda X. \text{card } \{x \in X. \text{bluish } X x\} \geq \text{RN } k \text{ (nat } \lceil l \text{ powr } (2/3) \rceil)$

definition *good-blue-book* :: [*'a set, 'a set* \times *'a set*] \Rightarrow *bool* **where**
good-blue-book $\equiv \lambda X. \lambda(S, T). \text{book } S T \text{ Blue} \wedge S \subseteq X \wedge T \subseteq X \wedge \text{card } T \geq (\mu \wedge \text{card } S) * \text{card } X / 2$

lemma *ex-good-blue-book*: *good-blue-book* *X* ($\{\}$, *X*)
by (*simp add: good-blue-book-def book-def*)

lemma *bounded-good-blue-book*: $\llbracket \text{good-blue-book } X (S, T); \text{finite } X \rrbracket \Longrightarrow \text{card } S \leq \text{card } X$
by (*simp add: card-mono finX good-blue-book-def*)

definition *best-blue-book-card* :: *'a set* \Rightarrow *nat* **where**
best-blue-book-card $\equiv \lambda X. \text{GREATEST } s. \exists S T. \text{good-blue-book } X (S, T) \wedge s = \text{card } S$

lemma *best-blue-book-is-best*: $\llbracket \text{good-blue-book } X (S, T); \text{finite } X \rrbracket \Longrightarrow \text{card } S \leq \text{best-blue-book-card } X$
unfolding *best-blue-book-card-def*
by (*smt (verit) Greatest-le-nat bounded-good-blue-book [of X]*)

lemma *ex-best-blue-book*: *finite* *X* $\Longrightarrow \exists S T. \text{good-blue-book } X (S, T) \wedge \text{card } S = \text{best-blue-book-card } X$
unfolding *best-blue-book-card-def*
by (*smt (verit) GreatestI-ex-nat bounded-good-blue-book ex-good-blue-book*)

definition *choose-blue-book* $\equiv \lambda(X, Y, A, B). @ (S, T). \text{good-blue-book } X (S, T) \wedge \text{card } S = \text{best-blue-book-card } X$

lemma *choose-blue-book-works*:
 $\llbracket \text{finite } X; (S, T) = \text{choose-blue-book } (X, Y, A, B) \rrbracket$
 $\Longrightarrow \text{good-blue-book } X (S, T) \wedge \text{card } S = \text{best-blue-book-card } X$
unfolding *choose-blue-book-def*
using *someI-ex [OF ex-best-blue-book]*
by (*metis (mono-tags, lifting) case-prod-conv someI-ex*)

lemma *choose-blue-book-subset*:

$\llbracket \text{finite } X; (S, T) = \text{choose-blue-book } (X, Y, A, B) \rrbracket \implies S \subseteq X \wedge T \subseteq X \wedge \text{disjnt } S \ T$

using *choose-blue-book-works good-blue-book-def book-def* **by** *fastforce*

expressing the complicated preconditions inductively

inductive *big-blue*

where $\llbracket \text{many-bluish } X; \text{good-blue-book } X \ (S, T); \text{card } S = \text{best-blue-book-card } X \rrbracket$
 $\implies \text{big-blue } (X, Y, A, B) \ (T, Y, A, B \cup S)$

lemma *big-blue-V-state*: $\llbracket \text{big-blue } U \ U'; \text{V-state } U \rrbracket \implies \text{V-state } U'$

by (*force simp: good-blue-book-def V-state-def elim!: big-blue.cases*)

lemma *big-blue-disjoint-state*: $\llbracket \text{big-blue } U \ U'; \text{disjoint-state } U \rrbracket \implies \text{disjoint-state } U'$

by (*elim big-blue.cases*) (*auto simp: book-def disjnt-iff good-blue-book-def disjoint-state-def*)

lemma *big-blue-RB-state*: $\llbracket \text{big-blue } U \ U'; \text{RB-state } U \rrbracket \implies \text{RB-state } U'$

apply (*clarsimp simp add: good-blue-book-def book-def RB-state-def all-edges-betw-un-Un1 all-edges-betw-un-Un2 elim!: big-blue.cases*)

by (*metis all-edges-betw-un-commute all-edges-betw-un-mono1 le-supI2 sup.orderE*)

lemma *big-blue-valid-state*: $\llbracket \text{big-blue } U \ U'; \text{valid-state } U \rrbracket \implies \text{valid-state } U'$

by (*meson big-blue-RB-state big-blue-V-state big-blue-disjoint-state valid-state-def*)

2.5 The central vertex

definition *central-vertex* :: $['a \ \text{set}, 'a] \Rightarrow \text{bool}$ **where**

$\text{central-vertex} \equiv \lambda X \ x. \ x \in X \wedge \text{card } (\text{Neighbours Blue } x \cap X) \leq \mu * \text{real } (\text{card } X)$

lemma *ex-central-vertex*:

assumes $\neg \text{termination-condition } X \ Y \ \neg \text{many-bluish } X$

shows $\exists x. \text{central-vertex } X \ x$

proof –

have $l \neq 0$

using *linorder-not-less assms unfolding many-bluish-def* **by** *force*

then have $*$: $\text{real } l \ \text{powr } (2/3) \leq \text{real } l \ \text{powr } (3/4)$

using *powr-mono* **by** *force*

then have $\text{card } \{x \in X. \text{bluish } X \ x\} < \text{card } X$

using *assms RN-mono*

unfolding *termination-condition-def many-bluish-def not-le*

by (*smt (verit, ccfv-SIG) linorder-not-le nat-ceiling-le-eq of-nat-le-iff*)

then obtain x **where** $x \in X \ \neg \text{bluish } X \ x$

by (*metis (mono-tags, lifting) mem-Collect-eq nat-neq-iff subsetI subset-antisym*)

then show *?thesis*

by (*meson bluish-def central-vertex-def linorder-linear*)

qed

lemma *finite-central-vertex-set*: $\text{finite } X \implies \text{finite } \{x. \text{central-vertex } X \ x\}$

by (simp add: central-vertex-def)

definition *max-central-vx* :: [*'a set, 'a set*] \Rightarrow *real* **where**
max-central-vx $\equiv \lambda X Y. \text{Max} (\text{weight } X Y \text{ ' } \{x. \text{central-vertex } X x\})$

lemma *central-vx-is-best*:

$\llbracket \text{central-vertex } X x; \text{finite } X \rrbracket \Longrightarrow \text{weight } X Y x \leq \text{max-central-vx } X Y$
unfolding *max-central-vx-def* **by** (simp add: finite-central-vertex-set)

lemma *ex-best-central-vx*:

$\llbracket \neg \text{termination-condition } X Y; \neg \text{many-bluish } X; \text{finite } X \rrbracket$
 $\Longrightarrow \exists x. \text{central-vertex } X x \wedge \text{weight } X Y x = \text{max-central-vx } X Y$
unfolding *max-central-vx-def*
by (*metis empty-iff ex-central-vertex finite-central-vertex-set mem-Collect-eq obtains-MAX*)

it's necessary to make a specific choice; a relational treatment might allow different vertices to be chosen, making a nonsense of the choice between steps 4 and 5

definition *choose-central-vx* $\equiv \lambda(X, Y, A, B). \text{@}x. \text{central-vertex } X x \wedge \text{weight } X Y x = \text{max-central-vx } X Y$

lemma *choose-central-vx-works*:

$\llbracket \neg \text{termination-condition } X Y; \neg \text{many-bluish } X; \text{finite } X \rrbracket$
 $\Longrightarrow \text{central-vertex } X (\text{choose-central-vx } (X, Y, A, B)) \wedge \text{weight } X Y (\text{choose-central-vx } (X, Y, A, B)) = \text{max-central-vx } X Y$
unfolding *choose-central-vx-def*
using *someI-ex [OF ex-best-central-vx]* **by** *force*

lemma *choose-central-vx-X*:

$\llbracket \neg \text{many-bluish } X; \neg \text{termination-condition } X Y; \text{finite } X \rrbracket \Longrightarrow \text{choose-central-vx } (X, Y, A, B) \in X$
using *central-vertex-def choose-central-vx-works* **by** *fastforce*

2.6 Red step

definition *reddish* $\equiv \lambda k X Y p x. \text{red-density } (Neighbours \text{ Red } x \cap X) (Neighbours \text{ Red } x \cap Y) \geq p - \text{alpha } (\text{hgt } p)$

inductive *red-step*

where $\llbracket \text{reddish } k X Y (\text{red-density } X Y) x; x = \text{choose-central-vx } (X, Y, A, B) \rrbracket$
 $\Longrightarrow \text{red-step } (X, Y, A, B) (Neighbours \text{ Red } x \cap X, Neighbours \text{ Red } x \cap Y, \text{insert } x A, B)$

lemma *red-step-V-state*:

assumes *red-step* $(X, Y, A, B) U' \neg \text{termination-condition } X Y$
 $\neg \text{many-bluish } X V\text{-state } (X, Y, A, B)$
shows *V-state* U'
proof –
have $X \subseteq V$

```

    using assms by (auto simp: V-state-def)
  then have choose-central-vx (X, Y, A, B) ∈ V
    using assms choose-central-vx-X by (fastforce simp: finX)
  with assms show ?thesis
    by (auto simp: V-state-def elim!: red-step.cases)
qed

lemma red-step-disjoint-state:
  assumes red-step (X, Y, A, B) U' ¬ termination-condition X Y
    ¬ many-bluish X V-state (X, Y, A, B) disjoint-state (X, Y, A, B)
  shows disjoint-state U'
proof -
  have choose-central-vx (X, Y, A, B) ∈ X
    using assms by (metis choose-central-vx-X finX)
  with assms show ?thesis
    by (auto simp: disjoint-state-def disjnt-iff not-own-Neighbour elim!: red-step.cases)
qed

lemma red-step-RB-state:
  assumes red-step (X, Y, A, B) U' ¬ termination-condition X Y
    ¬ many-bluish X V-state (X, Y, A, B) RB-state (X, Y, A, B)
  shows RB-state U'
proof -
  define x where x ≡ choose-central-vx (X, Y, A, B)
  have [simp]: finite X
    using assms by (simp add: finX)
  have x ∈ X
    using assms choose-central-vx-X by (metis ⟨finite X⟩ x-def)
  have A: all-edges-betw-un (insert x A) (insert x A) ⊆ Red
    if all-edges-betw-un A A ⊆ Red all-edges-betw-un A (X ∪ Y) ⊆ Red
    using that ⟨x ∈ X⟩ all-edges-betw-un-commute
  by (auto simp: all-edges-betw-un-insert2 all-edges-betw-un-Un2 intro!: all-uedges-betw-I)
  have B1: all-edges-betw-un (insert x A) (Neighbours Red x ∩ X) ⊆ Red
    if all-edges-betw-un A X ⊆ Red
    using that ⟨x ∈ X⟩ by (force simp: all-edges-betw-un-def in-Neighbours-iff)
  have B2: all-edges-betw-un (insert x A) (Neighbours Red x ∩ Y) ⊆ Red
    if all-edges-betw-un A Y ⊆ Red
    using that ⟨x ∈ X⟩ by (force simp: all-edges-betw-un-def in-Neighbours-iff)
  from assms A B1 B2 show ?thesis
    apply (clarsimp simp: RB-state-def simp flip: x-def elim!: red-step.cases)
    by (metis Int-Un-eq(2) Un-subset-iff all-edges-betw-un-Un2)
qed

lemma red-step-valid-state:
  assumes red-step (X, Y, A, B) U' ¬ termination-condition X Y
    ¬ many-bluish X valid-state (X, Y, A, B)
  shows valid-state U'
  by (meson assms red-step-RB-state red-step-V-state red-step-disjoint-state valid-state-def)

```

2.7 Density-boost step

inductive *density-boost*

where $\llbracket \neg \text{reddish } k \ X \ Y \ (\text{red-density } X \ Y) \ x; \ x = \text{choose-central-vx } (X, Y, A, B) \rrbracket$

$\implies \text{density-boost } (X, Y, A, B) \ (\text{Neighbours Blue } x \cap X, \ \text{Neighbours Red } x \cap Y, \ A, \ \text{insert } x \ B)$

lemma *density-boost-V-state:*

assumes *density-boost* $(X, Y, A, B) \ U' \neg \text{termination-condition } X \ Y$

$\neg \text{many-bluish } X \ V\text{-state } (X, Y, A, B)$

shows *V-state* U'

proof –

have $X \subseteq V$

using *assms* **by** $(\text{auto simp: } V\text{-state-def})$

then have *choose-central-vx* $(X, Y, A, B) \in V$

using *assms* *choose-central-vx-X finX* **by** *fastforce*

with *assms* **show** *?thesis*

by $(\text{auto simp: } V\text{-state-def elim!: } \text{density-boost.cases})$

qed

lemma *density-boost-disjoint-state:*

assumes *density-boost* $(X, Y, A, B) \ U' \neg \text{termination-condition } X \ Y$

$\neg \text{many-bluish } X \ V\text{-state } (X, Y, A, B) \ \text{disjoint-state } (X, Y, A, B)$

shows *disjoint-state* U'

proof –

have $X \subseteq V$

using *assms* **by** $(\text{auto simp: } V\text{-state-def})$

then have *choose-central-vx* $(X, Y, A, B) \in X$

using *assms* **by** $(\text{metis } \text{choose-central-vx-X finX})$

with *assms* **show** *?thesis*

by $(\text{auto simp: } \text{disjoint-state-def disjnt-iff not-own-Neighbour elim!: } \text{density-boost.cases})$

qed

lemma *density-boost-RB-state:*

assumes *density-boost* $(X, Y, A, B) \ U' \neg \text{termination-condition } X \ Y \neg \text{many-bluish } X \ V\text{-state } (X, Y, A, B)$

and *rb: RB-state* (X, Y, A, B)

shows *RB-state* U'

proof –

define x **where** $x \equiv \text{choose-central-vx } (X, Y, A, B)$

have $x \in X$

using *assms* **by** $(\text{metis } \text{choose-central-vx-X finX } x\text{-def})$

have *all-edges-betw-un* $A \ (\text{Neighbours Blue } x \cap X \cup \text{Neighbours Red } x \cap Y) \subseteq$

Red

if *all-edges-betw-un* $A \ (X \cup Y) \subseteq \text{Red}$

using *that* **by** $(\text{metis } \text{Int-Un-eq}(4) \ \text{Un-subset-iff all-edges-betw-un-Un2})$

moreover

have *all-edges-betw-un* $(\text{insert } x \ B) \ (\text{insert } x \ B) \subseteq \text{Blue}$

if *all-edges-betw-un* $B \ (B \cup X) \subseteq \text{Blue}$

using *that* $\langle x \in X \rangle$ **by** (*fastforce simp add: all-edges-betw-un-def*)
moreover
have *all-edges-betw-un* (*insert x B*) (*Neighbours Blue* $x \cap X \subseteq \text{Blue}$)
if *all-edges-betw-un B* ($B \cup X \subseteq \text{Blue}$)
using $\langle x \in X \rangle$ **that** **by** (*auto simp: all-edges-betw-un-def subset-iff in-Neighbours-iff*)
ultimately show *?thesis*
using *assms*
by (*auto simp: RB-state-def all-edges-betw-un-Un2 x-def [symmetric] elim!: density-boost.cases*)
qed

lemma *density-boost-valid-state:*
assumes *density-boost* (X, Y, A, B) $U' \neg \text{termination-condition } X Y \neg \text{many-bluish}$
 $X \text{ valid-state } (X, Y, A, B)$
shows *valid-state* U'
by (*meson assms density-boost-RB-state density-boost-V-state density-boost-disjoint-state valid-state-def*)

2.8 Execution steps 2–5 as a function

definition *next-state* :: $'a \text{ config} \Rightarrow 'a \text{ config}$ **where**
 $\text{next-state} \equiv \lambda(X, Y, A, B).$
if many-bluish X
then let (S, T) = *choose-blue-book* (X, Y, A, B) *in* ($T, Y, A, B \cup S$)
else let $x = \text{choose-central-vx}$ (X, Y, A, B) *in*
if reddish k X Y (red-density X Y) x
then (*Neighbours Red* $x \cap X, \text{Neighbours Red } x \cap Y, \text{insert } x A, B$)
else (*Neighbours Blue* $x \cap X, \text{Neighbours Red } x \cap Y, A, \text{insert } x B$)

lemma *next-state-valid:*
assumes *valid-state* (X, Y, A, B) $\neg \text{termination-condition } X Y$
shows *valid-state* ($\text{next-state } (X, Y, A, B)$)
proof (*cases many-bluish X*)
case *True*
with *assms finX* **have** $\bigwedge S T. \llbracket \text{choose-blue-book } (X, Y, A, B) = (S, T) \rrbracket$
 $\implies \text{big-blue } (X, Y, A, B) (T, Y, A, B \cup S)$
by (*metis big-blue.intros choose-blue-book-works valid-state-def*)
with *True* **have** $\text{big-blue } (X, Y, A, B) (\text{next-state } (X, Y, A, B))$
by (*auto simp: next-state-def split: prod.split*)
then show *?thesis*
using *assms big-blue-valid-state* **by** *blast*
next
case *non-bluish: False*
define x **where** $x = \text{choose-central-vx } (X, Y, A, B)$
show *?thesis*
proof (*cases reddish k X Y (red-density X Y) x*)
case *True*
with *non-bluish* **have** $\text{red-step } (X, Y, A, B) (\text{next-state } (X, Y, A, B))$
by (*simp add: next-state-def Let-def x-def red-step.intros split: prod.split*)

```

then show ?thesis
  using assms non-bluish red-step-valid-state by blast
next
  case False
  with non-bluish have density-boost  $(X, Y, A, B)$  (next-state  $(X, Y, A, B)$ )
    by (simp add: next-state-def Let-def x-def density-boost.intros split: prod.split)
  then show ?thesis
    using assms density-boost-valid-state non-bluish by blast
qed
qed

primrec stepper ::  $\text{nat} \Rightarrow 'a \text{ config}$  where
  stepper 0 =  $(X0, Y0, \{\}, \{\})$ 
| stepper (Suc n) =
  (let  $(X, Y, A, B)$  = stepper n in
   if termination-condition  $X Y$  then  $(X, Y, A, B)$ 
   else if even  $n$  then degree-reg  $(X, Y, A, B)$  else next-state  $(X, Y, A, B)$ )

lemma degree-reg-subset:
  assumes degree-reg  $(X, Y, A, B) = (X', Y', A', B')$ 
  shows  $X' \subseteq X \wedge Y' \subseteq Y$ 
  using assms by (auto simp: degree-reg-def X-degree-reg-def)

lemma next-state-subset:
  assumes next-state  $(X, Y, A, B) = (X', Y', A', B')$  finite  $X$ 
  shows  $X' \subseteq X \wedge Y' \subseteq Y$ 
  using assms choose-blue-book-subset
  apply (clarsimp simp: next-state-def valid-state-def Let-def split: if-split-asm
prod.splits)
  by (smt (verit) choose-blue-book-subset subset-eq)

lemma valid-state0: valid-state  $(X0, Y0, \{\}, \{\})$ 
  using XY0 by (simp add: valid-state-def V-state-def disjoint-state-def RB-state-def)

lemma valid-state-stepper [simp]: valid-state (stepper n)
proof (induction n)
  case  $0$ 
  then show ?case
    by (simp add: stepper-def valid-state0)
next
  case (Suc n)
  then show ?case
    by (force simp: next-state-valid degree-reg-valid-state split: prod.split)
qed

lemma V-state-stepper: V-state (stepper n)
  using valid-state-def valid-state-stepper by force

lemma RB-state-stepper: RB-state (stepper n)

```

using *valid-state-def valid-state-stepper* **by** *force*

lemma

assumes *stepper* $n = (X, Y, A, B)$

shows *stepper-A*: *clique* A *Red* $\wedge A \subseteq V$ **and** *stepper-B*: *clique* B *Blue* $\wedge B \subseteq V$

proof –

have $A \subseteq V$ $B \subseteq V$

using *V-state-stepper*[*of n*] *assms* **by** (*auto simp: V-state-def*)

moreover

have *all-edges-betw-un* A $A \subseteq \text{Red}$ *all-edges-betw-un* B $B \subseteq \text{Blue}$

using *RB-state-stepper*[*of n*] *assms* **by** (*auto simp: RB-state-def all-edges-betw-un-Un2*)

ultimately show *clique* A *Red* $\wedge A \subseteq V$ *clique* B *Blue* $\wedge B \subseteq V$

using *all-edges-betw-un-iff-clique* **by** *auto*

qed

lemma *card-B-limit*:

assumes *stepper* $n = (X, Y, A, B)$ **shows** *card* $B < l$

by (*metis B-less-l assms valid-state-stepper*)

definition *Xseq* $\equiv (\lambda(X, Y, A, B). X) \circ \text{stepper}$

definition *Yseq* $\equiv (\lambda(X, Y, A, B). Y) \circ \text{stepper}$

definition *Aseq* $\equiv (\lambda(X, Y, A, B). A) \circ \text{stepper}$

definition *Bseq* $\equiv (\lambda(X, Y, A, B). B) \circ \text{stepper}$

definition *pseq* $\equiv \lambda i. \text{red-density } (Xseq\ i) (Yseq\ i)$

lemma *Xseq-0* [*simp*]: *Xseq* $0 = X0$

by (*simp add: Xseq-def*)

lemma *Xseq-Suc-subset*: *Xseq* (*Suc* i) $\subseteq Xseq\ i$ **and** *Yseq-Suc-subset*: *Yseq* (*Suc* i) $\subseteq Yseq\ i$

apply (*simp-all add: Xseq-def Yseq-def split: if-split-asm prod.split*)

by (*metis V-state-stepper degree-reg-subset finX next-state-subset*)+

lemma *Xseq-antimono*: $j \leq i \implies Xseq\ i \subseteq Xseq\ j$

by (*simp add: Xseq-Suc-subset lift-Suc-antimono-le*)

lemma *Xseq-subset-V*: *Xseq* $i \subseteq V$

using *XY0 Xseq-0 Xseq-antimono* **by** *blast*

lemma *finite-Xseq*: *finite* (*Xseq* i)

by (*meson Xseq-subset-V finV finite-subset*)

lemma *Yseq-0* [*simp*]: *Yseq* $0 = Y0$

by (*simp add: Yseq-def*)

lemma *Yseq-antimono*: $j \leq i \implies Yseq\ i \subseteq Yseq\ j$

by (*simp add: Yseq-Suc-subset lift-Suc-antimono-le*)

lemma *Yseq-subset-V*: *Yseq* $i \subseteq V$

using $XY0$ $Yseq-0$ $Yseq-antimono$ **by** *blast*

lemma $finite-Yseq$: $finite (Yseq i)$
by (*meson* $Yseq-subset-V$ $finV$ $finite-subset$)

lemma $Xseq-Yseq-disjnt$: $disjnt (Xseq i) (Yseq i)$
using $XY0(1)$
unfolding $disjnt-iff$
by (*metis* $Xseq-0$ $Xseq-antimono$ $Yseq-0$ $Yseq-antimono$ $in-mono$ $zero-le$)

lemma $edge-card-eq-pee$:
 $edge-card Red (Xseq i) (Yseq i) = pseq i * card (Xseq i) * card (Yseq i)$
by (*simp* add : $pseq-def$ $gen-density-def$ $finite-Xseq$ $finite-Yseq$)

lemma $valid-state-seq$: $valid-state(Xseq i, Yseq i, Aseq i, Bseq i)$
using $valid-state-stepper[of i]$
by (*force simp*: $Xseq-def$ $Yseq-def$ $Aseq-def$ $Bseq-def$ *simp* del : $valid-state-stepper$ $split$: $prod.split$)

lemma $Aseq-less-k$: $card (Aseq i) < k$
by (*meson* $A-less-k$ $valid-state-seq$)

lemma $Aseq-0$ [*simp*]: $Aseq 0 = \{\}$
by (*simp* add : $Aseq-def$)

lemma $Aseq-Suc-subset$: $Aseq i \subseteq Aseq (Suc i)$ **and** $Bseq-Suc-subset$: $Bseq i \subseteq Bseq (Suc i)$
by (*auto simp*: $Aseq-def$ $Bseq-def$ $next-state-def$ $degree-reg-def$ $Let-def$ $split$: $prod.split$)

lemma
assumes $j \leq i$
shows $Aseq-mono$: $Aseq j \subseteq Aseq i$ **and** $Bseq-mono$: $Bseq j \subseteq Bseq i$
using *assms* **by** (*auto simp*: $Aseq-Suc-subset$ $Bseq-Suc-subset$ $lift-Suc-mono-le$)

lemma $Aseq-subset-V$: $Aseq i \subseteq V$
using $stepper-A[of i]$ **by** (*simp* add : $Aseq-def$ $split$: $prod.split$)

lemma $Bseq-subset-V$: $Bseq i \subseteq V$
using $stepper-B[of i]$ **by** (*simp* add : $Bseq-def$ $split$: $prod.split$)

lemma $finite-Aseq$: $finite (Aseq i)$ **and** $finite-Bseq$: $finite (Bseq i)$
by (*meson* $Aseq-subset-V$ $Bseq-subset-V$ $finV$ $finite-subset$) $+$

lemma $Bseq-less-l$: $card (Bseq i) < l$
by (*meson* $B-less-l$ $valid-state-seq$)

lemma $Bseq-0$ [*simp*]: $Bseq 0 = \{\}$
by (*simp* add : $Bseq-def$)

lemma *pee-eq-p0*: $pseq\ 0 = p0$
by (*simp add: pseq-def p0-def*)

lemma *pee-ge0*: $pseq\ i \geq 0$
by (*simp add: gen-density-ge0 pseq-def*)

lemma *pee-le1*: $pseq\ i \leq 1$
using *gen-density-le1 pseq-def* **by** *presburger*

lemma *pseq-0*: $p0 = pseq\ 0$
by (*simp add: p0-def pseq-def Xseq-def Yseq-def*)

The central vertex at each step (though only defined in some cases), $x-i$ in the paper

definition *cvx* $\equiv \lambda i. choose-central-vx\ (stepper\ i)$

the indexing of *beta* is as in the paper — and different from that of *Xseq*

definition

beta $\equiv \lambda i. let\ (X,Y,A,B) = stepper\ i\ in\ card(Neighbours\ Blue\ (cvx\ i) \cap X) / card\ X$

lemma *beta-eq*: $beta\ i = card(Neighbours\ Blue\ (cvx\ i) \cap Xseq\ i) / card\ (Xseq\ i)$
by (*simp add: beta-def cvx-def Xseq-def split: prod.split*)

lemma *beta-ge0*: $beta\ i \geq 0$
by (*simp add: beta-eq*)

2.9 The classes of execution steps

For R, B, S, D

datatype *stepkind* = *red-step* | *bblue-step* | *dboost-step* | *dreg-step* | *halted*

definition *next-state-kind* :: 'a *config* \Rightarrow *stepkind* **where**

next-state-kind $\equiv \lambda(X,Y,A,B).$

if *many-bluish* *X* then *bblue-step*

else let $x = choose-central-vx\ (X,Y,A,B)$ in

if *reddish* $k\ X\ Y$ (*red-density* *X* *Y*) x then *red-step*

else *dboost-step*

definition *stepper-kind* :: *nat* \Rightarrow *stepkind* **where**

stepper-kind $i =$

(let $(X,Y,A,B) = stepper\ i$ in

if *termination-condition* *X* *Y* then *halted*

else if *even* i then *dreg-step* else *next-state-kind* (X,Y,A,B))

definition *Step-class* $\equiv \lambda knd. \{n. stepper-kind\ n \in knd\}$

lemma *subset-Step-class*: $\llbracket i \in Step-class\ K'; K' \subseteq K \rrbracket \Longrightarrow i \in Step-class\ K$
by (*auto simp: Step-class-def*)

lemma *Step-class-Un*: $\text{Step-class } (K' \cup K) = \text{Step-class } K' \cup \text{Step-class } K$
by (*auto simp: Step-class-def*)

lemma *Step-class-insert*: $\text{Step-class } (\text{insert } \text{knd } K) = (\text{Step-class } \{\text{knd}\}) \cup (\text{Step-class } K)$
by (*auto simp: Step-class-def*)

lemma *Step-class-insert-NO-MATCH*:
NO-MATCH $\{ \} K \implies \text{Step-class } (\text{insert } \text{knd } K) = (\text{Step-class } \{\text{knd}\}) \cup (\text{Step-class } K)$
by (*auto simp: Step-class-def*)

lemma *Step-class-UNIV*: $\text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}, \text{dreg-step}, \text{halted}\} = \text{UNIV}$
using *Step-class-def stepkind.exhaust* **by** *auto*

lemma *Step-class-cases*:
 $i \in \text{Step-class } \{\text{stepkind.red-step}\} \vee i \in \text{Step-class } \{\text{bblue-step}\} \vee$
 $i \in \text{Step-class } \{\text{dboost-step}\} \vee i \in \text{Step-class } \{\text{dreg-step}\} \vee$
 $i \in \text{Step-class } \{\text{halted}\}$
using *Step-class-def stepkind.exhaust* **by** *auto*

lemmas *step-kind-defs* = *Step-class-def stepper-kind-def next-state-kind-def*
Xseq-def Yseq-def Aseq-def Bseq-def cvx-def Let-def

lemma *disjnt-Step-class*:
 $\text{disjnt } \text{knd } \text{knd}' \implies \text{disjnt } (\text{Step-class } \text{knd}) (\text{Step-class } \text{knd}')$
by (*auto simp: Step-class-def disjnt-iff*)

lemma *halted-imp-next-halted*: $\text{stepper-kind } i = \text{halted} \implies \text{stepper-kind } (\text{Suc } i) = \text{halted}$
by (*auto simp: step-kind-defs split: prod.split if-split-asm*)

lemma *halted-imp-ge-halted*: $\text{stepper-kind } i = \text{halted} \implies \text{stepper-kind } (i+n) = \text{halted}$
by (*induction n*) (*auto simp: halted-imp-next-halted*)

lemma *Step-class-halted-forever*: $\llbracket i \in \text{Step-class } \{\text{halted}\}; i \leq j \rrbracket \implies j \in \text{Step-class } \{\text{halted}\}$
by (*simp add: Step-class-def*) (*metis halted-imp-ge-halted le-iff-add*)

lemma *Step-class-not-halted*: $\llbracket i \notin \text{Step-class } \{\text{halted}\}; i \geq j \rrbracket \implies j \notin \text{Step-class } \{\text{halted}\}$
using *Step-class-halted-forever* **by** *blast*

lemma
assumes $i \notin \text{Step-class } \{\text{halted}\}$
shows *not-halted-pee-gt*: $\text{pseq } i > 1/k$

and $Xseq\text{-}gt0$: $card (Xseq\ i) > 0$
and $Xseq\text{-}gt\text{-}RN$: $card (Xseq\ i) > RN\ k\ (nat\ \lceil real\ l\ powr\ (3/4)\rceil)$
and $not\text{-}termination\text{-}condition$: $\neg\ termination\text{-}condition\ (Xseq\ i)\ (Yseq\ i)$
using $assms$
by ($auto\ simp$: $step\text{-}kind\text{-}defs\ termination\text{-}condition\text{-}def\ pseq\text{-}def\ split$: $if\text{-}split\text{-}asm\ prod.\ split\text{-}asm$)

lemma $not\text{-}halted\text{-}pee\text{-}gt0$:
assumes $i \notin Step\text{-}class\ \{halted\}$
shows $pseq\ i > 0$
using $not\text{-}halted\text{-}pee\text{-}gt$ [$OF\ assms$] $linorder\text{-}not\text{-}le\ order\text{-}less\text{-}le\text{-}trans$ **by** $fastforce$

lemma $Yseq\text{-}gt0$:
assumes $i \notin Step\text{-}class\ \{halted\}$
shows $card (Yseq\ i) > 0$
using $not\text{-}halted\text{-}pee\text{-}gt$ [$OF\ assms$]
using $card\text{-}gt\text{-}0\text{-}iff\ finite\text{-}Yseq\ pseq\text{-}def$ **by** $fastforce$

lemma $step\text{-}odd$: $i \in Step\text{-}class\ \{red\text{-}step, bblue\text{-}step, dboost\text{-}step\} \implies odd\ i$
by ($auto\ simp$: $Step\text{-}class\text{-}def\ stepper\text{-}kind\text{-}def\ split$: $if\text{-}split\text{-}asm\ prod.\ split\text{-}asm$)

lemma $step\text{-}even$: $i \in Step\text{-}class\ \{dreg\text{-}step\} \implies even\ i$
by ($auto\ simp$: $Step\text{-}class\text{-}def\ stepper\text{-}kind\text{-}def\ next\text{-}state\text{-}kind\text{-}def\ split$: $if\text{-}split\text{-}asm\ prod.\ split\text{-}asm$)

lemma $not\text{-}halted\text{-}odd\text{-}RBS$: $\llbracket i \notin Step\text{-}class\ \{halted\}; odd\ i \rrbracket \implies i \in Step\text{-}class\ \{red\text{-}step, bblue\text{-}step, dboost\text{-}step\}$
by ($auto\ simp$: $Step\text{-}class\text{-}def\ stepper\text{-}kind\text{-}def\ next\text{-}state\text{-}kind\text{-}def\ split$: $prod.\ split\text{-}asm$)

lemma $not\text{-}halted\text{-}even\text{-}dreg$: $\llbracket i \notin Step\text{-}class\ \{halted\}; even\ i \rrbracket \implies i \in Step\text{-}class\ \{dreg\text{-}step\}$
by ($auto\ simp$: $Step\text{-}class\text{-}def\ stepper\text{-}kind\text{-}def\ next\text{-}state\text{-}kind\text{-}def\ split$: $prod.\ split\text{-}asm$)

lemma $step\text{-}before\text{-}dreg$:
assumes $Suc\ i \in Step\text{-}class\ \{dreg\text{-}step\}$
shows $i \in Step\text{-}class\ \{red\text{-}step, bblue\text{-}step, dboost\text{-}step\}$
using $assms$ **by** ($auto\ simp$: $step\text{-}kind\text{-}defs\ split$: $if\text{-}split\text{-}asm\ prod.\ split\text{-}asm$)

lemma $dreg\text{-}before\text{-}step$:
assumes $Suc\ i \in Step\text{-}class\ \{red\text{-}step, bblue\text{-}step, dboost\text{-}step\}$
shows $i \in Step\text{-}class\ \{dreg\text{-}step\}$
using $assms$ **by** ($auto\ simp$: $Step\text{-}class\text{-}def\ stepper\text{-}kind\text{-}def\ split$: $if\text{-}split\text{-}asm\ prod.\ split\text{-}asm$)

lemma
assumes $i \in Step\text{-}class\ \{red\text{-}step, bblue\text{-}step, dboost\text{-}step\}$
shows $dreg\text{-}before\text{-}step'$: $i - Suc\ 0 \in Step\text{-}class\ \{dreg\text{-}step\}$
and $dreg\text{-}before\text{-}gt0$: $i > 0$
proof –

show $i > 0$
using *assms gr0I step-odd* **by** *force*
then show $i - \text{Suc } 0 \in \text{Step-class } \{\text{dreg-step}\}$
using *assms dreg-before-step Suc-pred* **by** *force*
qed

lemma *dreg-before-step1*:
assumes $i \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}\}$
shows $i - 1 \in \text{Step-class } \{\text{dreg-step}\}$
using *dreg-before-step' [OF assms]* **by** *auto*

lemma *step-odd-minus2*:
assumes $i \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}\}$ $i > 1$
shows $i - 2 \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}\}$
by (*metis Suc-1 Suc-diff-Suc assms dreg-before-step1 step-before-dreg*)

lemma *Step-class-iterates*:
assumes *finite* (*Step-class* $\{knd\}$)
obtains n **where** *Step-class* $\{knd\} = \{m. m < n \wedge \text{stepper-kind } m = knd\}$
proof –
have *eq*: (*Step-class* $\{knd\}$) = $(\bigcup i. \{m. m < i \wedge \text{stepper-kind } m = knd\})$
by (*auto simp: Step-class-def*)
then obtain n **where** n : (*Step-class* $\{knd\}$) = $(\bigcup i < n. \{m. m < i \wedge \text{stepper-kind } m = knd\})$
using *finite-countable-equals [OF assms]* **by** *blast*
with *Step-class-def*
have $\{m. m < n \wedge \text{stepper-kind } m = knd\} = (\bigcup i < n. \{m. m < i \wedge \text{stepper-kind } m = knd\})$
by *auto*
then show *?thesis*
by (*metis n that*)
qed

lemma *step-non-terminating-iff*:
 $i \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}, \text{dreg-step}\}$
 $\longleftrightarrow \neg \text{termination-condition } (X \text{seq } i) (Y \text{seq } i)$
by (*auto simp: step-kind-defs split: if-split-asm prod.split-asm*)

lemma *step-terminating-iff*:
 $i \in \text{Step-class } \{\text{halted}\} \longleftrightarrow \text{termination-condition } (X \text{seq } i) (Y \text{seq } i)$
by (*auto simp: step-kind-defs split: if-split-asm prod.split-asm*)

lemma *not-many-bluish*:
assumes $i \in \text{Step-class } \{\text{red-step}, \text{dboost-step}\}$
shows $\neg \text{many-bluish } (X \text{seq } i)$
using *assms*
by (*simp add: step-kind-defs split: if-split-asm prod.split-asm*)

lemma *stepper-XYseq*: *stepper* $i = (X, Y, A, B) \implies X = X \text{seq } i \wedge Y = Y \text{seq } i$

using *Xseq-def Yseq-def* by *fastforce*

lemma *cvx-works*:

assumes $i \in \text{Step-class } \{\text{red-step, dboost-step}\}$

shows *central-vertex* (*Xseq i*) (*cvx i*)

$\wedge \text{weight } (Xseq\ i) (Yseq\ i) (cvx\ i) = \text{max-central-vx } (Xseq\ i) (Yseq\ i)$

proof –

have $\neg \text{termination-condition } (Xseq\ i) (Yseq\ i)$

using *Step-class-def assms step-non-terminating-iff* by *fastforce*

then show *?thesis*

using *assms not-many-bluish[OF assms]*

apply (*simp add: Step-class-def Xseq-def cvx-def Yseq-def split: prod.split prod.split-asm*)

by (*metis V-state-stepper choose-central-vx-works finX*)

qed

lemma *cvx-in-Xseq*:

assumes $i \in \text{Step-class } \{\text{red-step, dboost-step}\}$

shows $cvx\ i \in Xseq\ i$

using *assms cvx-works[OF assms]*

by (*simp add: Xseq-def central-vertex-def cvx-def split: prod.split-asm*)

lemma *card-Xseq-pos*:

assumes $i \in \text{Step-class } \{\text{red-step, dboost-step}\}$

shows $\text{card } (Xseq\ i) > 0$

by (*metis assms card-0-eq cvx-in-Xseq empty-iff finite-Xseq gr0I*)

lemma *beta-le*:

assumes $i \in \text{Step-class } \{\text{red-step, dboost-step}\}$

shows $\beta\ i \leq \mu$

using *assms cvx-works[OF assms] $\mu 0 1$*

by (*simp add: beta-def central-vertex-def Xseq-def divide-simps split: prod.split-asm*)

2.10 Termination proof

Each step decreases the size of X

lemma *ex-nonempty-blue-book*:

assumes *mb: many-bluish X*

shows $\exists x \in X. \text{good-blue-book } X (\{x\}, \text{Neighbours Blue } x \cap X)$

proof –

have $RN\ k\ (\text{nat } \lceil \text{real } l\ \text{powr } (2 / 3) \rceil) > 0$

by (*metis kn0 ln0 RN-eq-0-iff gr0I of-nat-ceiling of-nat-eq-0-iff powr-nonneg-iff*)

then have $\{x \in X. \text{bluish } X\ x\} \neq \{\}$

using *mb* by (*force simp: many-bluish-def*)

then obtain x **where** $x \in X$ **and** $x: \text{bluish } X\ x$

by *auto*

have *book* $\{x\}$ (*Neighbours Blue* $x \cap X$) *Blue*

by (*force simp: book-def all-edges-betw-un-def in-Neighbours-iff*)

with x **show** *?thesis*

by (auto simp: bluish-def good-blue-book-def $\langle x \in X \rangle$)
 qed

lemma choose-blue-book-psubset:
 assumes many-bluish X and ST : choose-blue-book $(X, Y, A, B) = (S, T)$
 and finite X
 shows $T \neq X$

proof –
 obtain x where $x \in X$ and x : good-blue-book X $(\{x\}, \text{Neighbours Blue } x \cap X)$
 using ex-nonempty-blue-book assms by blast
 with $\langle \text{finite } X \rangle$ have best-blue-book-card $X \neq 0$
 unfolding valid-state-def
 by (metis best-blue-book-is-best card.empty card-seteq empty-not-insert finite.intros
 singleton-insert-inj-eq)
 then have $S \neq \{\}$
 by (metis $\langle \text{finite } X \rangle$ ST choose-blue-book-works card.empty)
 with $\langle \text{finite } X \rangle$ ST show ?thesis
 by (metis Int-absorb2 choose-blue-book-subset disjnt-def)
 qed

lemma next-state-smaller:
 assumes next-state $(X, Y, A, B) = (X', Y', A', B')$
 and finite X and nont: \neg termination-condition X Y
 shows $X' \subset X$

proof –
 have $X' \subseteq X$
 using assms next-state-subset by auto
 moreover have $X' \neq X$

proof –
 have *: $\neg X \subseteq \text{Neighbours } rb \ x \cap X$ if $x \in X$ $rb \subseteq E$ for x rb
 using that by (auto simp: Neighbours-def subset-iff)
 show ?thesis
proof (cases many-bluish X)
 case True
 with assms show ?thesis
 by (auto simp: next-state-def split: if-split-asm prod.split-asm
 dest!: choose-blue-book-psubset [OF True])
 next
 case False
 then have choose-central-vx $(X, Y, A, B) \in X$
 by (simp add: $\langle \text{finite } X \rangle$ choose-central-vx- X nont)
 with assms $*[\text{of - Red}] *[\text{of - Blue}] \langle X' \subseteq X \rangle$ Red-E Blue-E False
 choose-central-vx- X [OF False nont]
 show ?thesis
 by (fastforce simp: next-state-def Let-def split: if-split-asm prod.split-asm)
 qed
 qed
 ultimately show ?thesis
 by auto

qed

lemma *do-next-state*:

assumes $odd\ i \neg\ termination\ condition\ (Xseq\ i)\ (Yseq\ i)$

obtains $A\ B\ A'\ B'$ **where** $next\ state\ (Xseq\ i,\ Yseq\ i,\ A,\ B)$
 $=\ (Xseq\ (Suc\ i),\ Yseq\ (Suc\ i),\ A',\ B')$

using *assms*

by (*force simp: Xseq-def Yseq-def split: if-split-asm prod.split-asm prod.split*)

lemma *step-bound*:

assumes $i:\ Suc\ (2*i) \in\ Step\ class\ \{red\ step,\ bblue\ step,\ dboost\ step\}$

shows $card\ (Xseq\ (Suc\ (2*i))) + i \leq card\ X0$

using *i*

proof (*induction i*)

case *0*

then show *?case*

by (*metis Xseq-0 Xseq-Suc-subset add-0-right mult-0-right card-mono finite-X0*)

next

case (*Suc i*)

then have $nt:\ \neg\ termination\ condition\ (Xseq\ (Suc\ (2*i)))\ (Yseq\ (Suc\ (2*i)))$

unfolding *step-non-terminating-iff [symmetric]*

by (*metis Step-class-insert Suc-1 Un-iff dreg-before-step mult-Suc-right plus-1-eq-Suc plus-nat.simps(2) step-before-dreg*)

obtain $A\ B\ A'\ B'$ **where** *2*:

$next\ state\ (Xseq\ (Suc\ (2*i)),\ Yseq\ (Suc\ (2*i)),\ A,\ B) = (Xseq\ (Suc\ (Suc\ (2*i))),\ Yseq\ (Suc\ (Suc\ (2*i))),\ A',\ B')$

by (*meson nt Suc-double-not-eq-double do-next-state evenE*)

have $Xseq\ (Suc\ (Suc\ (2*i))) \subset Xseq\ (Suc\ (2*i))$

by (*meson 2 finite-Xseq assms next-state-smaller nt*)

then have $card\ (Xseq\ (Suc\ (Suc\ (Suc\ (2*i)))) < card\ (Xseq\ (Suc\ (2*i)))$

by (*meson Xseq-Suc-subset le-less-trans finite-Xseq psubset-card-mono*)

moreover have $card\ (Xseq\ (Suc\ (2*i))) + i \leq card\ X0$

using *Suc dreg-before-step step-before-dreg* **by** *force*

ultimately show *?case* **by** *auto*

qed

lemma *Step-class-halted-nonempty*: $Step\ class\ \{halted\} \neq \{\}$

proof –

define $i \equiv Suc\ (2 * Suc\ (card\ X0))$

have *odd i*

by (*auto simp: i-def*)

then have $i \notin Step\ class\ \{dreg\ step\}$

using *step-even* **by** *blast*

moreover have $i \notin Step\ class\ \{red\ step,\ bblue\ step,\ dboost\ step\}$

unfolding *i-def* **using** *step-bound le-add2 not-less-eq-eq* **by** *blast*

ultimately show *?thesis*

using $\langle odd\ i \rangle\ not\ halted\ odd\ RBS$ **by** *blast*

qed

definition *halted-point* \equiv *Inf* (*Step-class* {halted})

lemma *halted-point-halted*: *halted-point* \in *Step-class* {halted}
using *Step-class-halted-nonempty* *Inf-nat-def1*
by (*auto simp: halted-point-def*)

lemma *halted-point-minimal*:
shows $i \notin \text{Step-class } \{\text{halted}\} \longleftrightarrow i < \text{halted-point}$
using *Step-class-halted-nonempty*
by (*metis wellorder-Inf-le1 Inf-nat-def1 Step-class-not-halted halted-point-def less-le-not-le nle-le*)

lemma *halted-point-minimal'*: *stepper-kind* $i \neq \text{halted} \longleftrightarrow i < \text{halted-point}$
by (*simp add: Step-class-def flip: halted-point-minimal*)

lemma *halted-eq-Compl*:
Step-class {*dreg-step, red-step, bblue-step, dboost-step*} = \neg *Step-class* {halted}
using *Step-class-UNIV* [of] **by** (*auto simp: Step-class-def*)

lemma *before-halted-eq*:
shows $\{.. < \text{halted-point}\} = \text{Step-class } \{\text{dreg-step, red-step, bblue-step, dboost-step}\}$
using *halted-point-minimal* **by** (*force simp: halted-eq-Compl*)

lemma *finite-components*:
shows *finite* (*Step-class* {*dreg-step, red-step, bblue-step, dboost-step*})
by (*metis before-halted-eq finite-lessThan*)

lemma
shows *dreg-step-finite* [*simp*]: *finite* (*Step-class* {*dreg-step*})
and *red-step-finite* [*simp*]: *finite* (*Step-class* {*red-step*})
and *bblue-step-finite* [*simp*]: *finite* (*Step-class* {*bblue-step*})
and *dboost-step-finite* [*simp*]: *finite* (*Step-class* {*dboost-step*})
using *finite-components* **by** (*auto simp: Step-class-insert-NO-MATCH*)

lemma *halted-stepper-add-eq*: *stepper* (*halted-point* + i) = *stepper* (*halted-point*)
proof (*induction i*)
case 0
then show ?*case*
by *auto*
next
case (*Suc i*)
have *hlt*: *stepper-kind* (*halted-point*) = *halted*
using *Step-class-def halted-point-halted* **by** *force*
obtain $X Y A B$ **where** *: *stepper* (*halted-point*) = (X, Y, A, B)
by (*metis surj-pair*)
with *hlt* **have** *termination-condition X Y*
by (*simp add: stepper-kind-def next-state-kind-def split: if-split-asm*)
with * **show** ?*case*
by (*simp add: Suc*)

qed

lemma *halted-stepper-eq*:

assumes $i: i \geq \text{halted-point}$

shows $\text{stepper } i = \text{stepper } (\text{halted-point})$

using $\mu 01$ **by** (*metis* *assms* *halted-stepper-add-eq* *le-iff-add*)

lemma *below-halted-point-cardX*:

assumes $i < \text{halted-point}$

shows $\text{card } (X\text{seq } i) > 0$

using *Xseq-gt0* *assms* *halted-point-minimal* *halted-stepper-eq* $\mu 01$

by *blast*

end

sublocale $\text{Book}' \subseteq \text{Book}$ **where** $\mu = \gamma$

proof

show $0 < \gamma \ \gamma < 1$

using *ln0* *kn0* **by** (*auto simp:* $\gamma\text{-def}$)

qed (*use* *XY0* *density-ge-p0-min* **in** *auto*)

lemma (**in** *Book*) *Book'*:

assumes $\gamma = \text{real } l / (\text{real } k + \text{real } l)$

shows $\text{Book}' \ V \ E \ p0\text{-min} \ \text{Red} \ \text{Blue} \ l \ k \ \gamma \ X0 \ Y0$

proof qed (*use* *assms* *XY0* *density-ge-p0-min* **in** *auto*)

end

3 Big Blue Steps: theorems

theory *Big-Blue-Steps* **imports** *Book*

begin

3.1 Preliminaries

A bounded increasing sequence of finite sets eventually terminates

lemma *Union-incseq-finite*:

assumes *fin*: $\bigwedge n. \text{finite } (A \ n)$ **and** *N*: $\bigwedge n. \text{card } (A \ n) < N$ **and** *incseq* *A*

shows $\forall_F k$ *in sequentially*. $\bigcup (\text{range } A) = A \ k$

proof (*rule* *ccontr*)

assume $\neg ?thesis$

then have $\forall k. \exists l \geq k. \bigcup (\text{range } A) \neq A \ l$

using *eventually-sequentially* **by** *force*

then have $\forall k. \exists l \geq k. \exists m \geq l. A \ m \neq A \ l$

by (*smt* (*verit*, *ccfv-threshold*) $\langle \text{incseq } A \rangle$ *cSup-eq-maximum image-iff monotoneD* *nle-le* *rangeI*)

then have $\forall k. \exists l \geq k. A \ l - A \ k \neq \{\}$

```

  by (metis <incseq A> diff-shunt-var monotoneD nat-le-linear subset-antisym)
then obtain f where  $f: \bigwedge k. f k \geq k \wedge A (f k) - A k \neq \{\}$ 
  by metis
have  $\text{card } (A ((f \wedge^i) 0)) \geq i$  for  $i$ 
proof (induction  $i$ )
  case 0
  then show ?case
  by auto
next
  case (Suc  $i$ )
have  $\text{card } (A ((f \wedge^i) 0)) < \text{card } (A (f ((f \wedge^i) 0)))$ 
  by (metis Diff-cancel <incseq A> card-seteq f fin leI monotoneD)
then show ?case
  using Suc by simp
qed
with N show False
  using linorder-not-less by auto
qed

```

Two lemmas for proving "bigness lemmas" over a closed interval

```

lemma eventually-all-geI0:
assumes  $\forall_F l$  in sequentially.  $P a l$ 
   $\bigwedge l x. \llbracket P a l; a \leq x; x \leq b; l \geq L \rrbracket \implies P x l$ 
shows  $\forall_F l$  in sequentially.  $\forall x. a \leq x \wedge x \leq b \longrightarrow P x l$ 
by (smt (verit, del-insts) assms eventually-sequentially eventually-elim2)

```

```

lemma eventually-all-geI1:
assumes  $\forall_F l$  in sequentially.  $P b l$ 
   $\bigwedge l x. \llbracket P b l; a \leq x; x \leq b; l \geq L \rrbracket \implies P x l$ 
shows  $\forall_F l$  in sequentially.  $\forall x. a \leq x \wedge x \leq b \longrightarrow P x l$ 
by (smt (verit, del-insts) assms eventually-sequentially eventually-elim2)

```

Mehta's binomial function: convex on the entire real line and coinciding with gchoose under weak conditions

```

definition mfact  $\equiv \lambda a k. \text{if } a < \text{real } k - 1 \text{ then } 0 \text{ else prod } (\lambda i. a - \text{of-nat } i) \{0..<k\}$ 

```

Mehta's special rule for convexity, my proof

```

lemma convex-on-extend:
fixes  $f :: \text{real} \Rightarrow \text{real}$ 
assumes  $cf: \text{convex-on } \{k..\} f$  and  $mon: \text{mono-on } \{k..\} f$ 
  and  $fk: \bigwedge x. x < k \implies f x = f k$ 
shows convex-on UNIV  $f$ 
proof (intro convex-on-linorderI)
  fix  $t x y :: \text{real}$ 
  assume  $t: 0 < t < 1$  and  $x < y$ 
  let  $?u = ((1 - t) *_R x + t *_R y)$ 
  show  $f ?u \leq (1 - t) * f x + t * f y$ 
  proof (cases  $k \leq x$ )

```

```

    case True
    with <x < y> t show ?thesis
    by (intro convex-onD [OF cf]) auto
next
case False
then have x < k and fvk: f x = f k by (auto simp: fk)
show ?thesis
proof (cases k ≤ y)
  case True
  then have f y ≥ f k
    using mon mono-onD by auto
  have kle: k ≤ (1 - t) * k + t * y
    using True segment-bound-lemma t by auto
  have fle: f ((1 - t) *R k + t *R y) ≤ (1 - t) * f k + t * f y
    using t True by (intro convex-onD [OF cf]) auto
  with False
  show ?thesis
  proof (cases ?u < k)
    case True
    then show ?thesis
      using <f k ≤ f y> fvk fk segment-bound-lemma t by auto
  next
  case False
  have f ?u ≤ f ((1 - t) *R k + t *R y)
    using kle <x < k> False t by (intro mono-onD [OF mon]) auto
  then show ?thesis
    using fle fvk by auto
qed
next
case False
with <x < k> show ?thesis
  by (simp add: fk convex-bound-lt order-less-imp-le segment-bound-lemma t)
qed
qed
qed auto

```

```

lemma convex-mfact:
  assumes k > 0
  shows convex-on UNIV (λa. mfact a k)
  unfolding mfact-def
proof (rule convex-on-extend)
  show convex-on {real (k - 1)..} (λa. if a < real k - 1 then 0 else ∏ i = 0..k.
a - real i)
  using convex-gchoose-aux [of k] assms
  apply (simp add: convex-on-def Ball-def)
  by (smt (verit, del-insts) distrib-right mult-cancel-right2 mult-left-mono)
  show mono-on {real (k - 1)..} (λa. if a < real k - 1 then 0 else ∏ i = 0..k.
a - real i)
  using <k > 0> by (auto simp: mono-on-def intro!: prod-mono)

```

qed (use *assms gr0-conv-Suc* in force)

definition *mbinomial* :: *real* \Rightarrow *nat* \Rightarrow *real*
 where *mbinomial* $\equiv \lambda a k. \text{mfact } a \ k / \text{fact } k$

lemma *convex-mbinomial*: $k > 0 \implies \text{convex-on UNIV } (\lambda x. \text{mbinomial } x \ k)$
 by (*simp add: mbinomial-def convex-mfact convex-on-cdiv*)

lemma *mbinomial-eq-choose* [*simp*]: *mbinomial* (*real n*) *k* = *n choose k*
 by (*simp add: binomial-gbinomial gbinomial-prod-rev mbinomial-def mfact-def*)

lemma *mbinomial-eq-gchoose* [*simp*]: $k \leq a \implies \text{mbinomial } a \ k = a \ \text{gchoose } k$
 by (*simp add: gbinomial-prod-rev mbinomial-def mfact-def*)

3.2 Preliminaries: Fact D1

from appendix D, page 55

lemma *Fact-D1-73-aux*:

fixes $\sigma :: \text{real}$ **and** $m \ b :: \text{nat}$

assumes $\sigma: 0 < \sigma$ **and** $bm: \text{real } b < \text{real } m$

shows $((\sigma * m) \ \text{gchoose } b) * \text{inverse } (m \ \text{gchoose } b) = \sigma ^ b * (\prod i < b. 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i)))$

proof –

have $((\sigma * m) \ \text{gchoose } b) * \text{inverse } (m \ \text{gchoose } b) = (\prod i < b. (\sigma * m - i) / (\text{real } m - \text{real } i))$

using bm **by** (*simp add: gbinomial-prod-rev prod-dividef atLeast0LessThan*)

also have $\dots = \sigma ^ b * (\prod i < b. 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i)))$

using $bm \ \sigma$ **by** (*induction b*) (*auto simp: field-simps*)

finally show *?thesis* .

qed

This is fact 4.2 (page 11) as well as equation (73), page 55.

lemma *Fact-D1-73*:

fixes $\sigma :: \text{real}$ **and** $m \ b :: \text{nat}$

assumes $\sigma: 0 < \sigma \leq 1$ **and** $b: \text{real } b \leq \sigma * m / 2$

shows $(\sigma * m) \ \text{gchoose } b \in \{\sigma ^ b * (m \ \text{gchoose } b) * \exp(-(\text{real } b)^2 / (\sigma * m)) \dots \sigma ^ b * (m \ \text{gchoose } b)\}$

proof (*cases m=0 \vee b=0*)

case *True*

then show *?thesis*

using *True assms* **by** *auto*

next

case *False*

then have $\sigma * m / 2 < \text{real } m$

using σ **by** *auto*

with $b \ \sigma$ **False** **have** $bm: \text{real } b < \text{real } m$

by *linarith*

then have *nonz: m gchoose b \neq 0*

by (*simp add: flip: binomial-gbinomial*)

```

have EQ: (( $\sigma * m$ ) gchoose b) * inverse (m gchoose b) =  $\sigma ^ b * (\prod_{i < b}. 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i)))$ 
using Fact-D1-73-aux <0 <  $\sigma$ > bm by blast
also have ...  $\leq \sigma ^ b * 1$ 
proof (intro mult-left-mono prod-le-1 conjI)
  fix i assume i  $\in \{.. < b\}$ 
  with b  $\sigma$  bm show  $0 \leq 1 - (1 - \sigma) * i / (\sigma * (\text{real } m - i))$ 
  by (simp add: field-split-simps)
qed (use  $\sigma$  bm in auto)
finally have upper: ( $\sigma * m$ ) gchoose b  $\leq \sigma ^ b * (m \text{ gchoose } b)$ 
using nonz by (simp add: divide-simps flip: binomial-gbinomial)
have *:  $\exp(-2 * \text{real } i / (\sigma * m)) \leq 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i))$  if
i < b for i
proof -
  have i  $\leq m$ 
  using bm that by linarith
  have exp-le:  $1 - x \geq \exp(-2 * x)$  if  $0 \leq x$   $x \leq 1/2$  for x::real
  proof -
    have exp(-2 * x)  $\leq \text{inverse}(1 + 2 * x)$ 
    using exp-ge-add-one-self that by (simp add: exp-minus)
    also have ...  $\leq 1 - x$ 
    using that by (simp add: mult-left-le field-simps)
    finally show ?thesis .
  qed
  have exp(-2 * real i / ( $\sigma * m$ )) = exp(-2 * (i / ( $\sigma * m$ )))
  by simp
  also have ...  $\leq 1 - i / (\sigma * m)$ 
  using b that by (intro exp-le) auto
  also have ...  $\leq 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i))$ 
  using  $\sigma$  b that <i  $\leq m$ > by (simp add: field-split-simps)
  finally show ?thesis .
qed
have sum real {.. < b}  $\leq \text{real } b ^ 2 / 2$ 
by (induction b) (auto simp: power2-eq-square algebra-simps)
with  $\sigma$  have exp(- (real b ^ 2) / ( $\sigma * m$ ))  $\leq \exp(- (2 * (\sum_{i < b}. i) / (\sigma * m)))$ 
by (simp add: mult-less-0-iff divide-simps)
also have ... = exp( $\sum_{i < b}. -2 * \text{real } i / (\sigma * m)$ )
by (simp add: sum-negf sum-distrib-left sum-divide-distrib)
also have ... = ( $\prod_{i < b}. \exp(-2 * \text{real } i / (\sigma * m))$ )
using exp-sum by blast
also have ...  $\leq (\prod_{i < b}. 1 - ((1 - \sigma) * i) / (\sigma * (\text{real } m - \text{real } i)))$ 
using * by (force intro: prod-mono)
finally have exp(- (real b)2 / ( $\sigma * m$ ))  $\leq (\prod_{i < b}. 1 - (1 - \sigma) * i / (\sigma * (\text{real } m - \text{real } i)))$  .
with EQ have  $\sigma ^ b * \exp(- (\text{real } b ^ 2) / (\sigma * m)) \leq ((\sigma * m) \text{ gchoose } b) * \text{inverse}(\text{real } m \text{ gchoose } b)$ 
by (simp add:  $\sigma$ )
with  $\sigma$  bm have lower:  $\sigma ^ b * (\text{real } m \text{ gchoose } b) * \exp(- (\text{real } b ^ 2) / (\sigma * m)) \leq (\sigma * m) \text{ gchoose } b$ 

```

by (simp add: field-split-simps flip: binomial-gbinomial)
 with upper show ?thesis
 by simp
 qed

Exact at zero, so cannot be done entirely using the approximation method

lemma *exp-inequality-17*:
 fixes $x::\text{real}$
 assumes $0 \leq x \leq 1/7$
 shows $1 - 4*x/3 \geq \exp(-3*x/2)$
proof (cases $x \leq 1/12$)
 case True
 have $\exp(-3*x/2) \leq 1/(1 + (3*x)/2)$
 using exp-ge-add-one-self [of $3*x/2$] assms
 by (simp add: exp-minus divide-simps)
 also have $\dots \leq 1 - 4*x/3$
 using assms True mult-left-le [of $x*12$] by (simp add: field-simps)
 finally show ?thesis .
 next
 case False
 with assms have $x \in \{1/12..1/7\}$
 by auto
 then show ?thesis
 by (approximation 12 splitting: $x=5$)
 qed

additional part

lemma *Fact-D1-75*:
 fixes $\sigma::\text{real}$ and $m b::\text{nat}$
 assumes $\sigma: 0 < \sigma < 1$ and $b: \text{real } b \leq \sigma * m / 2$ and $b': b \leq m/7$ and $\sigma': \sigma \geq 7/15$
 shows $(\sigma*m) \text{ gchoose } b \geq \exp(- (3 * \text{real } b ^ 2) / (4*m)) * \sigma ^ b * (m \text{ gchoose } b)$
proof (cases $m=0 \vee b=0$)
 case True
 then show ?thesis
 using True assms by auto
 next
 case False
 with $b b' \sigma$ have $bm: \text{real } b < \text{real } m$
 by linarith
 have $*$: $\exp(- 3 * \text{real } i / (2*m)) \leq 1 - ((1-\sigma)*i) / (\sigma * (\text{real } m - \text{real } i))$
 if $i < b$ for i
proof -
 have $im: 0 \leq i/m \wedge i/m \leq 1/7$
 using b' that by auto
 have $\exp(- 3 * \text{real } i / (2*m)) \leq 1 - 4*i / (3*m)$
 using exp-inequality-17 [OF im] by (simp add: mult.commute)
 also have $\dots \leq 1 - 8*i / (7 * (\text{real } m - \text{real } b))$

```

    using b' nle-le by (fastforce simp: field-split-simps)
  also have ... ≤ 1 - ((1-σ)*i) / (σ * (real m - real i))
proof -
  have 1: (1 - σ) / σ ≤ 8/7
    using σ σ' that
    by (simp add: field-split-simps)
  have 2: 1 / (real m - real i) ≤ 1 / (real m - real b)
    using σ σ' b' that by (simp add: field-split-simps)
  have §: (1 - σ) / (σ * (real m - real i)) ≤ 8 / (7 * (real m - real b))
    using mult-mono [OF 1 2] b' that by auto
  show ?thesis
    using mult-left-mono [OF §, of i]
    by (simp add: mult-of-nat-commute)
qed
finally show ?thesis .
qed
have EQ: ((σ*m) gchoose b) * inverse (m gchoose b) = σ ^ b * (∏ i<b. 1 -
((1-σ)*i) / (σ * (real m - real i)))
  using Fact-D1-73-aux <0<σ> bm by blast
have sum real {..<b} ≤ real b ^ 2 / 2
  by (induction b) (auto simp: power2-eq-square algebra-simps)
with σ have exp (- (3 * real b ^ 2) / (4*m)) ≤ exp (- (3 * (∑ i<b. i) /
(2*m)))
  by (simp add: mult-less-0-iff divide-simps)
also have ... = exp (∑ i<b. -3 * real i / (2*m))
  by (simp add: sum-negf sum-distrib-left sum-divide-distrib)
also have ... = (∏ i<b. exp (-3 * real i / (2*m)))
  using exp-sum by blast
also have ... ≤ (∏ i<b. 1 - ((1-σ)*i) / (σ * (real m - real i)))
  using * by (force intro: prod-mono)
finally have exp (- (3 * real b ^ 2) / (4*m)) ≤ (∏ i<b. 1 - (1-σ) * i / (σ
* (real m - real i))) .
with EQ have σ ^ b * exp (- (3 * real b ^ 2) / (4*m)) ≤ ((σ*m) gchoose b) /
(m gchoose b)
  by (simp add: assms field-simps)
with σ bm show ?thesis
  by (simp add: field-split-simps flip: binomial-gbinomial)
qed

lemma power2-12: m ≥ 12 ⇒ 25 * m2 ≤ 2m
proof (induction m)
  case 0
  then show ?case by auto
next
  case (Suc m)
  then consider m=11 | m≥12
    by linarith
  then show ?case
proof cases

```

```

case 1
then show ?thesis
  by auto
next
case 2
then have  $Suc(m+m) \leq m*3$   $m \geq 3$ 
  using Suc by auto
then have  $25 * Suc(m+m) \leq 25 * (m*m)$ 
  by (metis le-trans mult-le-mono2)
with Suc show ?thesis
  by (auto simp: power2-eq-square algebra-simps 2)
qed
qed

```

How b and m are obtained from l

definition *b-of* **where** $b\text{-of} \equiv \lambda l::nat. nat[l\text{ powr } (1/4)]$

definition *m-of* **where** $m\text{-of} \equiv \lambda l::nat. nat[l\text{ powr } (2/3)]$

definition *Big-Blue-4-1* \equiv

$$\lambda \mu l. m\text{-of } l \geq 12 \wedge l \geq (6/\mu)\text{ powr } (12/5) \wedge l \geq 15$$

$$\wedge 1 \leq 5/4 * \exp(-\text{real}((b\text{-of } l)^2) / ((\mu - 2/l) * m\text{-of } l)) \wedge \mu > 2/l$$

$$\wedge 2/l \leq (\mu - 2/l) * ((5/4)\text{ powr } (1/b\text{-of } l) - 1)$$

Establishing the size requirements for 4.1. NOTE: it doesn't become clear until SECTION 9 that all bounds involving the parameter μ must hold for a RANGE of values

lemma *Big-Blue-4-1*:

assumes $0 < \mu 0$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-Blue-4-1 } \mu l$

proof –

have $3 : 3 / \mu 0 > 0$

using *assms* **by force**

have $2 : \mu 0 * nat \lceil 3 / \mu 0 \rceil > 2$

by (*smt (verit, best) mult.commute assms of-nat-ceiling pos-less-divide-eq*)

have $\forall^\infty l. 12 \leq m\text{-of } l$

unfolding *m-of-def* **by** *real-asymp*

moreover have $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow (6 / \mu)\text{ powr } (12 / 5) \leq l$

using *assms*

apply (*intro eventually-all-geI0, real-asymp*)

by (*smt (verit, ccfv-SIG) divide-pos-pos frac-le powr-mono2*)

moreover have $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 4 \leq 5 * \exp(-((\text{real}(b\text{-of } l))^2 / ((\mu - 2/l) * m\text{-of } l)))$

proof (*intro eventually-all-geI0 [where L = nat ⌈3/μ0⌉]*)

show $\forall^\infty l. 4 \leq 5 * \exp(-((\text{real}(b\text{-of } l))^2 / ((\mu 0 - 2/l) * m\text{-of } l)))$

unfolding *b-of-def m-of-def* **using** *assms* **by** *real-asymp*

next

fix $l \mu$

assume $\S : 4 \leq 5 * \exp(-((\text{real}(b\text{-of } l))^2 / ((\mu 0 - 2/l) * m\text{-of } l)))$

and $\mu 0 \leq \mu \leq \mu 1$ **and** $l\text{el} : nat \lceil 3 / \mu 0 \rceil \leq l$

```

then have 0: m-of l > 0
  using 3 of-nat-0-eq-iff by (fastforce simp: m-of-def)
have  $\mu 0 > 2/l$ 
  using lel assms by (auto simp: divide-simps mult.commute)
then show  $4 \leq 5 * \exp(-((\text{real } (b\text{-of } l))^2 / ((\mu - 2/l) * m\text{-of } l)))$ 
  using order-trans [OF §] by (simp add: 0 < $\mu 0 \leq \mu$ > frac-le)
qed
moreover have  $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 2/l < \mu$ 
  using assms by (intro eventually-all-geI0, real-asymp, linarith)
moreover have  $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 2/l \leq (\mu - 2/l) * ((5 / 4) \text{ powr } (1 / \text{real } (b\text{-of } l)) - 1)$ 
proof -
  have  $\bigwedge l \mu. \mu 0 \leq \mu \implies \mu 0 - 2/l \leq \mu - 2/l$ 
    by (auto simp: divide-simps ge-one-powr-ge-zero mult.commute)
  show ?thesis
    using assms
    unfolding b-of-def
    apply (intro eventually-all-geI0, real-asymp)
    by (smt (verit, best) divide-le-eq-1 ge-one-powr-ge-zero mult-right-mono
of-nat-0-le-iff zero-le-divide-1-iff)
qed
ultimately show ?thesis
  by (auto simp: Big-Blue-4-1-def eventually-conj-iff all-imp-conj-distrib)
qed

```

```

context Book
begin

```

lemma Blue-4-1:

```

  assumes  $X \subseteq V$  and manyb: many-bluish X and big: Big-Blue-4-1  $\mu$  l
  shows  $\exists S T. \text{good-blue-book } X (S, T) \wedge \text{card } S \geq l \text{ powr } (1/4)$ 
proof -
  have lpowr0[simp]:  $0 \leq \lceil l \text{ powr } r \rceil$  for r
    by (metis ceiling-mono ceiling-zero powr-ge-zero)
  define b where  $b \equiv b\text{-of } l$ 
  define W where  $W \equiv \{x \in X. \text{bluish } X x\}$ 
  define m where  $m \equiv m\text{-of } l$ 
  have  $m > 0 \ m \geq 6 \ m \geq 12 \ b > 0$ 
    using big by (auto simp: Big-Blue-4-1-def m-def b-def b-of-def)
  have Wbig:  $\text{card } W \geq \text{RN } k \ m$ 
    using manyb by (simp add: W-def m-def m-of-def many-bluish-def)
  with Red-Blue-RN obtain U where  $U \subseteq W$  and U-m-Blue:  $\text{size-clique } m \ U$ 
  Blue
  by (metis W-def < $X \subseteq V$ > mem-Collect-eq no-Red-clique subset-eq)
  then obtain  $\text{card } U = m$  and clique U Blue and  $U \subseteq V$  finite U
  by (simp add: finV finite-subset size-clique-def)
  have finite X
    using < $X \subseteq V$ > finV finite-subset by auto
  have  $k \leq \text{RN } k \ m$ 

```

using $\langle m \geq 12 \rangle$ **by** (*simp add: RN-3plus'*)
moreover have $\text{card } W \leq \text{card } X$
by (*simp add: W-def <finite X> card-mono*)
ultimately have $\text{card } X \geq l$
using *Wbig l-le-k* **by** *linarith*
then have $U \neq X$
by (*metis U-m-Blue <card U = m> le-eq-less-or-eq no-Blue-clique size-clique-smaller*)
then have $U \subset X$
using *W-def <U ⊆ W>* **by** *blast*
then have $\text{card } U < \text{card } X$
by (*meson <X ⊆ V> finV finite-subset psubset-card-mono*)
with $\langle X \subseteq V \rangle$ **have** $\text{card } X \setminus U = \text{card } X - \text{card } U$
by (*meson <U ⊂ X> card-Diff-subset finV finite-subset psubset-imp-subset*)
then have $\text{real } \text{card } X \setminus U = \text{real } (\text{card } X) - m$
using $\langle \text{card } U = m \rangle$ $\text{card } U < \text{card } X$ **by** *linarith*
have [*simp*]: $m \leq \text{card } X$
using $\langle \text{card } U = m \rangle$ $\text{card } U < \text{card } X$ *nless-le* **by** *blast*
have $l \text{ powr } (2/3) \leq \text{real } l \text{ powr } 1$
using *ln0* **by** (*intro powr-mono*) *auto*
then have $m \leq l$ $m \leq k$
using *l-le-k* **by** (*auto simp: m-def m-of-def*)
then have $m < \text{RN } k$ m
using $\langle 12 \leq m \rangle$ *RN-gt2* **by** *auto*
also have $\text{cX}: \text{RN } k$ $m \leq \text{card } X$
using *Wbig <card W ≤ card X>* **by** *linarith*
finally have $\text{card } U < \text{card } X$
using $\langle \text{card } U = m \rangle$ **by** *blast*

First part of (10)

have $\text{card } U * (\mu * \text{card } X - \text{card } U) = m * (\mu * (\text{card } X - \text{card } U)) - (1 - \mu) * m^2$
using $\text{card } U < \text{card } X$ **by** (*simp add: <card U = m> algebra-simps numeral-2-eq-2*)
also have $\dots \leq \text{real } (\text{card } (\text{Blue} \cap \text{all-edges-betw-un } U (X \setminus U)))$
proof –
have *dfam: disjoint-family-on* $(\lambda u. \text{Blue} \cap \text{all-edges-betw-un } \{u\} (X \setminus U))$ U
by (*auto simp: disjoint-family-on-def all-edges-betw-un-def*)
have $\mu * (\text{card } X - \text{card } U) \leq \text{card } (\text{Blue} \cap \text{all-edges-betw-un } \{u\} (X \setminus U)) + (1 - \mu) * m$
if $u \in U$ **for** u
proof –
have *NBU: Neighbours* $\text{Blue } u \cap U = U \setminus \{u\}$
using $\langle \text{clique } U \text{ Blue} \rangle$ *Red-Blue-all singleton-not-edge that*
by (*force simp: Neighbours-def clique-def*)
then have *NBX-split:* $(\text{Neighbours } \text{Blue } u \cap X) = (\text{Neighbours } \text{Blue } u \cap (X \setminus U)) \cup (U \setminus \{u\})$
using $\langle U \subset X \rangle$ **by** *blast*
moreover have $\text{Neighbours } \text{Blue } u \cap (X \setminus U) \cap (U \setminus \{u\}) = \{\}$
by *blast*
ultimately have $\text{card}(\text{Neighbours } \text{Blue } u \cap X) = \text{card}(\text{Neighbours } \text{Blue } u \cap (X \setminus U)) + \text{card}(U \setminus \{u\})$

$(X \setminus U)) + (m - 1)$
by (*simp add: card-Un-disjoint finite-Neighbours* $\langle \text{finite } U \rangle$ $\langle \text{card } U = m \rangle$
that)
then have $\mu * (\text{card } X) \leq \text{real } (\text{card } (\text{Neighbours } \text{Blue } u \cap (X \setminus U))) + \text{real}$
 $(m - 1)$
using *W-def* $\langle U \subseteq W \rangle$ *bluish-def* *that* **by force**
then have $\mu * (\text{card } X - \text{card } U) \leq \text{card } (\text{Neighbours } \text{Blue } u \cap (X \setminus U)) +$
 $\text{real } (m - 1) - \mu * \text{card } U$
by (*smt (verit) cardU-less-X nless-le of-nat-diff right-diff-distrib'*)
then have $*$: $\mu * (\text{card } X - \text{card } U) \leq \text{real } (\text{card } (\text{Neighbours } \text{Blue } u \cap$
 $(X \setminus U))) + (1 - \mu) * m$
using *assms* **by** (*simp add: card U = m* *left-diff-distrib*)
have *inj-on* $(\lambda x. \{u, x\})$ $(\text{Neighbours } \text{Blue } u \cap X)$
by (*simp add: doubleton-eq-iff inj-on-def*)
moreover have $(\lambda x. \{u, x\})$ ' $(\text{Neighbours } \text{Blue } u \cap (X \setminus U)) \subseteq \text{Blue} \cap$
all-edges-betw-un $\{u\}$ $(X \setminus U)$
using *Blue-E* **by** (*auto simp: Neighbours-def all-edges-betw-un-def*)
ultimately have $\text{card } (\text{Neighbours } \text{Blue } u \cap (X \setminus U)) \leq \text{card } (\text{Blue} \cap$
all-edges-betw-un $\{u\}$ $(X \setminus U))$
by (*metis NBX-split card-inj-on-le finite-Blue finite-Int inj-on-Un*)
with $*$ **show** *?thesis*
by *auto*
qed
then have $(\text{card } U) * (\mu * \text{real } (\text{card } X - \text{card } U))$
 $\leq (\sum x \in U. \text{card } (\text{Blue} \cap \text{all-edges-betw-un } \{x\} (X \setminus U))) + (1 - \mu) * m$
by (*meson sum-bounded-below*)
then have $m * (\mu * (\text{card } X - \text{card } U))$
 $\leq (\sum x \in U. \text{card } (\text{Blue} \cap \text{all-edges-betw-un } \{x\} (X \setminus U))) + (1 - \mu) * m^2$
by (*simp add: sum.distrib power2-eq-square card U = m mult-ac*)
also have $\dots \leq \text{card } (\bigcup u \in U. \text{Blue} \cap \text{all-edges-betw-un } \{u\} (X \setminus U)) + (1 - \mu)$
 $* m^2$
by (*simp add: dfam card-UN-disjoint' finite U flip: UN-simps*)
finally have $m * (\mu * (\text{card } X - \text{card } U))$
 $\leq \text{card } (\bigcup u \in U. \text{Blue} \cap \text{all-edges-betw-un } \{u\} (X \setminus U)) + (1 - \mu) *$
 m^2 .
moreover have $(\bigcup u \in U. \text{Blue} \cap \text{all-edges-betw-un } \{u\} (X \setminus U)) = (\text{Blue} \cap$
all-edges-betw-un $U (X \setminus U))$
by (*auto simp: all-edges-betw-un-def*)
ultimately show *?thesis*
by *simp*
qed
also have $\dots \leq \text{edge-card } \text{Blue } U (X \setminus U)$
by (*simp add: edge-card-def*)
finally have *edge-card-XU*: $\text{edge-card } \text{Blue } U (X \setminus U) \geq \text{card } U * (\mu * \text{card } X$
 $- \text{card } U)$.
define σ **where** $\sigma \equiv \text{blue-density } U (X \setminus U)$
then have $\sigma \geq 0$ **by** (*simp add: gen-density-ge0*)
have $\sigma \leq 1$
by (*simp add: sigma-def gen-density-le1*)

```

have 6: real (6*k) ≤ real (2 + k*m)
  by (metis mult.commute ‹6≤m› mult-le-mono2 of-nat-mono trans-le-add2)
then have km: k + m ≤ Suc (k * m)
  using big l-le-k ‹m ≤ l› by linarith
have m/2 * (2 + real k * (1-μ)) ≤ m/2 * (2 + real k)
  using assms μ01 by (simp add: algebra-simps)
also have ... ≤ (k - 1) * (m - 1)
  using big l-le-k 6 ‹m≤k› by (simp add: Big-Blue-4-1-def algebra-simps add-divide-distrib
km)
finally have (m/2) * (2 + k * (1-μ)) ≤ RN k m
  using RN-times-lower' [of k m] by linarith
then have μ - 2/k ≤ (μ * card X - card U) / (card X - card U)
  using kn0 assms cardU-less-X ‹card U = m› cX by (simp add: field-simps)
also have ... ≤ σ
  using ‹m>0› ‹card U = m› cardU-less-X cardXU edge-card-XU
  by (simp add: σ-def gen-density-def divide-simps mult-ac)
finally have eq10: μ - 2/k ≤ σ .
have 2 * b / m ≤ μ - 2/k
proof -
  have 512: 5/12 ≤ (1::real)
    by simp
  with big have l powr (5/12) ≥ ((6/μ) powr (12/5)) powr (5/12)
    by (simp add: Big-Blue-4-1-def powr-mono2)
  then have lge: l powr (5/12) ≥ 6/μ
    using assms μ01 powr-powr by force
  have 2 * b ≤ 2 * (l powr (1/4) + 1)
    by (simp add: b-def b-of-def del: zero-le-ceiling distrib-left-numeral)
  then have 2*b / m + 2/l ≤ 2 * (l powr (1/4) + 1) / l powr (2/3) + 2/l
    by (simp add: m-def m-of-def frac-le ln0 del: zero-le-ceiling distrib-left-numeral)
  also have ... ≤ (2 * l powr (1/4) + 4) / l powr (2/3)
    using ln0 lpowr23 by (simp add: pos-le-divide-eq pos-divide-le-eq add-divide-distrib
algebra-simps)
  also have ... ≤ (2 * l powr (1/4) + 4 * l powr (1/4)) / l powr (2/3)
    using big by (simp add: Big-Blue-4-1-def divide-right-mono ge-one-powr-ge-zero)
  also have ... = 6 / l powr (5/12)
    by (simp add: divide-simps flip: powr-add)
  also have ... ≤ μ
    using lge assms μ01 by (simp add: divide-le-eq mult.commute)
  finally have 2*b / m + 2/l ≤ μ .
  then show ?thesis
    using l-le-k ‹m>0› ln0
    by (smt (verit, best) frac-le of-nat-0-less-iff of-nat-mono)
qed
with eq10 have 2 / (m/b) ≤ σ
  by simp
moreover have l powr (2/3) ≤ nat ⌈real l powr (2/3)⌉
  using of-nat-ceiling by blast
ultimately have ble: b ≤ σ * m / 2
  using mult-left-mono ‹σ ≥ 0› big kn0 l-le-k

```

by (*simp add: Big-Blue-4-1-def powr-diff b-def m-def divide-simps*)
then have $\sigma > 0$
using $\langle 0 < b \rangle \langle 0 \leq \sigma \rangle$ *less-eq-real-def* **by** *force*

define Φ **where** $\Phi \equiv \sum v \in X \setminus U. \text{card} (\text{Neighbours Blue } v \cap U)$ *choose* b
 now for the material between (10) and (11)

have $\sigma * \text{real } m / 2 \leq m$
using $\langle \sigma \leq 1 \rangle \langle m > 0 \rangle$ **by** *auto*
with *ble* **have** $b \leq m$
by *linarith*
have $\mu^b * 1 * \text{card } X \leq (5/4 * \sigma^b) * (5/4 * \exp(- \text{real}(b^2) / (\sigma * m))) * (5/4 * (\text{card } X - m))$
proof (*intro mult-mono*)
have $2/k \leq 2/l$
by (*simp add: l-le-k frac-le ln0*)
also have $\dots \leq (\mu - 2/l) * ((5/4) \text{powr } (1/b) - 1)$
using *big* **by** (*simp add: Big-Blue-4-1-def b-def*)
also have $\dots \leq \sigma * ((5/4) \text{powr } (1/b) - 1)$
using $2 \langle 0 < b \rangle$ *eq10* **by** *auto*
finally have $2 / \text{real } k \leq \sigma * ((5/4) \text{powr } (1/b) - 1)$.
then have $1: \mu \leq (5/4) \text{powr}(1/b) * \sigma$
using *eq10* $\langle b > 0 \rangle$ **by** (*simp add: algebra-simps*)
show $\mu^b \leq 5/4 * \sigma^b$
using *power-mono*[*OF 1, of b*] *assms* $\langle \sigma > 0 \rangle \langle b > 0 \rangle$ $\mu 01$
by (*simp add: powr-mult powr-powr flip: powr-realpow*)
have $\mu - 2/l \leq \sigma$
using 2 *eq10* **by** *linarith*
moreover have $2/l < \mu$
using *big* **by** (*auto simp: Big-Blue-4-1-def*)
ultimately have $\exp(- \text{real}(b^2) / ((\mu - 2/l) * m)) \leq \exp(- \text{real}(b^2) / (\sigma * m))$
using $\langle \sigma > 0 \rangle \langle m > 0 \rangle$ **by** (*simp add: frac-le*)
then show $1 \leq 5/4 * \exp(- \text{real}(b^2) / (\sigma * \text{real } m))$
using *big unfolding* *Big-Blue-4-1-def b-def m-def*
by (*smt (verit, best) divide-minus-left frac-le mult-left-mono*)
have $25 * (\text{real } m * \text{real } m) \leq 2 \text{powr } m$
using *of-nat-mono* [*OF power2-12* [*OF* $\langle 12 \leq m \rangle$]] **by** (*simp add: power2-eq-square powr-realpow*)
then have $\text{real } (5 * m) \leq 2 \text{powr } (\text{real } m / 2)$
by (*simp add: powr-half-sqrt-powr power2-eq-square real-le-rsqrt*)
moreover
have $\text{card } X > 2 \text{powr } (m/2)$
by (*metis RN-commute RN-lower-nodiag* $\langle 6 \leq m \rangle \langle m \leq k \rangle$ *add-leE less-le-trans* *cX numeral-Bit0 of-nat-mono*)
ultimately have $5 * m \leq \text{real } (\text{card } X)$
by *linarith*
then show $\text{card } X \leq 5/4 * (\text{card } X - m)$
using $\langle \text{card } U = m \rangle$ *cardU-less-X* **by** *simp*

qed (use $\langle 0 \leq \sigma \rangle$ in auto)
also have $\dots = (125/64) * (\sigma \wedge b) * \exp(-(\text{real } b)^2 / (\sigma * m)) * (\text{card } X - m)$
by simp
also have $\dots \leq 2 * (\sigma \wedge b) * \exp(-(\text{real } b)^2 / (\sigma * m)) * (\text{card } X - m)$
by (intro mult-right-mono) (auto simp: $\langle 0 \leq \sigma \rangle$)
finally have $\mu \wedge b / 2 * \text{card } X \leq \sigma \wedge b * \exp(-\text{of-nat } (b^2) / (\sigma * m)) * \text{card } (X \setminus U)$
by (simp add: $\langle \text{card } U = m \rangle$ cardXU real-cardXU)
also have $\dots \leq 1 / (m \text{ choose } b) * ((\sigma * m) \text{ gchoose } b) * \text{card } (X \setminus U)$
proof (intro mult-right-mono)
have $0 < \text{real } m \text{ gchoose } b$
by (metis $\langle b \leq m \rangle$ binomial-gbinomial of-nat-0-less-iff zero-less-binomial-iff)
then have $\sigma \wedge b * ((\text{real } m \text{ gchoose } b) * \exp(-((\text{real } b)^2 / (\sigma * \text{real } m)))) \leq$
 $\sigma * \text{real } m \text{ gchoose } b$
using Fact-D1-73 [OF $\langle \sigma > 0 \rangle \langle \sigma \leq 1 \rangle$ ble] $\langle b \leq m \rangle$ cardU-less-X $\langle 0 < \sigma \rangle$
by (simp add: field-split-simps binomial-gbinomial)
then show $\sigma \wedge b * \exp(-\text{real } (b^2) / (\sigma * m)) \leq 1 / (m \text{ choose } b) * (\sigma * m$
gchoose b)
using $\langle b \leq m \rangle$ cardU-less-X $\langle 0 < \sigma \rangle \langle 0 < m \text{ gchoose } b \rangle$
by (simp add: field-split-simps binomial-gbinomial)
qed auto
also have $\dots \leq 1 / (m \text{ choose } b) * \Phi$
unfolding mult.assoc
proof (intro mult-left-mono)
have eq: edge-card Blue U $(X \setminus U) = (\sum i \in X \setminus U. \text{card } (\text{Neighbours Blue } i \cap U))$
proof (intro edge-card-eq-sum-Neighbours)
show finite $(X \setminus U)$
by (meson $\langle X \subseteq V \rangle$ finV finite-Diff finite-subset)
qed (use disjnt-def Blue-E in auto)
have $(\sum i \in X \setminus U. \text{card } (\text{Neighbours Blue } i \cap U)) / (\text{real } (\text{card } X) - m) =$
blue-density U $(X \setminus U) * m$
using $\langle m > 0 \rangle$ **by** (simp add: gen-density-def real-cardXU $\langle \text{card } U = m \rangle$ eq
divide-simps)
then have $*$: $(\sum i \in X \setminus U. \text{real } (\text{card } (\text{Neighbours Blue } i \cap U)) /_{\mathbb{R}} \text{real } (\text{card } (X \setminus U))) = \sigma * m$
by (simp add: σ -def divide-inverse-commute real-cardXU flip: sum-distrib-left)
have mbinomial $(\sum i \in X \setminus U. \text{real } (\text{card } (\text{Neighbours Blue } i \cap U)) /_{\mathbb{R}} (\text{card } (X \setminus U))) b$
 $\leq (\sum i \in X \setminus U. \text{inverse } (\text{real } (\text{card } (X \setminus U)))) * \text{mbinomial } (\text{card } (\text{Neighbours Blue } i \cap U)) b$
proof (rule convex-on-sum)
show finite $(X \setminus U)$
using cardU-less-X zero-less-diff **by** fastforce
show convex-on UNIV $(\lambda a. \text{mbinomial } a b)$
by (simp add: $\langle 0 < b \rangle$ convex-mbinomial)
show $(\sum i \in X \setminus U. \text{inverse } (\text{card } (X \setminus U))) = 1$
using cardU-less-X cardXU **by** force
qed (use $\langle U \subset X \rangle$ in auto)
with ble

```

show  $(\sigma * m \text{ choose } b) * \text{card } (X \setminus U) \leq \Phi$ 
  unfolding *  $\Phi$ -def
    by (simp add: cardU-less-X cardXU binomial-gbinomial divide-simps flip: sum-distrib-left sum-divide-distrib)
qed auto
finally have 11:  $\mu \wedge b / 2 * \text{card } X \leq \Phi / (m \text{ choose } b)$ 
  by simp

define  $\Omega$  where  $\Omega \equiv \text{nsets } U \ b$  — Choose a random subset of size  $b$ 
have  $\text{card } \Omega$ :  $\text{card } \Omega = m \text{ choose } b$ 
  by (simp add:  $\Omega$ -def  $\langle \text{card } U = m \rangle$ )
then have  $\text{fin } \Omega$ : finite  $\Omega$  and  $\Omega \neq \{\}$  and  $\text{card } \Omega > 0$ 
  using  $\langle b \leq m \rangle$  not-less by fastforce+
define  $M$  where  $M \equiv \text{uniform-count-measure } \Omega$ 
interpret  $P$ : prob-space  $M$ 
  using  $M$ -def  $\langle b \leq m \rangle$   $\text{card } \Omega$  fin $\Omega$  prob-space-uniform-count-measure by force
have measure-eq: measure  $M \ C = (\text{if } C \subseteq \Omega \text{ then } \text{card } C / \text{card } \Omega \text{ else } 0)$  for  $C$ 
  by (simp add: M-def fin $\Omega$  measure-uniform-count-measure-if)

define Int-NB where Int-NB  $\equiv \lambda S. \bigcap_{v \in S. \text{Neighbours Blue } v} v \cap (X \setminus U)$ 
have sum-card-NB:  $(\sum A \in \Omega. \text{card } (\bigcap (\text{Neighbours Blue } 'A) \cap Y))$ 
   $= (\sum v \in Y. \text{card } (\text{Neighbours Blue } v \cap U) \text{ choose } b)$ 
  if finite  $Y$   $Y \subseteq X \setminus U$  for  $Y$ 
  using that
proof (induction  $Y$ )
  case (insert  $y \ Y$ )
    have *:  $\Omega \cap \{A. \forall x \in A. y \in \text{Neighbours Blue } x\} = \text{nsets } (\text{Neighbours Blue } y \cap U) \ b$ 
       $\Omega \cap - \{A. \forall x \in A. y \in \text{Neighbours Blue } x\} = \Omega - \text{nsets } (\text{Neighbours Blue } y \cap U) \ b$ 
       $[\text{Neighbours Blue } y \cap U]^b \subseteq \Omega$ 
    using insert.prems by (auto simp:  $\Omega$ -def nsets-def in-Neighbours-iff insert-commute)
    then show ?case
      using insert fin $\Omega$ 
      by (simp add: Int-insert-right sum-Suc sum.If-cases if-distrib [of card] sum.subset-diff flip: insert.IH)
qed auto

have  $(\sum x \in \Omega. \text{card } (\text{if } x = \{\} \text{ then } UNIV \text{ else } \bigcap (\text{Neighbours Blue } 'x) \cap (X \setminus U)))$ 
   $= (\sum x \in \Omega. \text{card } (\bigcap (\text{Neighbours Blue } 'x) \cap (X \setminus U)))$ 
  unfolding  $\Omega$ -def nsets-def using  $\langle 0 < b \rangle$  by (force intro: sum.cong)
also have ...  $= (\sum v \in X \setminus U. \text{card } (\text{Neighbours Blue } v \cap U) \text{ choose } b)$ 
  by (metis sum-card-NB  $\langle X \subseteq V \rangle$  dual-order.refl fin V finite-Diff rev-finite-subset)
finally have sum  $(\text{card } o \text{Int-NB}) \ \Omega = \Phi$ 
  by (simp add:  $\Omega$ -def  $\Phi$ -def Int-NB-def)
moreover
have ennreal  $(P.\text{expectation } (\lambda S. \text{card } (\text{Int-NB } S))) = \text{sum } (\text{card } o \text{Int-NB}) \ \Omega / (\text{card } \Omega)$ 

```

```

    using integral-uniform-count-measure M-def finΩ by fastforce
  ultimately have P: P.expectation (λS. card (Int-NB S)) = Φ / (m choose b)
    by (metis Bochner-Integration.integral-nonneg cardΩ divide-nonneg-nonneg
ennreal-inj of-nat-0-le-iff)
  have False if ∧S. S ∈ Ω ⇒ card (Int-NB S) < Φ / (m choose b)
  proof -
    define L where L ≡ (λS. Φ / real (m choose b) - card (Int-NB S)) ‘ Ω
    have finite L L ≠ {}
      using L-def finΩ <Ω≠{}> by blast+
    define ε where ε ≡ Min L
    have ε > 0
      using that finΩ <Ω ≠ {}> by (simp add: L-def ε-def)
    then have ∧S. S ∈ Ω ⇒ card (Int-NB S) ≤ Φ / (m choose b) - ε
      using Min-le [OF <finite L>] by (fastforce simp: algebra-simps ε-def L-def)
    then have P.expectation (λS. card (Int-NB S)) ≤ Φ / (m choose b) - ε
      using P P.not-empty not-integrable-integral-eq <ε > 0>
    by (intro P.integral-le-const) (fastforce simp: M-def space-uniform-count-measure)+
    then show False
      using P <0 < ε> by auto
  qed
  then obtain S where S ∈ Ω and Sge: card (Int-NB S) ≥ Φ / (m choose b)
    using linorder-not-le by blast
  then have S ⊆ U
    by (simp add: Ω-def nsets-def subset-iff)
  have card S = b clique S Blue
    using <S ∈ Ω> <U ⊆ V> <clique U Blue> smaller-clique
    unfolding Ω-def nsets-def size-clique-def by auto
  have Φ / (m choose b) ≥ μ ^ b * card X / 2
    using 11 by simp
  then have S: card (Int-NB S) ≥ μ ^ b * card X / 2
    using Sge by linarith
  obtain v where v ∈ S
    using <0 < b> <card S = b> by fastforce
  have all-edges-betw-un S (S ∪ Int-NB S) ⊆ Blue
    using <clique S Blue>
    unfolding all-edges-betw-un-def Neighbours-def clique-def Int-NB-def by fastforce
  then have good-blue-book X (S, Int-NB S)
    using <S ⊆ U> <v ∈ S> <U ⊂ X> S <card S = b>
    unfolding good-blue-book-def book-def size-clique-def Int-NB-def disjnt-iff
    by blast
  then show ?thesis
    by (metis <card S = b> b-def b-of-def of-nat-ceiling)
  qed

```

Lemma 4.3

```

proposition bblue-step-limit:
  assumes big: Big-Blue-4-1 μ l
  shows card (Step-class {bblue-step}) ≤ l powr (3/4)
proof -

```

```

define BBLUES where BBLUES  $\equiv \lambda r. \{m. m < r \wedge \text{stepper-kind } m = \text{bblue-step}\}$ 
have cardB-ge:  $\text{card } (Bseq\ n) \geq b\text{-of } l * \text{card}(BBLUES\ n)$ 
for n
proof (induction n)
case 0 then show ?case by (auto simp: BBLUES-def)
next
case (Suc n)
show ?case
proof (cases stepper-kind n = bblue-step)
case True
have [simp]:  $\text{card } (\text{insert } n\ (BBLUES\ n)) = \text{Suc } (\text{card } (BBLUES\ n))$ 
by (simp add: BBLUES-def)
have card-B':  $\text{card } (Bseq\ (\text{Suc } n)) \geq b\text{-of } l * \text{card } (BBLUES\ n)$ 
using Suc.IH
by (meson Bseq-Suc-subset card-mono finite-Bseq le-trans)

define S where S  $\equiv \text{fst } (\text{choose-blue-book } (Xseq\ n, Yseq\ n, Aseq\ n, Bseq\ n))$ 
have BSuc:  $Bseq\ (\text{Suc } n) = Bseq\ n \cup S$ 
and manyb: many-bluish (Xseq n)
and cbb:  $\text{choose-blue-book } (Xseq\ n, Yseq\ n, Aseq\ n, Bseq\ n) = (S, Xseq\ (\text{Suc } n))$ 
and same:  $Aseq\ (\text{Suc } n) = Aseq\ n\ Yseq\ (\text{Suc } n) = Yseq\ n$ 
using True
by (force simp: S-def step-kind-defs next-state-def split: prod.split if-split-asm)+

have l14:  $l\ \text{pour } (1/4) \leq \text{card } S$ 
using Blue-4-1 [OF Xseq-subset-V manyb big]
by (smt (verit, best) choose-blue-book-works best-blue-book-is-best cbb
finite-Xseq of-nat-mono)
then have ble:  $b\text{-of } l \leq \text{card } S$ 
using b-of-def nat-ceiling-le-eq by presburger
have S: good-blue-book (Xseq n) (S, Xseq (Suc n))
by (metis cbb choose-blue-book-works finite-Xseq)
then have  $\text{card } S \leq \text{best-blue-book-card } (Xseq\ n)$ 
by (simp add: best-blue-book-is-best finite-Xseq)
have finS: finite S
using ln0 l14 card.infinite by force
have disjnt (Bseq n) (Xseq n)
using valid-state-seq [of n]
by (auto simp: Bseq-def Xseq-def valid-state-def disjoint-state-def disjnt-iff
split: prod.split-asm)
then have dBS: disjnt (Bseq n) S
using S cbb by (force simp: good-blue-book-def book-def disjnt-iff)
have eq:  $BBLUES(\text{Suc } n) = \text{insert } n\ (BBLUES\ n)$ 
using less-Suc-eq True unfolding BBLUES-def by blast
then have  $b\text{-of } l * \text{card } (BBLUES\ (\text{Suc } n)) = b\text{-of } l + b\text{-of } l * \text{card } (BBLUES\ n)$ 

```

```

    by auto
  also have ... ≤ card (Bseq n) + card S
    using ble card-B' Suc.IH by linarith
  also have ... ≤ card (Bseq n ∪ S)
    using ble dBS by (simp add: card-Un-disjnt finS finite-Bseq)
  finally have **: b-of l * card (BBLUES (Suc n)) ≤ card (Bseq (Suc n))
    using order.trans BSuc by argo
  then show ?thesis
    by (simp add: BBLUES-def)
next
case False
then have BBLUES(Suc n) = BBLUES n
  using less-Suc-eq by (auto simp: BBLUES-def)
then show ?thesis
  by (metis Bseq-Suc-subset Suc.IH card-mono finite-Bseq le-trans)
qed
qed
{ assume §: card (Step-class {bblue-step}) > l powr (3/4)
  then have fin: finite (Step-class {bblue-step})
    using card.infinite by fastforce
  then obtain n where n: (Step-class {bblue-step}) = {m. m < n ∧ stepper-kind
m = bblue-step}
    using Step-class-iterates by blast
  with § have card-gt: card {m. m < n ∧ stepper-kind m = bblue-step} > l powr
(3/4)
    by (simp add: n)
  have l = l powr (1/4) * l powr (3/4)
    by (simp flip: powr-add)
  also have ... ≤ b-of l * l powr (3/4)
    by (simp add: b-of-def mult-mono)
  also have ... ≤ b-of l * card {m. m < n ∧ stepper-kind m = bblue-step}
    using card-gt less-eq-real-def by fastforce
  also have ... ≤ card (Bseq n)
    using cardB-ge step-of-nat-mono unfolding BBLUES-def by blast
  also have ... < l
    by (simp add: Bseq-less-l)
  finally have False
    by simp
}
then show ?thesis by force
qed

```

lemma *red-steps-eq-A*:

```

  defines REDS ≡ λr. {i. i < r ∧ stepper-kind i = red-step}
  shows card (REDS n) = card (Aseq n)
proof (induction n)
  case 0
  then show ?case

```

```

    by (auto simp: REDS-def)
next
case (Suc n)
show ?case
proof (cases stepper-kind n = red-step)
  case True
  then have [simp]: REDS (Suc n) = insert n (REDS n) card (insert n (REDS
n)) = Suc (card (REDS n))
    by (auto simp: REDS-def)
  have Aeq: Aseq (Suc n) = insert (choose-central-vx (Xseq n, Yseq n, Aseq n, Bseq
n)) (Aseq n)
    using Suc.prem1 True
    by (auto simp: step-kind-defs next-state-def split: if-split-asm prod.split)
  have finite (Xseq n)
    using finite-Xseq by presburger
  then have choose-central-vx (Xseq n, Yseq n, Aseq n, Bseq n) ∈ Xseq n
    using True
  by (simp add: step-kind-defs choose-central-vx-X split: if-split-asm prod.split-asm)
  moreover have disjnt (Xseq n) (Aseq n)
    using valid-state-seq by (simp add: valid-state-def disjoint-state-def)
  ultimately have choose-central-vx (Xseq n, Yseq n, Aseq n, Bseq n) ∉ Aseq n
    by (simp add: disjnt-iff)
  then show ?thesis
    by (simp add: Aeq Suc.IH finite-Aseq)
next
case False
then have REDS(Suc n) = REDS n
  using less-Suc-eq unfolding REDS-def by blast
moreover have Aseq (Suc n) = Aseq n
  using False
  by (auto simp: step-kind-defs degree-reg-def next-state-def split: prod.split)
ultimately show ?thesis
  using Suc.IH by presburger
qed
qed

```

proposition *red-step-eq-Aseq*: $\text{card} (\text{Step-class } \{\text{red-step}\}) = \text{card} (\text{Aseq halted-point})$

proof –

```

  have card{i. i < halted-point ∧ stepper-kind i = red-step} = card (Aseq halted-point)
    by (rule red-steps-eq-A)
  moreover have (Step-class {red-step}) = {i. i < halted-point ∧ stepper-kind i
= red-step}
    using halted-point-minimal' by (fastforce simp: Step-class-def)
  ultimately show ?thesis
    by argo
qed

```

proposition *red-step-limit*: $\text{card} (\text{Step-class } \{\text{red-step}\}) < k$

using *Aseq-less-k red-step-eq-Aseq* by presburger

proposition *bblue-dboost-step-limit*:
assumes *big*: *Big-Blue-4-1* μ *l*
shows $\text{card}(\text{Step-class } \{\text{bblue-step}\}) + \text{card}(\text{Step-class } \{\text{dboost-step}\}) < l$
proof –
define *BDB* **where** $BDB \equiv \lambda r. \{i. i < r \wedge \text{stepper-kind } i \in \{\text{bblue-step}, \text{dboost-step}\}\}$
have $*$: $\text{card}(BDB \ n) \leq \text{card } B$ — looks clunky but gives access to all state components
if *stepper* $n = (X, Y, A, B)$ **for** $n \ X \ Y \ A \ B$
using *that*
proof (*induction* n *arbitrary*: $X \ Y \ A \ B$)
case 0
then show *?case*
by (*auto simp*: *BDB-def*)
next
case (*Suc* n)
obtain $X' \ Y' \ A' \ B'$ **where** *step-n*: *stepper* $n = (X', Y', A', B')$
by (*metis surj-pair*)
then obtain *valid-state* (X', Y', A', B') **and** *V-state* (X', Y', A', B')
and *disjst*: *disjoint-state* (X', Y', A', B') **and** *finite* X'
by (*metis finX valid-state-def valid-state-stepper*)
have $B' \subseteq B$
using *Suc.prem*s **by** (*auto simp*: *next-state-def Let-def degree-reg-def step-n split: prod.split-asm if-split-asm*)
show *?case*
proof (*cases* *stepper-kind* $n \in \{\text{bblue-step}, \text{dboost-step}\}$)
case *True*
then have $BDB \ (\text{Suc } n) = \text{insert } n \ (BDB \ n)$
by (*auto simp*: *BDB-def*)
moreover have $\text{card}(\text{insert } n \ (BDB \ n)) = \text{Suc}(\text{card}(BDB \ n))$
by (*simp add*: *BDB-def*)
ultimately have $\text{card-Suc}[simp]: \text{card}(BDB \ (\text{Suc } n)) = \text{Suc}(\text{card}(BDB \ n))$
by *presburger*
have *card-B'*: $\text{card}(BDB \ n) \leq \text{card } B'$
using *step-n BDB-def Suc.IH* **by** *blast*
consider *stepper-kind* $n = \text{bblue-step} \mid \text{stepper-kind } n = \text{dboost-step}$
using *True* **by** *force*
then have *Bigger*: $B' \subset B$
proof *cases*
case 1
then have \neg *termination-condition* $X' \ Y'$
by (*auto simp*: *stepper-kind-def step-n*)
with 1 **obtain** S **where** $A' = A \ Y' = Y$ **and** *manyb*: *many-bluish* X'
and *ccb*: *choose-blue-book* $(X', Y, A, B') = (S, X)$ **and** *le-cardB*: $B = B' \cup S$
using *Suc.prem*s
by (*auto simp*: *step-kind-defs next-state-def step-n split: prod.split-asm if-split-asm*)

then obtain $X' \subseteq V$ *finite* X'
using *Xseq-subset-V* \langle *finite* $X'\rangle$ *step-n stepper-XYseq* **by** *blast*
then have $l \text{ powr } (1/4) \leq \text{real } (\text{card } S)$
using *Blue-4-1* [*OF - manyb big*]
by (*smt (verit, best) of-nat-mono best-blue-book-is-best cbb choose-blue-book-works*)
then have $S \neq \{\}$
using *ln0* **by** *fastforce*
moreover have *disjnt* $B' S$
using *choose-blue-book-subset* [*OF* \langle *finite* $X'\rangle$] *disjst cbb*
unfolding *disjoint-state-def*
by (*smt (verit) in-mono* $\langle A' = A \rangle \langle Y' = Y \rangle$ *disjnt-iff old.prod.case*)
ultimately show *?thesis*
by (*metis* $\langle B' \subseteq B \rangle$ *disjnt-Un1 disjnt-self-iff-empty le-cardB psubsetI*)
next
case *2*
then have *choose-central-vx* $(X', Y', A', B') \in X'$
unfolding *step-kind-defs*
by (*auto simp:* \langle *finite* $X'\rangle$ *choose-central-vx-X step-n split: if-split-asm*)
moreover have *disjnt* $B' X'$
using *disjst disjnt-sym* **by** (*force simp:* *disjoint-state-def*)
ultimately have *choose-central-vx* $(X', Y', A', B') \notin B'$
by (*meson disjnt-iff*)
then show *?thesis*
using *2 Suc.prem*s
by (*auto simp:* *step-kind-defs next-state-def step-n split: if-split-asm*)
qed
moreover have *finite* B
by (*metis Suc.prem*s *V-state-stepper finB*)
ultimately show *?thesis*
by (*metis card-B' card-Suc card-seteq le-trans not-less-eq-eq psubset-eq*)
next
case *False*
then have *BDB* $(\text{Suc } n) = \text{BDB } n$
using *less-Suc-eq unfolding BDB-def* **by** *blast*
with $\langle B' \subseteq B \rangle$ *Suc* **show** *?thesis*
by (*metis V-state-stepper card-mono finB le-trans step-n*)
qed
qed
have *less-l:* $\text{card } (\text{BDB } n) < l$ **for** n
by (*meson card-B-limit * order.trans linorder-not-le prod-cases4*)
moreover have *fin:* $\bigwedge n. \text{finite } (\text{BDB } n) \text{ incseq } \text{BDB}$
by (*auto simp:* *BDB-def incseq-def*)
ultimately have ***:* $\bigcup^{\infty} n. (\text{range } \text{BDB}) = \text{BDB } n$
using *Union-incseq-finite* **by** *blast*
then have *finite* $(\bigcup (\text{range } \text{BDB}))$
using *BDB-def eventually-sequentially* **by** *force*
moreover have *Uneq:* $\bigcup (\text{range } \text{BDB}) = \text{Step-class } \{\text{bblue-step}, \text{dboost-step}\}$
by (*auto simp:* *Step-class-def BDB-def*)
ultimately have *fin:* *finite* $(\text{Step-class } \{\text{bblue-step}, \text{dboost-step}\})$

```

    by fastforce
  obtain n where  $\bigcup$  (range BDB) = BDB n
    using ** by force
  then have  $\text{card } (BDB\ n) = \text{card } (\text{Step-class } \{bblue\text{-step}\} \cup \text{Step-class } \{dboost\text{-step}\})$ 
    by (metis Step-class-insert Uneq)
  also have ... =  $\text{card } (\text{Step-class } \{bblue\text{-step}\}) + \text{card } (\text{Step-class } \{dboost\text{-step}\})$ 
    by (simp add: card-Un-disjnt disjnt-Step-class)
  finally show ?thesis
    by (metis less-l)
qed

end

end

```

4 Red Steps: theorems

theory Red-Steps imports Big-Blue-Steps

begin

Bhavik Mehta: choose-free Ramsey lower bound that's okay for very small p

lemma Ramsey-number-lower-simple:

fixes $p::\text{real}$

assumes $n: n^k * p \text{ powr } (k^2 / 4) + n^l * \exp(-p * l^2 / 4) < 1$

assumes $p01: 0 < p < 1$ and $k > 1\ l > 1$

shows $\neg \text{is-Ramsey-number } k\ l\ n$

proof (rule Ramsey-number-lower-gen)

have $(n \text{ choose } k) * p^{(k \text{ choose } 2)} \leq n^k * p \text{ powr } (\text{real } k^2 / 4)$

proof -

have $(n \text{ choose } k) * p^{(k \text{ choose } 2)} \leq \text{real } (\text{Suc } n - k)^k * p^{(k \text{ choose } 2)}$

using choose-le-power $p01$ by simp

also have ... = $\text{real } (\text{Suc } n - k)^k * p \text{ powr } (k * (\text{real } k - 1) / 2)$

by (metis choose-two-real $p01(1)$ powr-realpow)

also have ... $\leq n^k * p \text{ powr } (\text{real } k^2 / 4)$

using $p01 \langle k > 1 \rangle$ by (intro mult-mono powr-mono[^]) (auto simp: power2-eq-square)

finally show ?thesis .

qed

moreover

have $\text{real } (n \text{ choose } l) * (1 - p)^{(l \text{ choose } 2)} \leq n^l * \exp(-p * \text{real } l^2 / 4)$

proof -

show ?thesis

proof (intro mult-mono)

show $\text{real } (n \text{ choose } l) \leq n^l$

by (metis binomial-eq-0-iff binomial-le-pow not-le of-nat-le-iff zero-le)

have $l * p \leq 2 * (1 - \text{real } l) * -p$

using $assms$ by (auto simp: algebra-simps)

also have ... $\leq 2 * (1 - \text{real } l) * \ln(1 - p)$

```

    using p01 <l>1> ln-add-one-self-le-self2 [of -p]
    by (intro mult-left-mono-neg) auto
  finally have real l * (real l * p) ≤ real l * (2 * (1 - real l) * ln (1-p))
    using mult-left-mono <l>1> by fastforce
  with p01 show (1 - p)^(l choose 2) ≤ exp (- p * (real l)2 / 4)
    by (simp add: field-simps power2-eq-square powr-def choose-two-real flip:
powr-realpow)
  qed (use p01 in auto)
  qed
  ultimately
  show real (n choose k) * p^(k choose 2) + real (n choose l) * (1 - p)^(l choose
2) < 1
    using n by auto
  qed (use p01 in auto)

```

```

context Book
begin

```

4.1 Density-boost steps

4.1.1 Observation 5.5

lemma *sum-Weight-ge0*:

```

  assumes X ⊆ V Y ⊆ V disjnt X Y
  shows (∑ x∈X. ∑ x'∈X. Weight X Y x x') ≥ 0
proof -
  have finite X finite Y
    using assms finV finite-subset by blast+
  with Red-E have EXY: edge-card Red X Y = (∑ x∈X. card (Neighbours Red
x ∩ Y))
    by (metis <disjnt X Y> disjnt-sym edge-card-commute edge-card-eq-sum-Neighbours)
  have (∑ x∈X. ∑ x'∈X. red-density X Y * card (Neighbours Red x ∩ Y))
    = red-density X Y * card X * edge-card Red X Y
    using assms Red-E
  by (simp add: EXY power2-eq-square edge-card-eq-sum-Neighbours flip: sum-distrib-left)
  also have ... = red-density X Y2 * card X2 * card Y
    by (simp add: power2-eq-square gen-density-def)
  also have ... = ((∑ i∈Y. card (Neighbours Red i ∩ X)) / (real (card X) * real
(card Y)))2 * (card X)2 * card Y
    using Red-E <finite Y> assms
  by (simp add: psubset-eq gen-density-def edge-card-eq-sum-Neighbours)
  also have ... ≤ (∑ y∈Y. real ((card (Neighbours Red y ∩ X))2))
proof (cases card Y = 0)
  case False
  then have (∑ x∈Y. real (card (Neighbours Red x ∩ X)))2
    ≤ (∑ y∈Y. (real (card (Neighbours Red y ∩ X)))2) * card Y
    using sum-squared-le-sum-of-squares by blast
  then show ?thesis
    using assms False by (simp add: divide-simps power2-eq-square sum-nonneg)

```

qed (*auto simp: sum-nonneg*)
also have $\dots = (\sum x \in X. \sum x' \in X. \text{real} (\text{card} (\text{Neighbours Red } x \cap \text{Neighbours Red } x' \cap Y)))$
proof –
define $f :: 'a \times 'a \times 'a \Rightarrow 'a \times 'a \times 'a$ **where** $f \equiv \lambda(y, (x, x')). (x, (x', y))$
have $f : \text{bij-betw } f$ (*SIGMA y:Y. (Neighbours Red y \cap X) \times (Neighbours Red y \cap X)*)
 $(\text{SIGMA } x:X. \text{SIGMA } x':X. \text{Neighbours Red } x \cap \text{Neighbours Red } x' \cap Y)$
by (*auto simp: f-def bij-betw-def inj-on-def image-iff in-Neighbours-iff doubleton-eq-iff insert-commute*)
have $(\sum y \in Y. (\text{card} (\text{Neighbours Red } y \cap X))^2) = \text{card}(\text{SIGMA } y:Y. (\text{Neighbours Red } y \cap X) \times (\text{Neighbours Red } y \cap X))$
by (*simp add: <finite Y> finite-Neighbours power2-eq-square*)
also have $\dots = \text{card}(\text{Sigma } X (\lambda x. \text{Sigma } X (\lambda x'. \text{Neighbours Red } x \cap \text{Neighbours Red } x' \cap Y)))$
using *bij-betw-same-card f by blast*
also have $\dots = (\sum x \in X. \sum x' \in X. \text{card} (\text{Neighbours Red } x \cap \text{Neighbours Red } x' \cap Y))$
by (*simp add: <finite X> finite-Neighbours power2-eq-square*)
finally
have $(\sum y \in Y. (\text{card} (\text{Neighbours Red } y \cap X))^2) = (\sum x \in X. \sum x' \in X. \text{card} (\text{Neighbours Red } x \cap \text{Neighbours Red } x' \cap Y))$.
then show *?thesis*
by (*simp flip: of-nat-sum of-nat-power*)
qed
finally have $(\sum x \in X. \sum y \in X. \text{red-density } X \ Y * \text{card} (\text{Neighbours Red } x \cap Y)) \leq (\sum x \in X. \sum y \in X. \text{real} (\text{card} (\text{Neighbours Red } x \cap \text{Neighbours Red } y \cap Y)))$.
then show *?thesis*
by (*simp add: Weight-def sum-subtractf inverse-eq-divide flip: sum-divide-distrib*)
qed
end

4.1.2 Lemma 5.6

definition *Big-Red-5-6-Ramsey* \equiv

$$\begin{aligned}
& \lambda c \ l. \text{nat} \lceil \text{real } l \text{ powr } (3/4) \rceil \geq 3 \\
& \wedge (l \text{ powr } (3/4) * (c - 1/32) \leq -1) \\
& \wedge (\forall k \geq l. k * (c * l \text{ powr } (3/4) * \ln k - k \text{ powr } (7/8) / 4) \leq -1)
\end{aligned}$$

establishing the size requirements for 5.6

lemma *Big-Red-5-6-Ramsey*:

assumes $0 < c < 1/32$

shows $\forall^\infty l. \text{Big-Red-5-6-Ramsey } c \ l$

proof –

have *D34*: $\bigwedge l \ k. l \leq k \implies c * \text{real } l \text{ powr } (3/4) \leq c * \text{real } k \text{ powr } (3/4)$

by (*simp add: assms powr-mono2*)

have $D0: \forall^\infty l. l * (c * l \text{ powr } (3/4) * \ln l - l \text{ powr } (7/8) / 4) \leq -1$
using $\langle c > 0 \rangle$ **by** *real-asymp*
have $\bigwedge l k. l \leq k \implies c * \text{real } l \text{ powr } (3/4) * \ln k \leq c * \text{real } k \text{ powr } (3/4) * \ln k$
using $D34$ *le-eq-less-or-eq mult-right-mono* **by** *fastforce*
then have $D: \forall^\infty l. \forall k \geq l. k * (c * l \text{ powr } (3/4) * \ln k - \text{real } k \text{ powr } (7/8) / 4) \leq -1$
using *eventually-mono [OF eventually-all-ge-at-top [OF D0]]*
by (*smt (verit, ccfv-SIG) mult-left-mono of-nat-0-le-iff*)
show *?thesis*
using *assms*
unfolding *Big-Red-5-6-Ramsey-def eventually-conj-iff m-of-def*
by (*intro conjI eventually-all-ge-at-top D; real-asymp*)
qed

lemma *Red-5-6-Ramsey:*

assumes $0 < c < 1/32$ **and** $l \leq k$ **and** *big: Big-Red-5-6-Ramsey c l*
shows $\exp (c * l \text{ powr } (3/4) * \ln k) \leq \text{RN } k (\text{nat} [l \text{ powr } (3/4)])$

proof –

define r **where** $r \equiv \text{nat} [\exp (c * l \text{ powr } (3/4) * \ln k)]$

define s **where** $s \equiv \text{nat} [l \text{ powr } (3/4)]$

have $l \neq 0$

using *big* **by** (*force simp: Big-Red-5-6-Ramsey-def*)

have $3 \leq s$

using *assms* **by** (*auto simp: Big-Red-5-6-Ramsey-def s-def*)

also have $\dots \leq l$

using *powr-mono [of 3/4 1] <l ≠ 0>* **by** (*simp add: s-def*)

finally have $3 \leq l$.

then have $k \geq 3$ $\langle k > 0 \rangle$ $\langle l > 0 \rangle$

using *assms* **by** *auto*

define p **where** $p \equiv k \text{ powr } (-1/8)$

have $p01: 0 < p < 1$

using $\langle k \geq 3 \rangle$ *powr-less-one* **by** (*auto simp: p-def*)

have $r\text{-le}: r \leq k \text{ powr } (c * l \text{ powr } (3/4))$

using $p01$ $\langle k \geq 3 \rangle$ **unfolding** $r\text{-def}$ *powr-def* **by** *force*

have $\text{left}: r \wedge s * p \text{ powr } ((\text{real } s)^2 / 4) < 1/2$

proof –

have $A: r \text{ powr } s \leq k \text{ powr } (s * c * l \text{ powr } (3/4))$

using $r\text{-le}$ **by** (*smt (verit) mult.commute of-nat-0-le-iff powr-mono2 powr-powr*)

have $B: p \text{ powr } ((\text{real } s)^2 / 4) \leq k \text{ powr } (-(\text{real } s)^2 / 32)$

by (*simp add: powr-powr p-def power2-eq-square*)

have $C: (c * l \text{ powr } (3/4) - s/32) \leq -1$

using *big* **by** (*simp add: Big-Red-5-6-Ramsey-def s-def algebra-simps*) *linarith*

have $r \wedge s * p \text{ powr } ((\text{real } s)^2 / 4) \leq k \text{ powr } (s * (c * l \text{ powr } (3/4) - s / 32))$

using *mult-mono [OF A B] <s ≥ 3>*

by (*simp add: power2-eq-square algebra-simps powr-realpow' flip: powr-add*)

also have $\dots \leq k \text{ powr } -\text{real } s$

using C $\langle s \geq 3 \rangle$ *mult-left-mono <k ≥ 3>* **by** *fastforce*

also have $\dots \leq k \text{ powr } -3$

```

    using <k≥3> <s≥3> by (simp add: powr-minus powr-realpow)
  also have ... ≤ 3 powr -3
    using <k≥3> by (intro powr-mono2') auto
  also have ... < 1/2
    by auto
  finally show ?thesis .
qed
have right: r^k * exp (- p * (real k)^2 / 4) < 1/2
proof -
  have A: r^k ≤ exp (c * l powr (3/4) * ln k * k)
    using r-le <0 < k> <0 < l> by (simp add: powr-def exp-of-nat2-mult)
  have B: exp (- p * (real k)^2 / 4) ≤ exp (- k * k powr (7/8) / 4)
    using <k>0> by (simp add: p-def mult-ac power2-eq-square powr-mult-base)
  have r^k * exp (- p * (real k)^2 / 4) ≤ exp (k * (c * l powr (3/4) * ln k - k
powr (7/8) / 4))
    using mult-mono [OF A B] by (simp add: algebra-simps s-def flip: exp-add)
  also have ... ≤ exp (-1)
    using assms unfolding Big-Red-5-6-Ramsey-def by blast
  also have ... < 1/2
    by (approximation 5)
  finally show ?thesis .
qed
have ¬ is-Ramsey-number (nat[l powr (3/4)]) k (nat [exp (c * l powr (3/4) *
ln k)])
  using Ramsey-number-lower-simple [OF - p01] left right <k≥3> <l≥3>
  unfolding r-def s-def by force
then show ?thesis
  by (smt (verit) RN-commute is-Ramsey-number-RN le-nat-floor partn-lst-greater-resource)
qed

definition ineq-Red-5-6 ≡ λc l. ∀ k. l ≤ k → exp (c * real l powr (3/4) * ln k)
≤ RN k (nat[l powr (3/4)])

definition Big-Red-5-6 ≡
  λl. 6 + m-of l ≤ (1/128) * l powr (3/4) ∧ ineq-Red-5-6 (1/128) l

  establishing the size requirements for 5.6

lemma Big-Red-5-6: ∀∞l. Big-Red-5-6 l
proof -
  define c::real where c ≡ 1/128
  have 0 < c c < 1/32
    by (auto simp: c-def)
  then have ∀∞l. ineq-Red-5-6 c l
    unfolding ineq-Red-5-6-def using Red-5-6-Ramsey Big-Red-5-6-Ramsey exp-gt-zero
    by (smt (verit, del-Insts) eventually-sequentially)
  then show ?thesis
    unfolding Big-Red-5-6-def eventually-conj-iff m-of-def
    by (simp add: c-def; real-asymp)
qed

```

```

lemma (in Book) Red-5-6:
  assumes big: Big-Red-5-6 l
  shows  $RN\ k\ (nat\ [l\ powr\ (3/4)]) \geq k^6 * RN\ k\ (m\text{-of}\ l)$ 
proof –
  define c::real where  $c \equiv 1/128$ 
  have  $RN\ k\ (m\text{-of}\ l) \leq k^{(m\text{-of}\ l)}$ 
    by (metis RN-le-argpower' RN-mono diff-add-inverse diff-le-self le-refl le-trans)
  also have  $\dots \leq exp\ (m\text{-of}\ l * ln\ k)$ 
    using kn0 by (simp add: exp-of-nat-mult)
  finally have  $RN\ k\ (m\text{-of}\ l) \leq exp\ (m\text{-of}\ l * ln\ k)$ 
    by force
  then have  $k^6 * RN\ k\ (m\text{-of}\ l) \leq real\ k^6 * exp\ (m\text{-of}\ l * ln\ k)$ 
    by (simp add: kn0)
  also have  $\dots \leq exp\ (c * l\ powr\ (3/4) * ln\ k)$ 
proof –
  have  $(6 + real\ (m\text{-of}\ l)) * ln\ (real\ k) \leq (c * l\ powr\ (3/4)) * ln\ (real\ k)$ 
    unfolding mult-le-cancel-right
    using big kn0 by (auto simp: c-def Big-Red-5-6-def)
  then have  $ln\ (real\ k^6 * exp\ (m\text{-of}\ l * ln\ k)) \leq ln\ (exp\ (c * l\ powr\ (3/4) * ln\ k))$ 
    using kn0 by (simp add: ln-mult ln-powr algebra-simps flip: powr-numeral)
  then show ?thesis
    by (smt (verit) exp-gt-zero ln-le-cancel-iff)
qed
  also have  $\dots \leq RN\ k\ (nat\ [l\ powr\ (3/4)])$ 
    using assms l-le-k by (auto simp: ineq-Red-5-6-def Big-Red-5-6-def c-def)
  finally show  $k^6 * RN\ k\ (m\text{-of}\ l) \leq RN\ k\ (nat\ [l\ powr\ (3/4)])$ 
    using of-nat-le-iff by blast
qed

```

4.2 Lemma 5.4

definition *Big-Red-5-4* $\equiv \lambda l. Big-Red-5-6\ l \wedge (\forall k \geq l. real\ k + 2 * real\ k^6 \leq real\ k^7)$

establishing the size requirements for 5.4

```

lemma Big-Red-5-4:  $\forall^\infty l. Big-Red-5-4\ l$ 
  unfolding Big-Red-5-4-def eventually-conj-iff all-imp-conj-distrib
  apply (simp add: Big-Red-5-6)
  apply (intro conjI eventually-all-ge-at-top; real-asymp)
  done

```

context *Book*
begin

```

lemma Red-5-4:
  assumes i:  $i \in Step\text{-class}\ \{red\text{-step}, dboost\text{-step}\}$ 
  and big: Big-Red-5-4 l
  defines  $X \equiv Xseq\ i$  and  $Y \equiv Yseq\ i$ 

```

shows $\text{weight } X \ Y \ (\text{cvx } i) \geq - \text{card } X / (\text{real } k) \wedge 5$
proof –
have $l \neq 1$
using *big* **by** (*auto simp: Big-Red-5-4-def*)
with $ln0 \ l \leq k$ **have** $l > 1 \ k > 1$ **by** *linarith+*
let $?R = RN \ k \ (m \text{-of } l)$
have *finite* $X \ \text{finite } Y$
by (*auto simp: X-def Y-def finite-Xseq finite-Yseq*)
have *not-many-bluish*: $\neg \text{many-bluish } X$
using *i not-many-bluish unfolding X-def* **by** *blast*
have *nonterm*: $\neg \text{termination-condition } X \ Y$
using *X-def Y-def i step-non-terminating-iff* **by** (*force simp: Step-class-def*)
moreover **have** $l \text{ powr } (2/3) \leq l \text{ powr } (3/4)$
using $\langle l > 1 \rangle$ **by** (*simp add: powr-mono*)
ultimately **have** $RNX: ?R < \text{card } X$
unfolding *termination-condition-def m-of-def*
by (*meson RN-mono order.trans ceiling-mono le-refl nat-mono not-le*)
have $0 \leq (\sum x \in X. \sum x' \in X. \text{Weight } X \ Y \ x \ x')$
by (*simp add: X-def Y-def sum-Weight-ge0 Xseq-subset-V Yseq-subset-V Xseq-Yseq-disjnt*)
also **have** $\dots = (\sum y \in X. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y)$
unfolding *weight-def X-def*
by (*smt (verit) sum.cong sum.infinite sum.remove*)
finally **have** $ge0: 0 \leq (\sum y \in X. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y)$.
have *w-maximal*: $\text{weight } X \ Y \ (\text{cvx } i) \geq \text{weight } X \ Y \ x$
if *central-vertex* $X \ x$ **for** x
using *X-def Y-def <finite X> central-vx-is-best cvx-works i that* **by** *presburger*

have $|\text{real } (\text{card } (S \cap Y)) * (\text{real } (\text{card } X) * \text{real } (\text{card } Y)) -$
 $\text{real } (\text{edge-card } Red \ X \ Y) * \text{real } (\text{card } (T \cap Y))|$
 $\leq \text{real } (\text{card } X) * \text{real } (\text{card } Y) * \text{real } (\text{card } Y)$ **for** $S \ T$
using *card-mono [OF - Int-lower2] <finite X> <finite Y>*
by (*smt (verit, best) of-nat-mult edge-card-le mult.commute mult-right-mono*
of-nat-0-le-iff of-nat-mono)
then **have** $W1abs: |\text{Weight } X \ Y \ x \ y| \leq 1$ **for** $x \ y$
using *RNX edge-card-le [of X Y Red] <finite X> <finite Y>*
apply (*simp add: mult-ac Weight-def divide-simps gen-density-def*)
by (*metis Int-lower2 card-mono mult-of-nat-commute*)
then **have** $W1: \text{Weight } X \ Y \ x \ y \leq 1$ **for** $x \ y$
by (*smt (verit)*)
have *WW-le-cardX*: $\text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y \leq \text{card } X$ **if** $y \in X$ **for** y
proof –
have $\text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y = \text{sum } (\text{Weight } X \ Y \ y) \ X$
by (*simp add: <finite X> sum-diff1 that weight-def*)
also **have** $\dots \leq \text{card } X$
using $W1$ **by** (*smt (verit) real-of-card sum-mono*)
finally **show** *?thesis* .
qed
have $\text{weight } X \ Y \ x \leq \text{real } (\text{card}(X \setminus \{x\})) * 1$ **for** x
unfolding *weight-def* **by** (*meson DiffE abs-le-D1 sum-bounded-above W1*)

then have *wgt-le-X1*: $\text{weight } X \ Y \ x \leq \text{card } X - 1$ **if** $x \in X$ **for** x
using *that card-Diff-singleton One-nat-def* **by** (*smt (verit, best)*)
define *XB* **where** $XB \equiv \{x \in X. \text{bluish } X \ x\}$
have *card-XB*: $\text{card } XB < ?R$
using *not-many-bluish* **by** (*auto simp: m-of-def many-bluish-def XB-def*)
have $XB \subseteq X$ *finite XB*
using $\langle \text{finite } X \rangle$ **by** (*auto simp: XB-def*)
then have *cv-non-XB*: $\bigwedge y. y \in X \setminus XB \implies \text{central-vertex } X \ y$
by (*auto simp: central-vertex-def XB-def bluish-def*)
have $0 \leq (\sum y \in X. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y)$
by (*fact ge0*)
also have $\dots = (\sum y \in XB. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y) + (\sum y \in X \setminus XB. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y)$
using *sum.subset-diff [OF <XB⊆X>]* **by** (*smt (verit) X-def Xseq-subset-V finV finite-subset*)
also have $\dots \leq (\sum y \in XB. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y) + (\sum y \in X \setminus XB. \text{weight } X \ Y \ (cvx \ i) + 1)$
by (*intro add-mono sum-mono w-maximal W1 order-refl cv-non-XB*)
also have $\dots = (\sum y \in XB. \text{weight } X \ Y \ y + \text{Weight } X \ Y \ y \ y) + (\text{card } X - \text{card } XB) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
using $\langle XB \subseteq X \rangle \langle \text{finite } XB \rangle$ **by** (*simp add: card-Diff-subset*)
also have $\dots \leq \text{card } XB * \text{card } X + (\text{card } X - \text{card } XB) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
using *sum-bounded-above WW-le-cardX*
by (*smt (verit, cfv-threshold) XB-def mem-Collect-eq of-nat-mult*)
also have $\dots = \text{real } (?R * \text{card } X) + (\text{real } (\text{card } XB) - ?R) * \text{card } X + (\text{card } X - \text{card } XB) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
using *card-XB* **by** (*simp add: algebra-simps flip: of-nat-mult of-nat-diff*)
also have $\dots \leq \text{real } (?R * \text{card } X) + (\text{card } X - ?R) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
proof –
have $(\text{real } (\text{card } X) - \text{card } XB) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
 $\leq (\text{real } (\text{card } X) - ?R) * (\text{weight } X \ Y \ (cvx \ i) + 1) + (\text{real } (?R) - \text{card } XB) * (\text{weight } X \ Y \ (cvx \ i) + 1)$
by (*simp add: algebra-simps*)
also have $\dots \leq (\text{real } (\text{card } X) - ?R) * (\text{weight } X \ Y \ (cvx \ i) + 1) + (\text{real } (?R) - \text{card } XB) * \text{card } X$
using *RNX X-def i card-XB cvx-in-Xseq wgt-le-X1* **by** *fastforce*
finally show *?thesis*
by (*smt (verit, del-insts) RNX <XB ⊆ X> <finite X> card-mono nat-less-le of-nat-diff distrib-right*)
qed
finally have *weight-ge-0*: $0 \leq ?R * \text{card } X + (\text{card } X - ?R) * (\text{weight } X \ Y \ (cvx \ i) + 1)$.
have *rk61*: $\text{real } k^6 > 1$
using $\langle k > 1 \rangle$ **by** *simp*
have *k267*: $\text{real } k + 2 * \text{real } k^6 \leq (\text{real } k^7)$
using $\langle l \leq k \rangle$ **big by** (*auto simp: Big-Red-5-4-def*)
have *k-le*: $\text{real } k^6 + (?R * \text{real } k + ?R * (\text{real } k^6)) \leq 1 + ?R * (\text{real } k^7)$

using *mult-left-mono* [*OF k267, of ?R*] *assms*
by (*smt (verit, ccfv-SIG) distrib-left card-XB mult-le-cancel-right1 nat-less-real-le of-nat-0-le-iff zero-le-power*)
have [*simp*]: $\text{real } k^m = \text{real } k^n \iff m=n \text{ real } k^m < \text{real } k^n \iff m < n$ **for**
 $m\ n$
using $\langle 1 < k \rangle$ **by** *auto*
have *RN k (nat[l powr (3/4)])* $\geq k^6 * ?R$
using $\langle l \leq k \rangle$ *big Red-5-6* **by** (*auto simp: Big-Red-5-4-def*)
then have *cardX-ge*: $\text{card } X \geq k^6 * ?R$
by (*meson le-trans nat-le-linear nonterm termination-condition-def*)
have $-1 / (\text{real } k)^5 \leq -1 / (\text{real } k^6 - 1) + -1 / (\text{real } k^6 * ?R)$
using *rk61 card-XB mult-left-mono [OF k-le, of real k^5]*
by (*simp add: field-split-simps eval-nat-numeral*)
also have $\dots \leq - ?R / (\text{real } k^6 * ?R - ?R) + -1 / (\text{real } k^6 * ?R)$
using *card-XB rk61* **by** (*simp add: field-split-simps*)
finally have $-1 / (\text{real } k)^5 \leq - ?R / (\text{real } k^6 * ?R - ?R) + -1 / (\text{real } k^6$
 $* ?R)$.
also have $\dots \leq - ?R / (\text{real } (\text{card } X) - ?R) + -1 / \text{card } X$
proof (*intro add-mono divide-left-mono-neg*)
show $\text{real } k^6 * \text{real } ?R - \text{real } ?R \leq \text{real } (\text{card } X) - \text{real } ?R$
using *cardX-ge of-nat-mono* **by** *fastforce*
show $\text{real } k^6 * \text{real } ?R \leq \text{real } (\text{card } X)$
using *cardX-ge of-nat-mono* **by** *fastforce*
qed (*use RNX rk61 kn0 card-XB in auto*)
also have $\dots \leq \text{weight } X\ Y\ (\text{cvx } i) / \text{card } X$
using *RNX mult-left-mono [OF weight-ge-0, of card X]* **by** (*simp add: field-split-simps*)
finally show *?thesis*
using *RNX* **by** (*simp add: X-def Y-def divide-simps*)
qed

lemma *Red-5-7a*: $\varepsilon / k \leq \text{alpha } (\text{hgt } p)$
by (*simp add: alpha-ge hgt-gt0*)

lemma *Red-5-7b*:

assumes $p \geq \text{qfun } 0$ **shows** $\text{alpha } (\text{hgt } p) \leq \varepsilon * (p - \text{qfun } 0 + 1/k)$
proof –
have *qh-le-p*: $\text{qfun } (\text{hgt } p - \text{Suc } 0) \leq p$
by (*smt (verit) assms diff-Suc-less hgt-gt0 hgt-less-imp-qfun-less zero-less-iff-neq-zero*)
have $\text{alpha } (\text{hgt } p) = \varepsilon * (1 + \varepsilon)^{\text{hgt } p - 1} / k$
using *alpha-eq alpha-hgt-eq* **by** *blast*
also have $\dots = \varepsilon * (\text{qfun } (\text{hgt } p - 1) - \text{qfun } 0 + 1/k)$
by (*simp add: diff-divide-distrib qfun-eq*)
also have $\dots \leq \varepsilon * (p - \text{qfun } 0 + 1/k)$
by (*simp add: eps-ge0 mult-left-mono qh-le-p*)
finally show *?thesis* .
qed

lemma *Red-5-7c*:

assumes $p \leq \text{qfun } 1$ **shows** $\text{alpha } (\text{hgt } p) = \varepsilon / k$

using *alpha-hgt-eq Book-axioms assms hgt-Least* by *fastforce*

lemma *Red-5-8*:

assumes *i*: $i \in \text{Step-class } \{\text{dreg-step}\}$ **and** *x*: $x \in X\text{seq } (\text{Suc } i)$
shows $\text{card } (\text{Neighbours Red } x \cap Y\text{seq } (\text{Suc } i))$
 $\geq (1 - \varepsilon \text{ powr } (1/2)) * \text{pseq } i * (\text{card } (Y\text{seq } (\text{Suc } i)))$

proof –

obtain *X Y A B*

where *step*: $\text{stepper } i = (X, Y, A, B)$

and *nonterm*: $\neg \text{termination-condition } X Y$

and *even* *i*

and *Suc-i*: $\text{stepper } (\text{Suc } i) = \text{degree-reg } (X, Y, A, B)$

and *XY*: $X = X\text{seq } i \ Y = Y\text{seq } i$

using *i* by (*auto simp: step-kind-defs split: if-split-asm prod.split-asm*)

have $X\text{seq } (\text{Suc } i) = ((\lambda(X, Y, A, B). X) \circ \text{stepper}) (\text{Suc } i)$

by (*simp add: Xseq-def*)

also have $\dots = X\text{-degree-reg } X Y$

using *<even i> step nonterm* by (*auto simp: degree-reg-def*)

finally have $X\text{Suc}: X\text{seq } (\text{Suc } i) = X\text{-degree-reg } X Y$.

have $Y\text{Suc}: Y\text{seq } (\text{Suc } i) = Y\text{seq } i$

using *Suc-i step* by (*auto simp: degree-reg-def stepper-XYseq*)

have *p-gt-invok*: $(\text{pseq } i) > 1/k$

using *XY nonterm pseq-def termination-condition-def* by *auto*

have $\text{RedN}: (\text{pseq } i - \varepsilon \text{ powr } -(1/2) * \alpha (\text{hgt } (\text{pseq } i))) * \text{card } Y \leq \text{card } (\text{Neighbours Red } x \cap Y)$

using *x XY* by (*simp add: XSuc YSuc X-degree-reg-def pseq-def red-dense-def*)

show *?thesis*

proof (*cases pseq i ≥ qfun 0*)

case *True*

have $i \notin \text{Step-class } \{\text{halted}\}$

using *i* by (*simp add: Step-class-def*)

then have *p0*: $1/k < p0$

by (*metis Step-class-not-halted gr0I nat-less-le not-halted-pee-gt pee-eq-p0*)

have *0*: $\varepsilon \text{ powr } -(1/2) \geq 0$

by *simp*

have $\varepsilon \text{ powr } -(1/2) * \alpha (\text{hgt } (\text{pseq } i)) \leq \varepsilon \text{ powr } (1/2) * ((\text{pseq } i) - \text{qfun } 0 + 1/k)$

using *mult-left-mono [OF Red-5-7b [OF True] 0]*

by (*simp add: eps-def powr-mult-base flip: mult-ac*)

also have $\dots \leq \varepsilon \text{ powr } (1/2) * (\text{pseq } i)$

using *p0* by (*intro mult-left-mono*) (*auto simp flip: pee-eq-p0*)

finally have $\varepsilon \text{ powr } -(1/2) * \alpha (\text{hgt } (\text{pseq } i)) \leq \varepsilon \text{ powr } (1/2) * (\text{pseq } i)$.

then have $(1 - \varepsilon \text{ powr } (1/2)) * (\text{pseq } i) * (\text{card } Y) \leq ((\text{pseq } i) - \varepsilon \text{ powr } -(1/2) * \alpha (\text{hgt } (\text{pseq } i))) * \text{card } Y$

by (*intro mult-right-mono*) (*auto simp: algebra-simps*)

with *XY RedN YSuc* **show** *?thesis* by *fastforce*

next

case *False*

then have $\text{pseq } i \leq \text{qfun } 1$

by (smt (verit) One-nat-def alpha-Suc-eq alpha-ge0 q-Suc-diff)
 then have $\varepsilon \text{ powr } -(1/2) * \text{alpha } (\text{hgt } (\text{pseq } i)) = \varepsilon \text{ powr } (1/2) / k$
 using powr-mult-base [of ε] eps-gt0 by (force simp: Red-5-7c mult.commute)
 also have $\dots \leq \varepsilon \text{ powr } (1/2) * (\text{pseq } i)$
 using p-gt-invok
 by (smt (verit) divide-inverse inverse-eq-divide mult-left-mono powr-ge-zero)
 finally have $\varepsilon \text{ powr } -(1/2) * \text{alpha } (\text{hgt } (\text{pseq } i)) \leq \varepsilon \text{ powr } (1/2) * (\text{pseq } i)$.
 then have $(1 - \varepsilon \text{ powr } (1/2)) * \text{pseq } i * \text{card } Y \leq (\text{pseq } i - \varepsilon \text{ powr } -(1/2))$
 * $\text{alpha } (\text{hgt } (\text{pseq } i)) * \text{card } Y$
 by (intro mult-right-mono) (auto simp: algebra-simps)
 with XY RedN YSuc show ?thesis by fastforce
 qed
 qed

corollary *Y-Neighbours-nonempty-Suc*:

assumes $i: i \in \text{Step-class } \{\text{dreg-step}\}$ and $x: x \in X\text{seq } (\text{Suc } i)$ and $k \geq 2$
 shows $\text{Neighbours Red } x \cap Y\text{seq } (\text{Suc } i) \neq \{\}$

proof

assume con: $\text{Neighbours Red } x \cap Y\text{seq } (\text{Suc } i) = \{\}$
 have not-halted: $i \notin \text{Step-class } \{\text{halted}\}$
 using i by (auto simp: Step-class-def)
 then have 0: $\text{pseq } i > 0$
 using not-halted-pee-gt0 by blast
 have Y': $\text{card } (Y\text{seq } (\text{Suc } i)) > 0$
 using i Yseq-gt0 [OF not-halted] stepper-XYseq
 by (auto simp: step-kind-defs degree-reg-def split: if-split-asm prod.split-asm)
 have $(1 - \varepsilon \text{ powr } (1/2)) * \text{pseq } i * \text{card } (Y\text{seq } (\text{Suc } i)) \leq 0$
 using Red-5-8 [OF i x] con by simp
 with 0 Y' have $(1 - \varepsilon \text{ powr } (1/2)) \leq 0$
 by (simp add: mult-le-0-iff zero-le-mult-iff)
 then show False
 using $\langle k \geq 2 \rangle$ powr-le-cancel-iff [of k $1/8$ 0]
 by (simp add: eps-def powr-minus-divide powr-divide powr-powr)

qed

corollary *Y-Neighbours-nonempty*:

assumes $i: i \in \text{Step-class } \{\text{red-step}, \text{dboost-step}\}$ and $x: x \in X\text{seq } i$ and $k \geq 2$
 shows $\text{card } (\text{Neighbours Red } x \cap Y\text{seq } i) > 0$

proof (cases i)

case 0

with assms show ?thesis

by (auto simp: Step-class-def stepper-kind-def split: if-split-asm)

next

case ($\text{Suc } i'$)

then have $i' \in \text{Step-class } \{\text{dreg-step}\}$

by (metis dreg-before-step dreg-before-step i Step-class-insert Un-iff)

then have $\text{Neighbours Red } x \cap Y\text{seq } (\text{Suc } i') \neq \{\}$

using Suc *Y-Neighbours-nonempty-Suc* assms by blast

then show ?thesis

by (*simp add: Suc card-gt-0-iff finite-Neighbours*)
qed

end

4.3 Lemma 5.1

definition *Big-Red-5-1* $\equiv \lambda \mu l. (1-\mu) * \text{real } l > 1 \wedge l \text{ powr } (5/2) \geq 3 / (1-\mu)$
 $\wedge l \text{ powr } (1/4) \geq 4$
 $\wedge \text{Big-Red-5-4 } l \wedge \text{Big-Red-5-6 } l$

establishing the size requirements for 5.1

lemma *Big-Red-5-1*:

assumes $\mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-Red-5-1 } \mu l$

proof –

have $(\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 1 < (1-\mu) * \text{real } l)$

proof (*intro eventually-all-geI1*)

show $\bigwedge l \mu. \llbracket 1 < (1-\mu 1) * \text{real } l; \mu \leq \mu 1 \rrbracket \Longrightarrow 1 < (1-\mu) * l$

by (*smt (verit, best) mult-right-mono of-nat-0-le-iff*)

qed (*use assms in real-asymp*)

moreover have $(\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 3 / (1-\mu) \leq \text{real } l \text{ powr } (5/2))$

proof (*intro eventually-all-geI1*)

show $\bigwedge l \mu. \llbracket 3 / (1-\mu 1) \leq \text{real } l \text{ powr } (5/2); \mu \leq \mu 1 \rrbracket$

$\Longrightarrow 3 / (1-\mu) \leq \text{real } l \text{ powr } (5/2)$

by (*smt (verit, ccfv-SIG) assms frac-le*)

qed (*use assms in real-asymp*)

moreover have $\forall^\infty l. 4 \leq \text{real } l \text{ powr } (1 / 4)$

by *real-asymp*

ultimately show *?thesis*

using *assms Big-Red-5-6 Big-Red-5-4* **by** (*auto simp: Big-Red-5-1-def all-imp-conj-distrib eventually-conj-iff*)

qed

context *Book*

begin

lemma *card-cvx-Neighbours*:

assumes $i: i \in \text{Step-class } \{\text{red-step, dboost-step}\}$

defines $x \equiv \text{cvx } i$

defines $X \equiv X \text{seq } i$

defines $\text{NBX} \equiv \text{Neighbours Blue } x \cap X$

defines $\text{NRX} \equiv \text{Neighbours Red } x \cap X$

shows $\text{card NBX} \leq \mu * \text{card } X \text{ card NRX} \geq (1-\mu) * \text{card } X - 1$

proof –

obtain $x \in X \ X \subseteq V$

by (*metis Xseq-subset-V cvx-in-Xseq X-def i x-def*)

then have *card-NRBX*: $\text{card NRX} + \text{card NBX} = \text{card } X - 1$

using *Neighbours-RB [of x X] disjnt-Red-Blue-Neighbours*

by (simp add: NRX-def NBX-def finite-Neighbours subsetD flip: card-Un-disjnt)
 moreover have card-NBX-le: $\text{card NBX} \leq \mu * \text{card X}$
 by (metis cvx-works NBX-def X-def central-vertex-def i x-def)
 ultimately show $\text{card NBX} \leq \mu * \text{card X}$ $\text{card NRX} \geq (1-\mu) * \text{card X} - 1$
 by (auto simp: algebra-simps)
 qed

lemma Red-5-1:

assumes $i: i \in \text{Step-class } \{\text{red-step, dboost-step}\}$
 and Big: Big-Red-5-1 μ l
 defines $p \equiv \text{pseq } i$
 defines $x \equiv \text{cvx } i$
 defines $X \equiv \text{Xseq } i$ and $Y \equiv \text{Yseq } i$
 defines $\text{NBX} \equiv \text{Neighbours Blue } x \cap X$
 defines $\text{NRX} \equiv \text{Neighbours Red } x \cap X$
 defines $\text{NRY} \equiv \text{Neighbours Red } x \cap Y$
 defines $\beta \equiv \text{card NBX} / \text{card X}$
 shows $\text{red-density NRX NRY} \geq p - \text{alpha } (\text{hgt } p)$
 $\vee \text{red-density NBX NRY} \geq p + (1 - \varepsilon) * ((1-\beta) / \beta) * \text{alpha } (\text{hgt } p) \wedge \beta$
 > 0
 proof –
 have Red-5-4: $\text{weight X Y } x \geq - \text{real } (\text{card X}) / (\text{real } k)^5$
 using Big i Red-5-4 by (auto simp: Big-Red-5-1-def x-def X-def Y-def)
 have lA: $(1-\mu) * l > 1$ and $l \leq k$ and l144: $l \text{ powr } (1/4) \geq 4$
 using Big by (auto simp: Big-Red-5-1-def l-le-k)
 then have k-powr-14: $k \text{ powr } (1/4) \geq 4$
 by (smt (verit) divide-nonneg-nonneg of-nat-0-le-iff of-nat-mono powr-mono2)
 have $k \geq 256$
 using powr-mono2 [of 4, OF - - k-powr-14] by (simp add: powr-powr flip:
 powr-numeral)
 then have $k > 0$ by linarith
 have k52: $3 / (1-\mu) \leq k \text{ powr } (5/2)$
 using Big $\langle l \leq k \rangle$ unfolding Big-Red-5-1-def
 by (smt (verit) of-nat-0-le-iff of-nat-mono powr-mono2 zero-le-divide-iff)
 have RN-le-RN: $k^6 * \text{RN } k (m\text{-of } l) \leq \text{RN } k (\text{nat } \lceil l \text{ powr } (3/4) \rceil)$
 using Big $\langle l \leq k \rangle$ Red-5-6 by (auto simp: Big-Red-5-1-def)
 have l34-ge3: $l \text{ powr } (3/4) \geq 3$
 by (smt (verit, ccfv-SIG) l144 divide-nonneg-nonneg frac-le of-nat-0-le-iff powr-le1
 powr-less-cancel)
 note $XY = X\text{-def } Y\text{-def}$
 obtain $A B$
 where $\text{step: stepper } i = (X, Y, A, B)$
 and $\text{nonterm: } \neg \text{termination-condition X Y}$
 and $\text{odd } i$
 and $\text{non-mb: } \neg \text{many-bluish X}$ and $\text{card X} > 0$
 and $\text{not-halted: } i \notin \text{Step-class } \{\text{halted}\}$
 using i by (auto simp: XY step-kind-defs termination-condition-def split:
 if-split-asm prod.split-asm)
 with $Y\text{seq-gt0 XY}$ have $\text{card Y} \neq 0$

by *blast*
have *cX-RN*: $\text{card } X > \text{RN } k \text{ (nat } \lceil l \text{ powr } (3/4) \rceil)$
by (*meson linorder-not-le nonterm termination-condition-def*)
then have *X-gt-k*: $\text{card } X > k$
by (*metis l34-ge3 RN-3plus' of-nat-numeral order.trans le-natceiling-iff not-less*)
have $0 < \text{RN } k \text{ (m-of } l)$
using *RN-eq-0-iff m-of-def many-bluish-def non-mb* **by** *presburger*
then have $k^4 \leq k^6 * \text{RN } k \text{ (m-of } l)$
by (*simp add: eval-nat-numeral*)
also have $\dots < \text{card } X$
using *cX-RN RN-le-RN* **by** *linarith*
finally have $\text{card } X > k^4$.
have $x \in X$
using *cvx-in-Xseq i XY x-def* **by** *blast*
have $X \subseteq V$
by (*simp add: Xseq-subset-V XY*)
have *finite NRX finite NBX finite NRY*
by (*auto simp: NRX-def NBX-def NRY-def finite-Neighbours*)
have *disjnt X Y*
using *Xseq-Yseq-disjnt step stepper-XYseq* **by** *blast*
then have *disjnt NRX NRY disjnt NBX NRY*
by (*auto simp: NRX-def NBX-def NRY-def disjnt-iff*)
have *card-NRBX*: $\text{card } \text{NRX} + \text{card } \text{NBX} = \text{card } X - 1$
using *Neighbours-RB [of x X] <finite NRX> <x∈X> <X⊆V> disjnt-Red-Blue-Neighbours*
by (*simp add: NRX-def NBX-def finite-Neighbours subsetD flip: card-Un-disjnt*)
obtain *card-NBX-le*: $\text{card } \text{NBX} \leq \mu * \text{card } X$ **and** $\text{card } \text{NRX} \geq (1 - \mu) * \text{card } X - 1$
unfolding *NBX-def NRX-def X-def x-def* **using** *card-cvx-Neighbours i* **by** *metis*
with *lA <l≤k> X-gt-k* **have** $\text{card } \text{NRX} > 0$
by (*smt (verit, best) of-nat-0 μ01 gr0I mult-less-cancel-left-pos nat-less-real-le of-nat-mono*)
have $\text{card } \text{NRY} > 0$
using *Y-Neighbours-nonempty [OF i] <k≥256> NRY-def <finite NRY> <x ∈ X> card-0-eq XY* **by** *force*
show *?thesis*
proof (*cases* ($\sum y \in \text{NRX}. \text{Weight } X \ Y \ x \ y$) $\geq -\text{alpha} \text{ (hgt } p) * \text{card } \text{NRX} * \text{card } \text{NRY} / \text{card } Y$)
case *True*
then have $(p - \text{alpha} \text{ (hgt } p)) * (\text{card } \text{NRX} * \text{card } \text{NRY}) \leq (\sum y \in \text{NRX}. p * \text{card } \text{NRY} + \text{Weight } X \ Y \ x \ y * \text{card } Y)$
using $\langle \text{card } Y \neq 0 \rangle$ **by** (*simp add: field-simps sum-distrib-left sum.distrib*)
also have $\dots = (\sum y \in \text{NRX}. \text{card} (\text{Neighbours Red } x \cap \text{Neighbours Red } y \cap Y))$
using $\langle \text{card } Y \neq 0 \rangle$ **by** (*simp add: Weight-def pseq-def XY NRY-def field-simps p-def*)
also have $\dots = \text{edge-card Red NRY NRX}$
using $\langle \text{disjnt NRX NRY} \rangle \langle \text{finite NRX} \rangle$
by (*simp add: disjnt-sym edge-card-eq-sum-Neighbours Red-E psubset-imp-subset NRY-def Int-ac*)

also have $\dots = \text{edge-card Red NRX NRY}$
by (*simp add: edge-card-commute*)
finally have $(p - \alpha (\text{hgt } p)) * \text{real} (\text{card NRX} * \text{card NRY}) \leq \text{real}$
(edge-card Red NRX NRY) .
then show *?thesis*
using $\langle \text{card NRX} \rangle > 0 \rangle \langle \text{card NRY} \rangle > 0 \rangle$
by (*simp add: NRX-def NRY-def gen-density-def field-split-simps XY*)
next
case *False*
have $x \in X$
unfolding *x-def* **using** *cvx-in-Xseq i XY* **by** *blast*
with *Neighbours-RB[of x X]* **have** $Xx: X - \{x\} = \text{NBX} \cup \text{NRX}$
using *Xseq-subset-V NRX-def NBX-def XY* **by** *blast*
have *disjnt: NBX \cap NRX = {}*
by (*auto simp: Blue-eq NRX-def NBX-def disjoint-iff in-Neighbours-iff*)
then have $\text{weight } X \ Y \ x = (\sum y \in \text{NRX}. \text{Weight } X \ Y \ x \ y) + (\sum y \in \text{NBX}.$
Weight X Y x y)
by (*simp add: weight-def Xx sum.union-disjoint finite-Neighbours NRX-def*
NBX-def)
with *False*
have *15: $(\sum y \in \text{NBX}. \text{Weight } X \ Y \ x \ y)$*
 $\geq \text{weight } X \ Y \ x + \alpha (\text{hgt } p) * \text{card NRX} * \text{card NRY} / \text{card } Y$
by *linarith*
have *pm1: pseq (i-1) > 1/k*
by (*meson Step-class-not-halted diff-le-self not-halted not-halted-pee-gt*)
have *β -eq: $\beta = \text{card NBX} / \text{card } X$*
using *NBX-def β -def XY* **by** *blast*
have $\beta \leq \mu$
by (*simp add: β -eq $\langle 0 < \text{card } X \rangle \text{card-NBX-le pos-divide-le-eq}$*)
have *im1: $i-1 \in \text{Step-class } \{\text{dreg-step}\}$*
using *i $\langle \text{odd } i \rangle \text{dreg-before-step}$*
by (*metis Step-class-insert Un-iff One-nat-def odd-Suc-minus-one*)
have $\varepsilon \leq 1/4$
using $\langle k \rangle > 0 \rangle k\text{-powr-14}$ **by** (*simp add: eps-def powr-minus-divide*)
then have $\varepsilon \text{ powr } (1/2) \leq (1/4) \text{ powr } (1/2)$
by (*simp add: eps-def powr-mono2*)
then have $A: 1/2 \leq 1 - \varepsilon \text{ powr } (1/2)$
by (*simp add: powr-divide*)
have $le: 1 / (2 * \text{real } k) \leq (1 - \varepsilon \text{ powr } (1/2)) * \text{pseq } (i-1)$
using *pm1 $\langle k \rangle > 0 \rangle \text{mult-mono [OF A less-imp-le [OF pm1]] A}$* **by** *simp*
have $\text{card } Y / (2 * \text{real } k) \leq (1 - \varepsilon \text{ powr } (1/2)) * \text{pseq } (i-1) * \text{card } Y$
using *mult-left-mono [OF le]* **by** (*metis mult.commute divide-inverse inverse-eq-divide*
of-nat-0-le-iff)
also have $\dots \leq \text{card NRY}$
using *pm1 Red-5-8 im1* **by** (*metis NRY-def One-nat-def $\langle \text{odd } i \rangle \langle x \in X \rangle$*
XY odd-Suc-minus-one)
finally have $Y\text{-NRY}: \text{card } Y / (2 * \text{real } k) \leq \text{card NRY} .$
have $\text{NBX} \neq \{\}$
proof

```

assume empty:  $NBX = \{\}$ 
then have  $cNRX$ :  $\text{card } NRX = \text{card } X - 1$ 
  using card-NRBX by auto
have  $\text{card } X > 3$ 
  using  $\langle k \geq 256 \rangle$  X-gt-k by linarith
then have  $2 * \text{card } X / \text{real } (\text{card } X - 1) < 3$ 
  by (simp add: divide-simps)
also have  $\dots \leq k^2$ 
  using mult-mono [OF  $\langle k \geq 256 \rangle$   $\langle k \geq 256 \rangle$ ] by (simp add: power2-eq-square)
flip: of-nat-mult
also have  $\dots \leq \varepsilon * k^3$ 
  using  $\langle k \geq 256 \rangle$  by (simp add: eps-def flip: powr-numeral powr-add)
finally have  $(\text{real } (2 * \text{card } X) / \text{real } (\text{card } X - 1)) * k^2 < \varepsilon * \text{real } (k^3)$ 
 $* k^2$ 
  using  $\langle k > 0 \rangle$  by (intro mult-strict-right-mono) auto
then have  $\text{real } (2 * \text{card } X) / \text{real } (\text{card } X - 1) * k^2 < \varepsilon * \text{real } (k^5)$ 
  by (simp add: mult.assoc flip: of-nat-mult)
then have  $0 < - \text{real } (\text{card } X) / (\text{real } k)^5 + (\varepsilon / k) * \text{real } (\text{card } X - 1)$ 
 $* (1 / (2 * \text{real } k))$ 
  using  $\langle k > 0 \rangle$  X-gt-k by (simp add: field-simps power2-eq-square)
also have  $- \text{real } (\text{card } X) / (\text{real } k)^5 + (\varepsilon / k) * \text{real } (\text{card } X - 1) * (1$ 
 $/ (2 * \text{real } k))$ 
   $\leq - \text{real } (\text{card } X) / (\text{real } k)^5 + (\varepsilon / k) * \text{real } (\text{card } NRX) * (\text{card}$ 
 $NRX / \text{card } Y)$ 
  using Y-NRY  $\langle k > 0 \rangle$   $\langle \text{card } Y \neq 0 \rangle$ 
  by (intro add-mono mult-mono) (auto simp: cNRX eps-def divide-simps)
also have  $\dots = - \text{real } (\text{card } X) / (\text{real } k)^5 + (\varepsilon / k) * \text{real } (\text{card } NRX)$ 
 $* \text{card } NRX / \text{card } Y$ 
  by simp
also have  $\dots \leq - \text{real } (\text{card } X) / (\text{real } k)^5 + \text{alpha } (\text{hgt } p) * \text{real } (\text{card}$ 
 $NRX) * \text{card } NRX / \text{card } Y$ 
  using alpha-ge [OF hgt-gt0]
  by (intro add-mono mult-right-mono divide-right-mono) auto
also have  $\dots \leq 0$ 
  using empty 15 Red-5-4 by auto
finally show False
  by simp
qed
have  $\text{card } NBX > 0$ 
  by (simp add:  $\langle NBX \neq \{\} \rangle$   $\langle \text{finite } NBX \rangle$  card-gt-0-iff)
then have  $0 < \beta$ 
  by (simp add: beta-eq  $\langle 0 < \text{card } X \rangle$ )
have  $\beta \leq \mu$ 
  using X-gt-k card-NBX-le by (simp add: beta-eq NBX-def divide-simps)
have  $cNRX$ :  $\text{card } NRX = (1 - \beta) * \text{card } X - 1$ 
  using X-gt-k card-NRBX by (simp add: beta-eq divide-simps)
have  $cNBX$ :  $\text{card } NBX = \beta * \text{card } X$ 
  using  $\langle 0 < \text{card } X \rangle$  by (simp add: beta-eq)
let  $?E16 = p + ((1 - \beta) / \beta) * \text{alpha } (\text{hgt } p) - \text{alpha } (\text{hgt } p) / (\beta * \text{card } X) +$ 

```

$weight\ X\ Y\ x * card\ Y / (\beta * card\ X * card\ NRY)$
have $p * card\ NBX * card\ NRY + alpha\ (hgt\ p) * card\ NRX * card\ NRY +$
 $weight\ X\ Y\ x * card\ Y$
 $\leq (\sum\ y \in NBX. p * card\ NRY + Weight\ X\ Y\ x\ y * card\ Y)$
using 15 $\langle card\ Y \neq 0 \rangle$ **apply** (*simp add: sum-distrib-left sum.distrib*)
by (*simp only: sum-distrib-right divide-simps split: if-split-asm*)
also have $\dots \leq (\sum\ y \in NBX. card\ (Neighbours\ Red\ x \cap Neighbours\ Red\ y \cap$
 $Y))$
using $\langle card\ Y \neq 0 \rangle$ **by** (*simp add: Weight-def pseq-def XY NRY-def field-simps*
p-def)
also have $\dots = edge-card\ Red\ NRY\ NBX$
using $\langle disjoint\ NBX\ NRY \rangle \langle finite\ NBX \rangle$
by (*simp add: disjoint-sym edge-card-eq-sum-Neighbours Red-E psubset-imp-subset*
NR Y-def Int-ac)
also have $\dots = edge-card\ Red\ NBX\ NRY$
by (*simp add: edge-card-commute*)
finally have *Red-bound*:
 $p * card\ NBX * card\ NRY + alpha\ (hgt\ p) * card\ NRX * card\ NRY + weight$
 $X\ Y\ x * card\ Y \leq edge-card\ Red\ NBX\ NRY .$
then have $(p * card\ NBX * card\ NRY + alpha\ (hgt\ p) * card\ NRX * card$
 $NR Y + weight\ X\ Y\ x * card\ Y)$
 $/ (card\ NBX * card\ NRY) \leq red-density\ NBX\ NRY$
by (*metis divide-le-cancel gen-density-def of-nat-less-0-iff*)
then have $p + alpha\ (hgt\ p) * card\ NRX / card\ NBX + weight\ X\ Y\ x * card$
 $Y / (card\ NBX * card\ NRY) \leq red-density\ NBX\ NRY$
using $\langle card\ NBX > 0 \rangle \langle card\ NRY > 0 \rangle$ **by** (*simp add: add-divide-distrib*)
then have 16: $?E16 \leq red-density\ NBX\ NRY$
using $\langle \beta > 0 \rangle \langle card\ X > 0 \rangle$
by (*simp add: cNRX cNBX algebra-simps add-divide-distrib diff-divide-distrib*)
consider $qfun\ 0 \leq p \mid p \leq qfun\ 1$
by (*smt (verit) alpha-Suc-eq alpha-ge0 One-nat-def q-Suc-diff*)
then have *alpha-le-1*: $alpha\ (hgt\ p) \leq 1$
proof cases
case 1
have $p * \epsilon + \epsilon / real\ k \leq 1 + \epsilon * p0$
proof (*intro add-mono*)
show $p * \epsilon \leq 1$
by (*simp add: assms(3) eps-ge0 eps-le1 mult-le-one pee-le1*)
have $p0 > 1/k$
by (*metis Step-class-not-halted diff-le-self not-halted not-halted-pee-gt*
diff-is-0-eq' pee-eq-p0)
then show $\epsilon / real\ k \leq \epsilon * p0$
by (*metis divide-inverse eps-ge0 mult-left-mono less-eq-real-def mult-cancel-right1*)
qed
then show *?thesis*
using *Red-5-7b [OF 1]* **by** (*simp add: algebra-simps*)
next
case 2
show *?thesis*

```

    using Red-5-7c [OF 2] <k≥256> eps-less1 by simp
  qed
  have B: - (3 / (real k^4)) ≤ (-2 / real k^4) - alpha (hgt p) / card X
    using <card X > k^4> <card Y ≠ 0> <0 < k> alpha-le-1 by (simp add:
algebra-simps frac-le)
    have - (3 / (β * real k^4)) ≤ (-2 / real k^4) / β - alpha (hgt p) / (β *
card X)
      using <β>0> divide-right-mono [OF B, of β] <k>0> by (simp add: field-simps)
    also have ... = (- real (card X) / real k^5) * card Y / (β * real (card X) *
(card Y / (2 * real k))) - alpha (hgt p) / (β * card X)
      using <card Y ≠ 0> <0 < card X>
      by (simp add: field-split-simps eval-nat-numeral)
    also have ... ≤ (- real (card X) / real k^5) * card Y / (β * real (card X)
* card NRY) - alpha (hgt p) / (β * card X)
      using Y-NRY <k>0> <card NRY > 0> <card X > 0> <card Y ≠ 0> <β>0>
      by (intro diff-mono divide-right-mono mult-left-mono divide-left-mono-neg)
    auto
    also have ... ≤ weight X Y x * card Y / (β * real (card X) * card NRY) -
alpha (hgt p) / (β * card X)
      using Red-5-4 <k>0> <0 < β>
      by (intro diff-mono divide-right-mono mult-right-mono) auto
    finally have - (3 / (β * real k^4)) ≤ weight X Y x * card Y / (β * real (card
X) * card NRY) - alpha (hgt p) / (β * card X) .
      then have 17: p + ((1-β)/β) * alpha (hgt p) - 3 / (β * real k^4) ≤ ?E16
      by simp
      have 3 / real k^4 ≤ (1-μ) * ε^2 / k
      using <k>0> μ01 mult-left-mono [OF k52, of k]
      by (simp add: field-simps eps-def powr-powr powr-mult-base flip: powr-numeral
powr-add)
      also have ... ≤ (1-β) * ε^2 / k
      using <β≤μ>
      by (intro divide-right-mono mult-right-mono) auto
      also have ... ≤ (1-β) * ε * alpha (hgt p)
      using Red-5-7a [of p] eps-ge0 <β≤μ> μ01
      unfolding power2-eq-square divide-inverse mult.assoc
      by (intro mult-mono) auto
      finally have †: 3 / real k^4 ≤ (1-β) * ε * alpha (hgt p) .
      have p + (1 - ε) * ((1-β) / β) * alpha (hgt p) + 3 / (β * real k^4) ≤ p +
((1-β)/β) * alpha (hgt p)
      using <0<β> <k>0> mult-left-mono [OF †, of β] by (simp add: field-simps)
      with 16 17 have p + (1 - ε) * ((1 - β) / β) * alpha (hgt p) ≤ red-density
NBX NRY
      by linarith
      then show ?thesis
      using <0 < β> NBX-def NRY-def XY by fastforce
  qed
qed

```

This and the previous result are proved under the assumption of a sufficiently large l

corollary *Red-5-2*:

assumes $i: i \in \text{Step-class } \{\text{dboost-step}\}$
and *Big*: *Big-Red-5-1* μ l
shows $pseq (Suc\ i) - pseq\ i \geq (1 - \varepsilon) * ((1 - \text{beta}\ i) / \text{beta}\ i) * \text{alpha}\ (\text{hgt}\ (pseq\ i)) \wedge$
 $\text{beta}\ i > 0$

proof –

let $?x = cvx\ i$
obtain $X\ Y\ A\ B$
where *step*: *stepper* $i = (X, Y, A, B)$
and *nonterm*: $\neg \text{termination-condition}\ X\ Y$
and *odd* i
and *non-mb*: $\neg \text{many-bluish}\ X$
and *nonredd*: $\neg \text{reddish}\ k\ X\ Y\ (\text{red-density}\ X\ Y)\ (\text{choose-central-vx}\ (X, Y, A, B))$
and *Xeq*: $X = Xseq\ i$ **and** *Yeq*: $Y = Yseq\ i$
using i
by (*auto simp*: *step-kind-defs split*: *if-split-asm prod.split-asm*)
then have $?x \in Xseq\ i$
by (*simp add*: *choose-central-vx-X cvx-def finite-Xseq*)
then have *central-vertex* $(Xseq\ i)\ (cvx\ i)$
by (*metis Xeq choose-central-vx-works cvx-def finite-Xseq step non-mb nonterm*)
with Xeq have $\text{card}\ (\text{Neighbours}\ \text{Blue}\ (cvx\ i) \cap Xseq\ i) \leq \mu * \text{card}\ (Xseq\ i)$
by (*simp add*: *central-vertex-def*)
then have $\beta eq: \text{card}\ (\text{Neighbours}\ \text{Blue}\ (cvx\ i) \cap Xseq\ i) = \text{beta}\ i * \text{card}\ (Xseq\ i)$
 i
using Xeq step by (*auto simp*: *beta-def*)
have *SUC*: *stepper* $(Suc\ i) = (\text{Neighbours}\ \text{Blue}\ ?x \cap X, \text{Neighbours}\ \text{Red}\ ?x \cap Y, A, \text{insert}\ ?x\ B)$
using *step nonterm* $\langle \text{odd}\ i \rangle$ *non-mb nonredd*
by (*simp add*: *stepper-def next-state-def Let-def cvx-def*)
have *pseq*: $pseq\ i = \text{red-density}\ X\ Y$
by (*simp add*: *pseq-def Xeq Yeq*)
have *choose-central-vx* $(X, Y, A, B) = cvx\ i$
by (*simp add*: *cvx-def step*)
with nonredd have $\text{red-density}\ (\text{Neighbours}\ \text{Red}\ (cvx\ i) \cap X)\ (\text{Neighbours}\ \text{Red}\ (cvx\ i) \cap Y)$
 $< pseq\ i - \text{alpha}\ (\text{hgt}\ (\text{red-density}\ X\ Y))$
using nonredd by (*simp add*: *reddish-def pseq*)
then have $pseq\ i + (1 - \varepsilon) * ((1 - \text{beta}\ i) / \text{beta}\ i) * \text{alpha}\ (\text{hgt}\ (pseq\ i))$
 $\leq \text{red-density}\ (\text{Neighbours}\ \text{Blue}\ (cvx\ i) \cap Xseq\ i)$
 $(\text{Neighbours}\ \text{Red}\ (cvx\ i) \cap Yseq\ i) \wedge \text{beta}\ i > 0$
using Red-5-1 Un-iff Xeq Yeq assms gen-density-ge0 pseq Step-class-insert
by (*smt (verit, ccfv-threshold) βeq divide-eq-eq*)
moreover have $\text{red-density}\ (\text{Neighbours}\ \text{Blue}\ (cvx\ i) \cap Xseq\ i)$
 $(\text{Neighbours}\ \text{Red}\ (cvx\ i) \cap Yseq\ i) \leq pseq\ (Suc\ i)$
using SUC Xeq Yeq stepper-XYseq by (*simp add*: *pseq-def*)
ultimately show *?thesis*
by *linarith*
qed

end

4.4 Lemma 5.3

This is a weaker consequence of the previous results

definition

Big-Red-5-3 \equiv
 $\lambda \mu l. \text{Big-Red-5-1 } \mu l$
 $\wedge (\forall k \geq l. k > 1 \wedge 1 / (\text{real } k)^2 \leq \mu \wedge 1 / (\text{real } k)^2 \leq 1 / (k / \text{eps } k / (1 - \text{eps } k) + 1))$

establishing the size requirements for 5.3. The one involving μ , namely $1 / (\text{real } k)^2 \leq \mu$, will be useful later with "big beta".

lemma *Big-Red-5-3*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-Red-5-3 } \mu l$

using *assms Big-Red-5-1*

apply (*simp add: Big-Red-5-3-def eps-def eventually-conj-iff all-imp-conj-distrib*)

apply (*intro conjI strip eventually-all-geI0 eventually-all-ge-at-top*)

apply (*real-asymp|force*) $+$

done

context *Book*

begin

corollary *Red-5-3*:

assumes $i: i \in \text{Step-class } \{\text{dboost-step}\}$

and *big: Big-Red-5-3* μl

shows $\text{pseq } (\text{Suc } i) \geq \text{pseq } i \wedge \text{beta } i \geq 1 / (\text{real } k)^2$

proof

have $k > 1$ **and** *big51: Big-Red-5-1* μl

using *l-le-k big* **by** (*auto simp: Big-Red-5-3-def*)

let $?h = \text{hgt } (\text{pseq } i)$

have $?h > 0$

by (*simp add: hgt-gt0 kn0 pee-le1*)

then obtain $\alpha: \text{alpha } ?h \geq 0$ **and** $*$: $\text{alpha } ?h \geq \varepsilon / k$

using *alpha-ge0* $\langle k > 1 \rangle$ *alpha-ge* **by** *auto*

moreover have $-5/4 = -1/4 - (1::\text{real})$

by *simp*

ultimately have $\alpha 54: \text{alpha } ?h \geq k \text{ powr } (-5/4)$

unfolding *eps-def* **by** (*metis powr-diff of-nat-0-le-iff powr-one*)

have $\beta: \text{beta } i \leq \mu$

by (*metis Step-class-insert Un-iff beta-le i*)

have $(1 - \varepsilon) * ((1 - \text{beta } i) / \text{beta } i) * \text{alpha } ?h \geq 0$

using *beta-ge0* [*of i*] *eps-le1* $\alpha \beta \mu 01 \langle k > 1 \rangle$

by (*simp add: zero-le-mult-iff zero-le-divide-iff*)

then show $\text{pseq } (\text{Suc } i) \geq \text{pseq } i$

```

    using Red-5-2 [OF i big51] by linarith
  have pseq (Suc i) - pseq i ≤ 1
    by (smt (verit) pee-ge0 pee-le1)
  with Red-5-2 [OF i big51]
  have (1 - ε) * ((1 - beta i) / beta i) * alpha ?h ≤ 1 and beta-gt0: beta i > 0
    by linarith+
  with * have (1 - ε) * ((1 - beta i) / beta i) * ε / k ≤ 1
    by (smt (verit, best) mult.commute eps-ge0 mult-mono mult-nonneg-nonpos
of-nat-0-le-iff times-divide-eq-right zero-le-divide-iff)
  then have (1 - ε) * ((1 - beta i) / beta i) ≤ k / ε
    using beta-ge0 [of i] eps-gt0 kn0
    by (auto simp: divide-simps mult-less-0-iff mult-of-nat-commute split: if-split-asm)
  then have (1 - beta i) / beta i ≤ k / ε / (1 - ε)
    by (smt (verit) eps-less1 mult.commute pos-le-divide-eq <1 < k>)
  then have 1 / beta i ≤ k / ε / (1 - ε) + 1
    using beta-gt0 by (simp add: diff-divide-distrib)
  then have 1 / (k / ε / (1 - ε) + 1) ≤ beta i
    using beta-gt0 eps-gt0 eps-less1 [OF <k>1>] kn0
    apply (simp add: divide-simps split: if-split-asm)
    by (smt (verit, cfv-SIG) mult.commute mult-less-0-iff)
  moreover have 1 / k^2 ≤ 1 / (k / ε / (1 - ε) + 1)
    using Big-Red-5-3-def l-le-k big eps-def by (metis (no-types, lifting) of-nat-power)
  ultimately show beta i ≥ 1 / (real k)^2
    by auto
qed

```

```

corollary beta-gt0:
  assumes i ∈ Step-class {dboost-step}
    and Big-Red-5-3 μ l
  shows beta i > 0
  by (meson Big-Red-5-3-def Book.Red-5-2 Book-axioms assms)

```

end

end

5 Bounding the Size of Y

```

theory Bounding-Y imports Red-Steps

```

```

begin

```

yet another telescope variant, with weaker promises but a different conclusion; as written it holds even if $n = 0$

```

lemma prod-lessThan-telescope-mult:
  fixes f::nat ⇒ 'a::field
  assumes ∧i. i < n ⇒ f i ≠ 0
  shows (∏ i < n. f (Suc i) / f i) * f 0 = f n
  using assms

```

by (induction n) (auto simp: divide-simps)

5.1 The following results together are Lemma 6.4

Compared with the paper, all the indices are greater by one!!

context *Book*
begin

lemma *Y-6-4-Red*:

assumes $i \in \text{Step-class } \{\text{red-step}\}$
shows $\text{pseq } (\text{Suc } i) \geq \text{pseq } i - \text{alpha } (\text{hgt } (\text{pseq } i))$
using *assms*
by (auto simp: step-kind-defs next-state-def reddish-def pseq-def
split: if-split-asm prod.split)

lemma *Y-6-4-DegreeReg*:

assumes $i \in \text{Step-class } \{\text{dreg-step}\}$
shows $\text{pseq } (\text{Suc } i) \geq \text{pseq } i$
using *assms red-density-X-degree-reg-ge [OF Xseq-Yseq-disjnt, of i]*
by (auto simp: step-kind-defs degree-reg-def pseq-def split: if-split-asm prod.split-asm)

lemma *Y-6-4-Bblue*:

assumes $i: i \in \text{Step-class } \{\text{bblue-step}\}$
shows $\text{pseq } (\text{Suc } i) \geq \text{pseq } (i-1) - (\varepsilon \text{ powr } (-1/2)) * \text{alpha } (\text{hgt } (\text{pseq } (i-1)))$
proof –
define *X* where $X \equiv X\text{seq } i$
define *Y* where $Y \equiv Y\text{seq } i$
obtain *A B S T*
where *step*: *stepper* $i = (X, Y, A, B)$
and *nonterm*: $\neg \text{termination-condition } X Y$
and *odd* i
and *mb*: *many-bluish* X
and *bluebook*: $(S, T) = \text{choose-blue-book } (X, Y, A, B)$
using i
by (simp add: X-def Y-def step-kind-defs split: if-split-asm prod.split-asm)
(metis mk-edge.cases)
then have *X1-eq*: $X\text{seq } (\text{Suc } i) = T$
by (force simp: Xseq-def next-state-def split: prod.split)
have *Y1-eq*: $Y\text{seq } (\text{Suc } i) = Y$
using i by (simp add: Y-def step-kind-defs next-state-def split: if-split-asm
prod.split-asm prod.split)
have *disjnt* $X Y$
using *Xseq-Yseq-disjnt* X-def Y-def by blast
obtain *fin*: *finite* X *finite* Y
by (metis V-state-stepper finX finY step)
have $X \neq \{\}$ $Y \neq \{\}$
using *gen-density-def nonterm termination-condition-def* by fastforce+
define i' where $i' = i-1$
then have *Suci'*: $\text{Suc } i' = i$

```

  by (simp add: <odd i>)
  have i': i' ∈ Step-class {dreg-step}
  by (metis dreg-before-step Step-class-insert Suci' UnCI i)
  then have Xseq (Suc i') = X-degree-reg (Xseq i') (Yseq i')
        Yseq (Suc i') = Yseq i'
        and nonterm': ¬ termination-condition (Xseq i') (Yseq i')
  by (auto simp: degree-reg-def X-degree-reg-def step-kind-defs split: if-split-asm
prod.split-asm)
  then have Xeq: X = X-degree-reg (Xseq i') (Yseq i')
        and Yeq: Y = Yseq i'
  using Suci' by (auto simp: X-def Y-def)
  define pm where pm ≡ (pseq i' - ε powr (-1/2) * alpha (hgt (pseq i')))
  have T ⊆ X
  using bluebook by (simp add: choose-blue-book-subset fin)
  then have T-reds: ∧x. x ∈ T ⇒ pm * card Y ≤ card (Neighbours Red x ∩
Y)
  by (auto simp: Xeq Yeq pm-def X-degree-reg-def pseq-def red-dense-def)
  have good-blue-book X (S,T)
  by (meson bluebook choose-blue-book-works fin)
  then have Tne: False if card T = 0
  using μ01 <X ≠ {}> fin by (simp add: good-blue-book-def pos-prod-le that)
  have pm * card T * card Y = (∑ x∈T. pm * card Y)
  by simp
  also have ... ≤ (∑ x∈T. card (Neighbours Red x ∩ Y))
  using T-reds by (simp add: sum-bounded-below)
  also have ... = edge-card Red T Y
  using <disjnt X Y> <finite X> <T⊆X> Red-E
  by (metis disjnt-subset1 disjnt-sym edge-card-commute edge-card-eq-sum-Neighbours
finite-subset)
  also have ... = red-density T Y * card T * card Y
  using fin <T⊆X> by (simp add: finite-subset gen-density-def)
  finally have pm ≤ red-density T Y
  using fin <Y≠{}> Yeq Yseq-gt0 Tne nonterm' step-terminating-iff by fastforce
  then show ?thesis
  by (simp add: X1-eq Y1-eq i'-def pseq-def pm-def)
qed

```

The basic form is actually *Red-5-3*. This variant covers a gap of two, thanks to degree regularisation

corollary *Y-6-4-dbooSt*:

assumes $i: i \in \text{Step-class } \{\text{dboost-step}\}$ and big: *Big-Red-5-3* μl
shows $pseq (Suc i) \geq pseq (i-1)$

proof –

```

  have odd ii-1 ∈ Step-class {dreg-step}
  using step-odd i by (auto simp: Step-class-insert-NO-MATCH dreg-before-step)
  then show ?thesis
  using Red-5-3 Y-6-4-DegreeReg assms <odd i> by fastforce
qed

```

5.2 Towards Lemmas 6.3

definition $Z\text{-class} \equiv \{i \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}\}.$
 $\text{pseq } (\text{Suc } i) < \text{pseq } (i-1) \wedge \text{pseq } (i-1) \leq p0\}$

lemma $\text{finite-Z-class: finite } (Z\text{-class})$
using finite-components **by** $(\text{auto simp: } Z\text{-class-def Step-class-insert-NO-MATCH})$

lemma $Y\text{-6-3:}$

assumes $\text{big53: Big-Red-5-3 } \mu l$ **and** $\text{big41: Big-Blue-4-1 } \mu l$
shows $(\sum i \in Z\text{-class. pseq } (i-1) - \text{pseq } (\text{Suc } i)) \leq 2 * \varepsilon$

proof –

define \mathcal{S} **where** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$

define \mathcal{R} **where** $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$

define \mathcal{B} **where** $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$

{ fix i

assume $i: i \in \mathcal{S}$

moreover have $\text{odd } i$

using $\text{step-odd [of } i\text{]} i$ **by** $(\text{force simp: } \mathcal{S}\text{-def Step-class-insert-NO-MATCH})$

ultimately have $i-1 \in \text{Step-class } \{\text{dreg-step}\}$

by $(\text{simp add: } \mathcal{S}\text{-def dreg-before-step Step-class-insert-NO-MATCH})$

then have $\text{pseq } (i-1) \leq \text{pseq } i \wedge \text{pseq } i \leq \text{pseq } (\text{Suc } i)$

using $\text{big53 } \mathcal{S}\text{-def}$

by $(\text{metis Red-5-3 One-nat-def } Y\text{-6-4-DegreeReg } \langle \text{odd } i \rangle i \text{ odd-Suc-minus-one})$

}

then have $\text{dboost: } \mathcal{S} \cap Z\text{-class} = \{\}$

by $(\text{fastforce simp: } Z\text{-class-def})$

{ fix i

assume $i: i \in \mathcal{B} \cap Z\text{-class}$

then have $i-1 \in \text{Step-class } \{\text{dreg-step}\}$

using $\text{dreg-before-step step-odd } i$ **by** $(\text{force simp: } \mathcal{B}\text{-def Step-class-insert-NO-MATCH})$

have $\text{pseq: pseq } (\text{Suc } i) < \text{pseq } (i-1) \text{ pseq } (i-1) \leq p0$ **and** $iB: i \in \mathcal{B}$

using i **by** $(\text{auto simp: } Z\text{-class-def})$

have $\text{hgt } (\text{pseq } (i-1)) = 1$

proof –

have $\text{hgt } (\text{pseq } (i-1)) \leq 1$

by $(\text{smt } (\text{verit, del-insts}) \text{hgt-Least less-one pseq(2) qfun0 qfun-strict-mono})$

then show $?thesis$

by $(\text{metis One-nat-def Suc-pred' diff-is-0-eq hgt-gt0})$

qed

then have $\text{pseq } (i-1) - \text{pseq } (\text{Suc } i) \leq \varepsilon \text{ powr } (-1/2) * \text{alpha } 1$

using $\text{pseq } iB Y\text{-6-4-Bblue } \mu 01$ **by** $(\text{fastforce simp: } \mathcal{B}\text{-def})$

also have $\dots \leq 1/k$

proof –

have $k \text{ powr } (-1/8) \leq 1$

using $kn0$ **by** $(\text{simp add: ge-one-powr-ge-zero powr-minus-divide})$

then show $?thesis$

by $(\text{simp add: alpha-eq eps-def powr-powr divide-le-cancel flip: powr-add})$

qed

finally have $\text{pseq } (i-1) - \text{pseq } (\text{Suc } i) \leq 1/k$.

```

}
then have ( $\sum i \in \mathcal{B} \cap Z\text{-class}. pseq (i-1) - pseq (Suc i)$ )
   $\leq \text{card } (\mathcal{B} \cap Z\text{-class}) * (1/k)$ 
  using sum-bounded-above by (metis (mono-tags, lifting))
also have  $\dots \leq \text{card } (\mathcal{B}) * (1/k)$ 
  using bblue-step-finite
  by (simp add: B-def divide-le-cancel card-mono)
also have  $\dots \leq l \text{ powr } (3/4) / k$ 
  using big41 by (simp add: B-def kn0 frac-le bblue-step-limit)
also have  $\dots \leq \varepsilon$ 
proof -
  have  $l \text{ powr } (3/4) \leq k \text{ powr } (3/4)$ 
    by (simp add: l-le-k powr-mono2)
  have  $3/4 - (1::\text{real}) = - 1/4$ 
    by simp
  then show ?thesis
    using divide-right-mono [OF *, of k]
    by (metis eps-def of-nat-0-le-iff powr-diff powr-one)
qed
finally have bblue: ( $\sum i \in \mathcal{B} \cap Z\text{-class}. pseq(i-1) - pseq (Suc i)$ )  $\leq \varepsilon$  .
{ fix i
  assume i:  $i \in \mathcal{R} \cap Z\text{-class}$ 
  then have pee-alpha:  $pseq (i-1) - pseq (Suc i)$ 
     $\leq pseq (i-1) - pseq i + \alpha (\text{hgt } (pseq i))$ 
    using Y-6-4-Red by (force simp: R-def)
  have pee-le:  $pseq (i-1) \leq pseq i$ 
    using dreg-before-step Y-6-4-DegreeReg[of i-1] i step-odd
    by (simp add: R-def Step-class-insert-NO-MATCH)
  consider (1)  $\text{hgt } (pseq i) = 1$  | (2)  $\text{hgt } (pseq i) > 1$ 
    by (metis hgt-gt0 less-one nat-neq-iff)
  then have  $pseq (i-1) - pseq i + \alpha (\text{hgt } (pseq i)) \leq \varepsilon / k$ 
  proof cases
    case 1
      then show ?thesis
        by (smt (verit) Red-5-7c kn0 pee-le hgt-works)
    next
      case 2
        then have p-gt-q:  $pseq i > \text{qfun } 1$ 
          by (meson hgt-Least not-le zero-less-one)
        have pee-le-q0:  $pseq (i-1) \leq \text{qfun } 0$ 
          using 2 Z-class-def i by auto
        also have pee2:  $\dots \leq pseq i$ 
          using alpha-eq p-gt-q by (smt (verit, best) kn0 qfun-mono zero-le-one)
        finally have  $pseq (i-1) \leq pseq i$  .
        then have  $pseq (i-1) - pseq i + \alpha (\text{hgt } (pseq i))$ 
           $\leq \text{qfun } 0 - pseq i + \varepsilon * (pseq i - \text{qfun } 0 + 1/k)$ 
          using Red-5-7b pee-le-q0 pee2 by fastforce
        also have  $\dots \leq \varepsilon / k$ 
        using kn0 pee2 by (simp add: algebra-simps) (smt (verit) affine-ineq eps-le1)

```

```

    finally show ?thesis .
  qed
  with pcc-alpha have pseq (i-1) - pseq (Suc i) ≤ ε / k
    by linarith
}
then have (∑ i ∈ R ∩ Z-class. pseq (i-1) - pseq (Suc i))
  ≤ card (R ∩ Z-class) * (ε / k)
  using sum-bounded-above by (metis (mono-tags, lifting))
also have ... ≤ card (R) * (ε / k)
  using eps-ge0 assms red-step-finite
  by (simp add: R-def divide-le-cancel mult-le-cancel-right card-mono)
also have ... ≤ k * (ε / k)
  using red-step-limit R-def μ01
  by (smt (verit, best) divide-nonneg-nonneg eps-ge0 mult-mono nat-less-real-le
of-nat-0-le-iff)
  also have ... ≤ ε
    using eps-ge0 by force
  finally have red: (∑ i ∈ R ∩ Z-class. pseq (i-1) - pseq (Suc i)) ≤ ε .
  have *: finite (B) finite (R) ∧ x. x ∈ B ⇒ x ∉ R
    using finite-components by (auto simp: B-def R-def Step-class-def)
  have eq: Z-class = S ∩ Z-class ∪ B ∩ Z-class ∪ R ∩ Z-class
    by (auto simp: Z-class-def B-def R-def S-def Step-class-insert-NO-MATCH)
  show ?thesis
    using bblue red
    by (subst eq) (simp add: sum.union-disjoint dboost disjoint-iff *)
qed

```

5.3 Lemma 6.5

lemma *Y-6-5-Red*:

```

  assumes i: i ∈ Step-class {red-step} and k ≥ 16
  defines h ≡ λi. hgt (pseq i)
  shows h (Suc i) ≥ h i - 2
  proof (cases h i ≤ 3)
  case True
  have h (Suc i) ≥ 1
    by (simp add: h-def Suc-leI hgt-gt0)
  with True show ?thesis
    by linarith
  next
  case False
  have k > 0 using assms by auto
  have ε ≤ 1/2
    using ⟨k ≥ 16⟩ by (simp add: eps-eq-sqrt divide-simps real-le-rsqrt)
  moreover have 0 ≤ x ∧ x ≤ 1/2 ⇒ x * (1 + x)2 + 1 ≤ (1 + x)2 for x::real
    by sos
  ultimately have §: ε * (1 + ε)2 + 1 ≤ (1 + ε)2
    using eps-ge0 by presburger
  have le1: ε + 1 / (1 + ε)2 ≤ 1

```

using *mult-left-mono* [*OF* §, *of inverse* $((1 + \varepsilon)^2)$]
by (*simp add: ring-distrib inverse-eq-divide*) (*smt (verit)*)
have $0: 0 \leq (1 + \varepsilon) \wedge (h\ i - \text{Suc } 0)$
using *eps-ge0* **by** *auto*
have *lesspi*: $qfun\ (h\ i - 1) < pseq\ i$
using *False hgt-Least* [*of h i - 1 pseq i*] **unfolding** *h-def* **by** *linarith*
have $A: (1 + \varepsilon) \wedge h\ i = (1 + \varepsilon) * (1 + \varepsilon) \wedge (h\ i - \text{Suc } 0)$
using *False power.simps* **by** (*metis h-def Suc-pred hgt-gt0*)
have $B: (1 + \varepsilon) \wedge (h\ i - 3) = 1 / (1 + \varepsilon)^2 * (1 + \varepsilon) \wedge (h\ i - \text{Suc } 0)$
using *eps-gt0 False*
by (*simp add: divide-simps Suc-diff-Suc numeral-3-eq-3 flip: power-add*)
have $qfun\ (h\ i - 3) \leq qfun\ (h\ i - 1) - (qfun\ (h\ i) - qfun\ (h\ i - 1))$
using *kn0 mult-left-mono [OF le1 0]*
by (*simp add: qfun-eq A B algebra-simps divide-right-mono flip: add-divide-distrib*
diff-divide-distrib)
also have $\dots < pseq\ i - \alpha\ (h\ i)$
using *lesspi* **by** (*simp add: alpha-def*)
also have $\dots \leq pseq\ (\text{Suc } i)$
using *Y-6-4-Red i* **by** (*force simp: h-def*)
finally have $qfun\ (h\ i - 3) < pseq\ (\text{Suc } i)$.
with *hgt-greater* **show** *?thesis*
unfolding *h-def* **by** *force*
qed

lemma *Y-6-5-DegreeReg*:
assumes $i \in \text{Step-class } \{\text{dreg-step}\}$
shows $hgt\ (pseq\ (\text{Suc } i)) \geq hgt\ (pseq\ i)$
using *hgt-mono Y-6-4-DegreeReg assms* **by** *presburger*

corollary *Y-6-5-dbooSt*:
assumes $i \in \text{Step-class } \{\text{dboost-step}\}$ **and** *Big-Red-5-3* $\mu\ l$
shows $hgt\ (pseq\ (\text{Suc } i)) \geq hgt\ (pseq\ i)$
using *kn0 Red-5-3 assms hgt-mono* **by** *blast*

this remark near the top of page 19 only holds in the limit

lemma $\forall^\infty k. (1 + \text{eps } k) \text{ powr } (- \text{real } (\text{nat } \lfloor 2 * \text{eps } k \text{ powr } (-1/2) \rfloor)) \leq 1 - \text{eps } k \text{ powr } (1/2)$
unfolding *eps-def* **by** *real-asymp*

end

definition *Big-Y-6-5-Bblue* \equiv
 $\lambda l. \forall k \geq l. (1 + \text{eps } k) \text{ powr } (- \text{real } (\text{nat } \lfloor 2 * (\text{eps } k \text{ powr } (-1/2)) \rfloor)) \leq 1 - \text{eps } k \text{ powr } (1/2)$

establishing the size requirements for Y 6.5

lemma *Big-Y-6-5-Bblue*:
shows $\forall^\infty l. \text{Big-Y-6-5-Bblue } l$
unfolding *Big-Y-6-5-Bblue-def eps-def* **by** (*intro eventually-all-ge-at-top; real-asymp*)

lemma (in *Book*) *Y-6-5-Bblue*:
fixes $\kappa::\text{real}$
defines $\kappa \equiv \varepsilon \text{ powr } (-1/2)$
assumes $i: i \in \text{Step-class } \{\text{bblue-step}\}$ **and** $\text{big}: \text{Big-}Y-6-5\text{-Bblue } l$
defines $h \equiv \text{hgt } (\text{pseq } (i-1))$
shows $\text{hgt } (\text{pseq } (\text{Suc } i)) \geq h - 2*\kappa$
proof (cases $h > 2*\kappa + 1$)
case *True*
then have $0 < h - 1$
by (smt (verit, best) $\kappa\text{-def one-less-of-natD powr-non-neg zero-less-diff}$)
with *True* **have** $\text{pseq } (i-1) > \text{qfun } (h-1)$
by (simp add: $h\text{-def hgt-less-imp-qfun-less}$)
then have $\text{qfun } (h-1) - \varepsilon \text{ powr } (1/2) * (1 + \varepsilon) ^ (h-1) / k < \text{pseq } (i-1) -$
 $\kappa * \text{alpha } h$
using $\langle 0 < h-1 \rangle$ *Y-6-4-Bblue* [OF i] eps-ge0
apply (simp add: $\text{alpha-eq } \kappa\text{-def}$)
by (smt (verit, best) $\text{field-sum-of-halves mult.assoc mult.commute powr-mult-base}$)
also have $\dots \leq \text{pseq } (\text{Suc } i)$
using *Y-6-4-Bblue* i $h\text{-def } \kappa\text{-def}$ **by** *blast*
finally have $A: \text{qfun } (h-1) - \varepsilon \text{ powr } (1/2) * (1 + \varepsilon) ^ (h-1) / k < \text{pseq } (\text{Suc}$
 $i)$.
have $ek0: 0 < 1 + \varepsilon$
by (smt (verit, best) eps-ge0)
have $\text{less-h}: \text{nat } \lfloor 2*\kappa \rfloor < h$
using *True* $\langle 0 < h - 1 \rangle$ **by** *linarith*
have $\text{qfun } (h - \text{nat } \lfloor 2*\kappa \rfloor - 1) = p0 + ((1 + \varepsilon) ^ (h - \text{nat } \lfloor 2*\kappa \rfloor - 1) - 1)$
 $/ k$
by (simp add: qfun-eq)
also have $\dots \leq p0 + ((1 - \varepsilon \text{ powr } (1/2)) * (1 + \varepsilon) ^ (h-1) - 1) / k$
proof -
have $\text{ge0}: (1 + \varepsilon) ^ (h-1) \geq 0$
using eps-ge0 **by** *auto*
have $(1 + \varepsilon) ^ (h - \text{nat } \lfloor 2*\kappa \rfloor - 1) = (1 + \varepsilon) ^ (h-1) * (1 + \varepsilon) \text{ powr } -$
 $\text{real}(\text{nat } \lfloor 2*\kappa \rfloor)$
using $\text{less-h } ek0$ **by** (simp add: $\text{algebra-simps flip: powr-realpow powr-add}$)
also have $\dots \leq (1 - \varepsilon \text{ powr } (1/2)) * (1 + \varepsilon) ^ (h-1)$
using $\text{big } l\text{-le-}k$ **unfolding** $\kappa\text{-def Big-}Y-6-5\text{-Bblue-def}$
by (metis $\text{mult.commute ge0 mult-left-mono}$)
finally have $(1 + \varepsilon) ^ (h - \text{nat } \lfloor 2*\kappa \rfloor - 1)$
 $\leq (1 - \varepsilon \text{ powr } (1/2)) * (1 + \varepsilon) ^ (h-1)$.
then show *?thesis*
by (intro $\text{add-left-mono divide-right-mono diff-right-mono}$) *auto*
qed
also have $\dots \leq \text{qfun } (h-1) - \varepsilon \text{ powr } (1/2) * (1 + \varepsilon) ^ (h-1) / \text{real } k$
using $kn0 \text{ eps-ge0}$ **by** (simp add: $\text{qfun-eq powr-half-sqrt field-simps}$)
also have $\dots < \text{pseq } (\text{Suc } i)$
using A **by** *blast*
finally have $\text{qfun } (h - \text{nat } \lfloor 2*\kappa \rfloor - 1) < \text{pseq } (\text{Suc } i)$.

then have $h - \text{nat } \lfloor 2 * \kappa \rfloor \leq \text{hgt } (\text{pseq } (\text{Suc } i))$
using *hgt-greater* **by force**
with *less-h* **show** *?thesis*
unfolding *κ -def*
by (*smt (verit) less-imp-le-nat of-nat-diff of-nat-floor of-nat-mono powr-ge-zero*)
next
case *False*
then show *?thesis*
by (*smt (verit, del-insts) of-nat-0 hgt-gt0 nat-less-real-le*)
qed

5.4 Lemma 6.2

definition *Big-Y-6-2* $\equiv \lambda \mu l. \text{Big-Y-6-5-Bblue } l \wedge \text{Big-Red-5-3 } \mu l \wedge \text{Big-Blue-4-1 } \mu l$

$$\wedge (\forall k \geq l. ((1 + \text{eps } k)^2) * \text{eps } k \text{ powr } (1/2) \leq 1 \\ \wedge (1 + \text{eps } k) \text{ powr } (2 * \text{eps } k \text{ powr } (-1/2)) \leq 2 \wedge k \geq 16)$$

establishing the size requirements for 6.2

lemma *Big-Y-6-2*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0 .. \mu 1\} \longrightarrow \text{Big-Y-6-2 } \mu l$

using *assms Big-Y-6-5-Bblue Big-Red-5-3 Big-Blue-4-1*

unfolding *Big-Y-6-2-def eps-def*

apply (*simp add: eventually-conj-iff all-imp-conj-distrib*)

apply (*intro conjI strip eventually-all-geI1 eventually-all-ge-at-top; real-asymp*)

done

context *Book*

begin

Following Bhavik in excluding the even steps (degree regularisation). Assuming it hasn't halted, the conclusion also holds for the even cases anyway.

proposition *Y-6-2*:

defines *RBS* $\equiv \text{Step-class } \{\text{red-step, bblue-step, dboost-step}\}$

assumes *j*: $j \in \text{RBS}$ **and** *big*: *Big-Y-6-2* μl

shows $\text{pseq } (\text{Suc } j) \geq p0 - 3 * \varepsilon$

proof (*cases pseq (Suc j) $\geq p0$*)

case *True*

then show *?thesis*

by (*smt (verit) eps-ge0*)

next

case *False*

then have *pj-less*: $\text{pseq } (\text{Suc } j) < p0$ **by** *linarith*

have *big53*: *Big-Red-5-3* μl

and *Y63*: $(\sum i \in \text{Z-class}. \text{pseq } (i-1) - \text{pseq } (\text{Suc } i)) \leq 2 * \varepsilon$

and *Y65B*: $\bigwedge i. i \in \text{Step-class } \{\text{bbblue-step}\} \implies \text{hgt } (\text{pseq } (\text{Suc } i)) \geq \text{hgt } (\text{pseq } (i-1)) - 2 * (\varepsilon \text{ powr } (-1/2))$

and *big1*: $((1 + \varepsilon)^2) * \varepsilon \text{ powr } (1/2) \leq 1$ **and** *big2*: $(1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/2)) \leq 2$

```

and  $k \geq 16$ 
using big Y-6-5-Bblue Y-6-3 kn0 l-le-k by (auto simp: Big-Y-6-2-def)
have Y64-S:  $\bigwedge i. i \in \text{Step-class } \{\text{dboost-step}\} \implies \text{pseq } i \leq \text{pseq } (\text{Suc } i)$ 
using big53 Red-5-3 by simp
define J where  $J \equiv \{j'. j' < j \wedge \text{pseq } j' \geq p0 \wedge \text{even } j'\}$ 
have finite J
by (auto simp: J-def)
have  $\text{pseq } 0 = p0$ 
by (simp add: pee-eq-p0)
have odd-RBS: odd i if i ∈ RBS for i
using step-odd that unfolding RBS-def by blast
with odd-pos j have  $j > 0$  by auto
have non-halted:  $j \notin \text{Step-class } \{\text{halted}\}$ 
using j by (auto simp: Step-class-def RBS-def)
have exists:  $J \neq \{\}$ 
using  $\langle 0 < j \rangle \langle \text{pseq } 0 = p0 \rangle$  by (force simp: J-def less-eq-real-def)
define j' where  $j' \equiv \text{Max } J$ 
have  $j' \in J$ 
using  $\langle \text{finite } J \rangle \text{ exists}$  by (force simp: j'-def)
then have  $j' < j$  even j' and  $\text{pSj}'$ :  $\text{pseq } j' \geq p0$ 
by (auto simp: J-def odd-RBS)
have maximal:  $j'' \leq j'$  if j'' ∈ J for j''
using  $\langle \text{finite } J \rangle \text{ exists}$  by (simp add: j'-def that)
have  $\text{pseq } (j'+2) - 2 * \varepsilon \leq \text{pseq } (j'+2) - (\sum i \in \text{Z-class. pseq } (i-1) - \text{pseq } (\text{Suc } i))$ 
using Y63 by simp
also have  $\dots \leq \text{pseq } (\text{Suc } j)$ 
proof -
define Z where  $Z \equiv \lambda j. \{i. \text{pseq } (\text{Suc } i) < \text{pseq } (i-1) \wedge j'+2 < i \wedge i \leq j \wedge i \in \text{RBS}\}$ 
have Zsub:  $Z i \subseteq \{\text{Suc } j' < .. i\}$  for i
by (auto simp: Z-def)
then have finZ: finite (Z i) for i
by (meson finite-greaterThanAtMost finite-subset)
have  $*$ :  $(\sum i \in Z j. \text{pseq } (i-1) - \text{pseq } (\text{Suc } i)) \leq (\sum i \in \text{Z-class. pseq } (i-1) - \text{pseq } (\text{Suc } i))$ 
proof (intro sum-mono2 [OF finite-Z-class])
show  $Z j \subseteq \text{Z-class}$ 
proof
fix i
assume  $i: i \in Z j$ 
then have dreg:  $i-1 \in \text{Step-class } \{\text{dreg-step}\}$  and  $i \neq 0$   $j' < i$ 
by (auto simp: Z-def RBS-def dreg-before-step)
with i dreg maximal have  $\text{pseq } (i-1) < p0$ 
unfolding Z-def J-def
using Suc-less-eq2 less-eq-Suc-le odd-RBS by fastforce
then show  $i \in \text{Z-class}$ 
using i by (simp add: Z-def RBS-def Z-class-def)
qed

```

```

    show  $0 \leq pseq (i-1) - pseq (Suc i)$  if  $i \in Z\text{-class} - Z j$  for  $i$ 
    using that by (auto simp: Z-def Z-class-def)
  qed
  then have  $pseq (j'+2) - (\sum_{i \in Z\text{-class}} pseq (i-1) - pseq (Suc i))$ 
     $\leq pseq (j'+2) - (\sum_{i \in Z j} pseq (i-1) - pseq (Suc i))$ 
    by auto
  also have  $\dots \leq pseq (Suc j)$ 
  proof -
    have  $pseq (j'+2) - pseq (Suc m) \leq (\sum_{i \in Z m} pseq (i-1) - pseq (Suc i))$ 
      if  $m \in RBS$   $j' < m$   $m \leq j$  for  $m$ 
      using that
    proof (induction m rule: less-induct)
      case (less m)
      then have odd m
        using odd-RBS by blast
      show ?case
      proof (cases  $j'+2 < m$ )
        case True
        with less.prem
          have Z-if:  $Z m = (\text{if } pseq (Suc m) < pseq (m-1) \text{ then insert } m (Z (m-2)) \text{ else } Z (m-2))$ 
          by (auto simp: Z-def)
          (metis le-diff-conv2 Suc-leI add-2-eq-Suc' add-leE even-Suc nat-less-le odd-RBS)+
          have  $m-2 \in RBS$ 
            using True  $\langle m \in RBS \rangle$  step-odd-minus2 by (auto simp: RBS-def)
          then have *:  $pseq (j'+2) - pseq (m - Suc 0) \leq (\sum_{i \in Z (m-2)} pseq (i-1) - pseq (Suc i))$ 
            using less.IH True less  $\langle j' \in J \rangle$  by (force simp: J-def Suc-less-eq2)
          moreover have  $m \notin Z (m-2)$ 
            by (auto simp: Z-def)
          ultimately show ?thesis
            by (simp add: Z-if finZ)
        case False
      next
      case False
      then have [simp]:  $m = Suc j'$ 
        using  $\langle odd m \rangle \langle j' < m \rangle \langle even j' \rangle$  by presburger
      have  $Z m = \{\}$ 
        by (auto simp: Z-def)
      then show ?thesis
        by simp
    qed
  qed
  then show ?thesis
    using j J-def  $\langle j' \in J \rangle \langle j' < j \rangle$  by force
  qed
  finally show ?thesis .
  qed
  finally have p2-le-pSuc:  $pseq (j'+2) - 2 * \varepsilon \leq pseq (Suc j)$  .

```

```

have Suc j' ∈ RBS
  unfolding RBS-def
proof (intro not-halted-odd-RBS)
  show Suc j' ∉ Step-class {halted}
    using Step-class-halted-forever Suc-leI ⟨j' < j⟩ non-halted by blast
qed (use ⟨even j'⟩ in auto)
then have pseq (j'+2) < p0
  using maximal[of j'+2] False ⟨j' < j⟩ j odd-RBS
  by (simp add: J-def) (smt (verit, best) Suc-lessI even-Suc)
then have le1: hgt (pseq (j'+2)) ≤ 1
  by (smt (verit) kn0 hgt-Least qfun0 qfun-strict-mono zero-less-one)
moreover
have j'-dreg: j' ∈ Step-class {dreg-step}
  using RBS-def ⟨Suc j' ∈ RBS⟩ dreg-before-step by blast
have 1: ε powr -(1/2) ≥ 1
  using kn0 by (simp add: eps-def powr-powr ge-one-powr-ge-zero)
consider (R) Suc j' ∈ Step-class {red-step}
  | (B) Suc j' ∈ Step-class {bbblue-step}
  | (S) Suc j' ∈ Step-class {dboost-step}
  by (metis Step-class-insert UnE ⟨Suc j' ∈ RBS⟩ RBS-def)
note j'-cases = this
then have hgt-le-hgt: hgt (pseq j') ≤ hgt (pseq (j'+2)) + 2 * ε powr (-1/2)
proof cases
  case R
  have real (hgt (pseq j')) ≤ hgt (pseq (Suc j'))
    using Y-6-5-DegreeReg[OF j'-dreg] kn0 by (simp add: eval-nat-numeral)
  also have ... ≤ hgt (pseq (j'+2)) + 2 * ε powr (-1/2)
    using Y-6-5-Red[OF R ⟨k≥16⟩] 1 by (simp add: eval-nat-numeral)
  finally show ?thesis .
next
  case B
  show ?thesis
    using Y65B [OF B] by simp
next
  case S
  then show ?thesis
    using Y-6-4-DegreeReg ⟨pseq (j'+2) < p0⟩ Y64-S j'-dreg pSj' by force
qed
ultimately have B: hgt (pseq j') ≤ 1 + 2 * ε powr (-1/2)
  by linarith
have 2 ≤ real k powr (1/2)
  using ⟨k≥16⟩ by (simp add: powr-half-sqrt real-le-rsqrt)
then have 8: 2 ≤ real k powr 1 * real k powr -(1/8)
  unfolding powr-add [symmetric] using ⟨k≥16⟩ order.trans nle-le by fastforce
have p0 - ε ≤ qfun 0 - 2 * ε powr (1/2) / k
  using mult-left-mono [OF 8, of k powr (-1/8)] kn0
  by (simp add: qfun-eq eps-def powr-powr field-simps flip: powr-add)
also have ... ≤ pseq j' - ε powr (-1/2) * alpha (hgt (pseq j'))
proof -

```

```

have 2: (1 + ε) ^ (hgt (pseq j') - Suc 0) ≤ 2
  using B big2 kn0 eps-ge0
  by (smt (verit) diff-Suc-less hgt-gt0 nat-less-real-le powr-mono powr-realpow)
have *: x ≥ 0 ⇒ inverse (x powr (1/2)) * x = x powr (1/2) for x::real
  by (simp add: inverse-eq-divide powr-half-sqrt real-div-sqrt)
have p0 - pseq j' ≤ 0
  by (simp add: pSj')
also have ... ≤ 2 * ε powr (1/2) / k - (ε powr (1/2)) * (1 + ε) ^ (hgt
(pseq j') - 1) / k
  using mult-left-mono [OF 2, of ε powr (1/2) / k]
  by (simp add: field-simps diff-divide-distrib)
finally have p0 - 2 * ε powr (1/2) / k
  ≤ pseq j' - (ε powr (1/2)) * (1 + ε) ^ (hgt (pseq j') - 1) / k
  by simp
with * [OF eps-ge0] show ?thesis
  by (simp add: alpha-hgt-eq powr-minus) (metis mult.assoc)
qed
also have ... ≤ pseq (j'+2)
  using j'-cases
proof cases
case R
  have hs-le3: hgt (pseq (Suc j')) ≤ 3
    using le1 Y-6-5-Red[OF R <k≥16>] by simp
  then have h-le3: hgt (pseq j') ≤ 3
    using Y-6-5-DegreeReg [OF j'-dreg] by simp
  have alpha1: alpha (hgt (pseq (Suc j'))) ≤ ε * (1 + ε) ^ 2 / k
    by (metis alpha-Suc-eq alpha-mono hgt-gt0 hs-le3 numeral-nat(3))
  have alpha2: alpha (hgt (pseq j')) ≥ ε / k
    by (simp add: Red-5-7a)
  have pseq j' - ε powr (- 1/2) * alpha (hgt (pseq j'))
    ≤ pseq (Suc j') - alpha (hgt (pseq (Suc j')))
  proof -
    have alpha (hgt (pseq (Suc j'))) ≤ (1 + ε)2 * alpha (hgt (pseq j'))
      using alpha1 mult-left-mono [OF alpha2, of (1 + ε)2]
      by (simp add: mult.commute)
    also have ... ≤ inverse (ε powr (1/2)) * alpha (hgt (pseq j'))
      using mult-left-mono [OF big1, of alpha (hgt (pseq j'))] eps-gt0 alpha-ge0
      by (simp add: divide-simps mult-ac)
    finally have alpha (hgt (pseq (Suc j')))
      ≤ inverse (ε powr (1/2)) * alpha (hgt (pseq j')) .
  then show ?thesis
    using Y-6-4-DegreeReg[OF j'-dreg] by (simp add: powr-minus)
  qed
also have ... ≤ pseq (j'+2)
  by (simp add: R Y-6-4-Red)
finally show ?thesis .
next
case B
  then show ?thesis

```

```

    using Y-6-4-Bblue by force
next
  case S
  show ?thesis
    using Y-6-4-DegreeReg S <pseq (j'+2) < p0> Y64-S j'-dreg pSj' by fastforce
  qed
  finally have p0 - ε ≤ pseq (j'+2) .
  then have p0 - 3 * ε ≤ pseq (j'+2) - 2 * ε
    by simp
  with p2-le-pSuc show ?thesis
    by linarith
qed

corollary Y-6-2-halted:
  assumes big: Big-Y-6-2 μ l
  shows pseq halted-point ≥ p0 - 3 * ε
proof (cases halted-point=0)
  case True
  then show ?thesis
    by (simp add: eps-ge0 pee-eq-p0)
next
  case False
  then have halted-point-1 ∉ Step-class {halted}
    by (simp add: halted-point-minimal)
  then consider halted-point-1 ∈ Step-class {red-step,bblue-step,dboost-step}
    | halted-point-1 ∈ Step-class {dreg-step}
    using not-halted-even-dreg not-halted-odd-RBS by blast
  then show ?thesis
proof cases
  case 1
  with False Y-6-2[of halted-point-1] big show ?thesis by simp
next
  case m1-dreg: 2
  then have *: pseq halted-point ≥ pseq (halted-point-1)
    using False Y-6-4-DegreeReg[of halted-point-1] by simp
  have odd halted-point
    using m1-dreg False step-even[of halted-point-1] by simp
  then consider halted-point=1 | halted-point ≥ 2
    by (metis False less-2-cases One-nat-def not-le)
  then show ?thesis
proof cases
  case 1
  with * eps-gt0 kn0 show ?thesis
    by (simp add: pee-eq-p0)
next
  case 2
  then have m2: halted-point-2 ∈ Step-class {red-step,bblue-step,dboost-step}
    using step-before-dreg[of halted-point-2] m1-dreg
    by (simp flip: Suc-diff-le)

```

then obtain j where j : halted-point-1 = Suc j
using 2 not0-implies-Suc by fastforce
then have $pseq$ (Suc j) $\geq p0 - 3 * \varepsilon$
by (metis m2 Suc-1 Y-6-2 big diff-Suc-1 diff-Suc-eq-diff-pred)
with $* j$ show ?thesis by simp
qed
qed
qed
end

5.5 Lemma 6.1

context $P0$ -min
begin

definition ok -fun-61 $\equiv \lambda k. (2 * real\ k) * \log\ 2\ (1 - 2 * eps\ k\ powr\ (1/2) / p0$ -min)

lemma ok -fun-61-works:

assumes $p0$ -min $> 2 * eps\ k\ powr\ (1/2)$
shows $2\ powr\ (ok$ -fun-61 $k) = (1 - 2 * (eps\ k) powr\ (1/2) / p0$ -min) $^ (2*k)$
using $p0$ -min assms
by (simp add: powr-def ok-fun-61-def log-def flip: powr-realpow)

lemma ok -fun-61: ok -fun-61 $\in o$ (real)

unfolding eps -def ok-fun-61-def
using $p0$ -min by real-asymp

definition

Big -Y-6-1 \equiv
 $\lambda \mu\ l. Big$ -Y-6-2 $\mu\ l \wedge (\forall k \geq l. eps\ k\ powr\ (1/2) \leq 1/3 \wedge p0$ -min $> 2 * eps\ k$
 $powr\ (1/2))$

establishing the size requirements for 6.1

lemma Big -Y-6-1:

assumes $0 < \mu 0\ \mu 1 < 1$
shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow Big$ -Y-6-1 $\mu\ l$
using $p0$ -min assms Big -Y-6-2
unfolding Big -Y-6-1-def eps -def
apply (simp add: eventually-conj-iff all-imp-conj-distrib)
apply (intro conjI strip eventually-all-ge-at-top eventually-all-geI0; real-asymp)
done

end

lemma (in Book) Y-6-1:

assumes big: Big -Y-6-1 $\mu\ l$
defines $st \equiv Step$ -class {red-step,dboost-step}
shows $card$ (Yseq halted-point) / $card\ Y0 \geq 2\ powr\ (ok$ -fun-61 $k) * p0 \wedge card\ st$

```

proof –
  have big13:  $\varepsilon \text{ powr } (1/2) \leq 1/3$ 
    and big-p0:  $p0\text{-min} > 2 * \varepsilon \text{ powr } (1/2)$ 
    and big62: Big-Y-6-2  $\mu$  l
    and big41: Big-Blue-4-1  $\mu$  l
    using big l-le-k by (auto simp: Big-Y-6-1-def Big-Y-6-2-def)
  with l-le-k have dboost-step-limit:  $\text{card } (\text{Step-class } \{\text{dboost-step}\}) < k$ 
    using bblue-dboost-step-limit by fastforce
  define p0m where  $p0m \equiv p0 - 2 * \varepsilon \text{ powr } (1/2)$ 
  have  $p0m > 0$ 
    using big-p0 p0-ge by (simp add: p0m-def)
  let ?RS = Step-class {red-step,dboost-step}
  let ?BD = Step-class {bblue-step,dreg-step}
  have not-halted-below-m:  $i \notin \text{Step-class } \{\text{halted}\}$  if  $i < \text{halted-point}$  for i
    using that by (simp add: halted-point-minimal)
  have BD-card:  $\text{card } (Yseq\ i) = \text{card } (Yseq\ (\text{Suc } i))$ 
    if  $i \in ?BD$  for i
  proof –
    have  $Yseq\ (\text{Suc } i) = Yseq\ i$ 
      using that
      by (auto simp: step-kind-defs next-state-def degree-reg-def split: prod.split
if-split-asm)
    with p0-01 kn0 show ?thesis
      by auto
  qed
  have RS-card:  $p0m * \text{card } (Yseq\ i) \leq \text{card } (Yseq\ (\text{Suc } i))$ 
    if  $i \in ?RS$  for i
  proof –
    have Yeq:  $Yseq\ (\text{Suc } i) = \text{Neighbours Red } (cvx\ i) \cap Yseq\ i$ 
      using that
      by (auto simp: step-kind-defs next-state-def split: prod.split if-split-asm)
    have odd i
      using that step-odd by (auto simp: Step-class-def)
    moreover have i-not-halted:  $i \notin \text{Step-class } \{\text{halted}\}$ 
      using that by (auto simp: Step-class-def)
    ultimately have iminus1-dreg:  $i - 1 \in \text{Step-class } \{\text{dreg-step}\}$ 
      by (simp add: dreg-before-step not-halted-odd-RBS)
    have  $p0m * \text{card } (Yseq\ i) \leq (1 - \varepsilon \text{ powr } (1/2)) * pseq\ (i-1) * \text{card } (Yseq\ i)$ 
    proof (cases i=1)
      case True
        with p0-01 show ?thesis
          by (simp add: p0m-def pee-eq-p0 algebra-simps mult-right-mono)
      next
        case False
          with  $\langle \text{odd } i \rangle$  have  $i > 2$ 
            by (metis Suc-lessI dvd-refl One-nat-def odd-pos one-add-one plus-1-eq-Suc)
          have  $i - 2 \in \text{Step-class } \{\text{red-step}, \text{bblue-step}, \text{dboost-step}\}$ 
          proof (intro not-halted-odd-RBS)
            show  $i - 2 \notin \text{Step-class } \{\text{halted}\}$ 

```

```

    using i-not-halted Step-class-not-halted diff-le-self by blast
  show odd (i-2)
    using <2 < i> <odd i> by auto
qed
then have Y62: pseq (i-1) ≥ p0 - 3 * ε
  using Y-6-2 [OF - big62] <2 < i> by (metis Suc-1 Suc-diff-Suc Suc-lessD)
show ?thesis
proof (intro mult-right-mono)
  have ε powr (1/2) * pseq (i-1) ≤ ε powr (1/2) * 1
    by (metis mult.commute mult-right-mono powr-ge-zero pee-le1)
  moreover have 3 * ε ≤ ε powr (1/2)
  proof -
    have 3 * ε = 3 * (ε powr (1/2))2
      using eps-ge0 powr-half-sqrt real-sqrt-pow2 by presburger
    also have ... ≤ 3 * ((1/3) * ε powr (1/2))
      by (smt (verit) big13 mult-right-mono power2-eq-square powr-ge-zero)
    also have ... ≤ ε powr (1/2)
      by simp
    finally show ?thesis .
  qed
  ultimately show p0m ≤ (1 - ε powr (1/2)) * pseq (i - 1)
    using Y62 by (simp add: p0m-def algebra-simps)
  qed auto
qed
also have ... ≤ card (Neighbours Red (cvx i) ∩ Yseq i)
  using Red-5-8 [OF iminus1-dreg] cvx-in-Xseq that <odd i>
  by fastforce
finally show ?thesis
  by (simp add: Yeq)
qed
define ST where ST ≡ λi. ?RS ∩ {..}
have ST (Suc i) = (if i ∈ ?RS then insert i (ST i) else ST i) for i
  by (auto simp: ST-def less-Suc-eq)
then have [simp]: card (ST (Suc i)) = (if i ∈ ?RS then Suc (card (ST i)) else
card (ST i)) for i
  by (simp add: ST-def)
have STm: ST halted-point = st
  by (auto simp: ST-def st-def Step-class-def simp flip: halted-point-minimal)
have p0m ^ card (ST i) ≤ (∏j<i. card (Yseq(Suc j)) / card (Yseq j)) if
i ≤ halted-point for i
  using that
proof (induction i)
  case 0
  then show ?case
    by (auto simp: ST-def)
next
  case (Suc i)
  then have i: i ∉ Step-class {halted}
    by (simp add: not-halted-below-m)

```

```

consider (RS)  $i \in ?RS$ 
  | (BD)  $i \in ?BD \wedge i \notin ?RS$ 
  using  $i$  stepkind.exhaust by (auto simp: Step-class-def)
then show ?case
proof cases
  case RS
  then have  $p0m \wedge \text{card } (ST \text{ (Suc } i)) = p0m * p0m \wedge \text{card } (ST \text{ } i)$ 
    by simp
  also have  $\dots \leq p0m * (\prod_{j < i} \text{card } (Yseq(\text{Suc } j)) / \text{card } (Yseq \text{ } j))$ 
    using Suc Suc-leD <0 < p0m> mult-left-mono by auto
  also have  $\dots \leq (\text{card } (Yseq \text{ (Suc } i)) / \text{card } (Yseq \text{ } i)) * (\prod_{j < i} \text{card } (Yseq$ 
(Suc j)) / \text{card } (Yseq j))
  proof (intro mult-right-mono)
    show  $p0m \leq \text{card } (Yseq \text{ (Suc } i)) / \text{card } (Yseq \text{ } i)$ 
      by (simp add: RS RS-card Yseq-gt0 i pos-le-divide-eq)
    qed (simp add: prod-nonneg)
  also have  $\dots = (\prod_{j < \text{Suc } i} \text{card } (Yseq \text{ (Suc } j)) / \text{card } (Yseq \text{ } j))$ 
    by simp
  finally show ?thesis .
next
  case BD
  with Yseq-gt0 [OF i] show ?thesis
    by (simp add: Suc Suc-leD BD-card)
  qed
qed
then have  $p0m \wedge \text{card } (ST \text{ halted-point}) \leq (\prod_{j < \text{halted-point}} \text{card } (Yseq(\text{Suc}$ 
j)) / \text{card } (Yseq j))
  by blast
also have  $\dots = \text{card } (Yseq \text{ halted-point}) / \text{card } (Yseq \text{ } 0)$ 
proof –
  have  $\bigwedge i. i < \text{halted-point} \implies \text{card } (Yseq \text{ } i) \neq 0$ 
    by (metis Yseq-gt0 less-irrefl not-halted-below-m)
  then show ?thesis
    using card-XY0 prod-lessThan-telescope-mult [of halted-point  $\lambda i. \text{real } (\text{card}$ 
(Yseq i))]
    by (simp add: nonzero-eq-divide-eq)
  qed
finally have  $*$ :  $(p0 - 2 * \varepsilon \text{ powr } (1/2)) \wedge \text{card } st \leq \text{card } (Yseq \text{ halted-point})$ 
/ \text{card } (Y0)
  by (simp add: STm p0m-def)
  – Asymptotic part of the argument
have st-le-2k:  $\text{card } st \leq 2 * k$ 
proof –
  have  $st \subseteq \text{Step-class } \{\text{red-step}, \text{dboost-step}\}$ 
    by (auto simp: st-def Step-class-insert-NO-MATCH)
  moreover have finite (Step-class {red-step,dboost-step})
    using finite-components by (auto simp: Step-class-insert-NO-MATCH)
  ultimately have  $\text{card } st \leq \text{card } (\text{Step-class } \{\text{red-step}, \text{dboost-step}\})$ 
    using card-mono by blast

```

```

also have ... = card (Step-class {red-step} ∪ Step-class {dboost-step})
  by (auto simp: Step-class-insert-NO-MATCH)
also have ... ≤ k+k
  by (meson add-le-mono card-Un-le dboost-step-limit le-trans less-imp-le-nat
red-step-limit)
  finally show ?thesis
    by auto
qed
have 2 powr (ok-fun-61 k) * p0 ^ card st ≤ (p0 - 2 * ε powr (1/2)) ^ card st
proof -
  have 2 powr (ok-fun-61 k) = (1 - 2 * ε powr(1/2) / p0-min) ^ (2*k)
    using big-p0 ok-fun-61-works by blast
  also have ... ≤ (1 - 2 * ε powr(1/2) / p0) ^ (2*k)
    using p0-ge p0-min big-p0 by (intro power-mono) (auto simp: frac-le)
  also have ... ≤ (1 - 2 * ε powr(1/2) / p0) ^ card st
    using big-p0 p0-01 <0 < p0m>
    by (intro power-decreasing st-le-2k) (auto simp: p0m-def)
  finally have §: 2 powr ok-fun-61 k ≤ (1 - 2 * ε powr (1/2) / p0) ^ card st .
  have (1 - 2 * ε powr (1/2) / p0) ^ card st * p0 ^ card st
    = ((1 - 2 * ε powr (1/2) / p0) * p0) ^ card st
    by (simp add: power-mult-distrib)
  also have ... = (p0 - 2 * ε powr (1/2)) ^ card st
    using p0-01 by (simp add: algebra-simps)
  finally show ?thesis
    using mult-right-mono [OF §, of p0 ^ card st] p0-01 by auto
qed
with * show ?thesis
  by linarith
qed
end

```

6 Bounding the Size of X

theory Bounding- X **imports** Bounding- Y

begin

6.1 Preliminaries

lemma sum-odds-even:

fixes $f :: \text{nat} \Rightarrow 'a :: \text{ab-group-add}$

assumes even m

shows $(\sum i \in \{i. i < m \wedge \text{odd } i\}. f (\text{Suc } i) - f (i - \text{Suc } 0)) = f m - f 0$

using *assms*

proof (induction m rule: less-induct)

case (less m)

show ?case

proof (cases $m < 2$)

```

    case True
    with <even m> show ?thesis
      by fastforce
  next
  case False
  have eq: {i. i < m ∧ odd i} = insert (m-1) {i. i < m-2 ∧ odd i}
  proof
    show {i. i < m ∧ odd i} ⊆ insert (m-1) {i. i < m-2 ∧ odd i}
      using <even m> by clarify presburger
    qed (use False less in auto)
  have [simp]: ¬ (m - Suc 0 < m - 2)
    by linarith
  show ?thesis
    using False by (simp add: eq less flip: numeral-2-eq-2)
  qed
qed

lemma sum-odds-odd:
  fixes f :: nat ⇒ 'a :: ab-group-add
  assumes odd m
  shows (∑ i ∈ {i. i < m ∧ odd i}. f (Suc i) - f (i - Suc 0)) = f (m-1) - f 0
  proof -
    have eq: {i. i < m ∧ odd i} = {i. i < m-1 ∧ odd i}
      using assms not-less-iff-gr-or-eq by fastforce
    show ?thesis
      by (simp add: sum-odds-even eq assms)
  qed

context Book
begin

  the set of moderate density-boost steps (page 20)

  definition dboost-star where
    dboost-star ≡ {i ∈ Step-class {dboost-step}. real (hgt (pseq (Suc i))) - hgt (pseq
    i) ≤ ε powr (-1/4)}

  definition bigbeta where
    bigbeta ≡ let S = dboost-star in if S = {} then μ else (card S) * inverse (∑ i ∈ S.
    inverse (beta i))

  lemma dboost-star-subset: dboost-star ⊆ Step-class {dboost-step}
    by (auto simp: dboost-star-def)

  lemma finite-dboost-star: finite (dboost-star)
    by (meson dboost-step-finite dboost-star-subset finite-subset)

  lemma bigbeta-ge0: bigbeta ≥ 0
    using μ01 by (simp add: bigbeta-def Let-def beta-ge0 sum-nonneg)

```

```

lemma bigbeta-ge-square:
  assumes big: Big-Red-5-3  $\mu$  l
  shows bigbeta  $\geq 1 / (\text{real } k)^2$ 
proof -
  have k:  $1 / (\text{real } k)^2 \leq \mu$ 
    using big kn0 l-le-k by (auto simp: Big-Red-5-3-def)
  have fin: finite (dboost-star)
    using assms finite-dboost-star by blast
  have R53:  $\forall i \in \text{Step-class } \{\text{dboost-step}\}. 1 / (\text{real } k)^2 \leq \text{beta } i$ 
    using Red-5-3 assms by blast
  show  $1 / (\text{real } k)^2 \leq \text{bigbeta}$ 
  proof (cases dboost-star = {})
    case True
    then show ?thesis
      using assms k by (simp add: bigbeta-def)
  next
    case False
    then have card-gt0:  $\text{card } (\text{dboost-star}) > 0$ 
      by (meson card-gt-0-iff dboost-star-subset fin finite-subset)
    moreover have *:  $\forall i \in \text{dboost-star}. \text{beta } i > 0 \wedge (\text{real } k)^2 \geq \text{inverse } (\text{beta } i)$ 
    i)
      using R53 kn0 assms by (simp add: beta-gt0 field-simps dboost-star-def)
    ultimately have  $(\sum_{i \in \text{dboost-star}} \text{inverse } (\text{beta } i)) \leq \text{card } (\text{dboost-star}) * (\text{real } k)^2$ 
      by (simp add: sum-bounded-above)
    moreover have  $(\sum_{i \in \text{dboost-star}} \text{inverse } (\text{beta } i)) \neq 0$ 
      by (metis * False fin inverse-positive-iff-positive less-irrefl sum-pos)
    ultimately show ?thesis
      using False card-gt0 k bigbeta-ge0
      by (simp add: bigbeta-def Let-def divide-simps split: if-split-asm)
  qed
qed

```

```

lemma bigbeta-gt0:
  assumes big: Big-Red-5-3  $\mu$  l
  shows bigbeta  $> 0$ 
by (smt (verit) kn0 assms bigbeta-ge-square of-nat-zero-less-power-iff zero-less-divide-iff)

```

```

lemma bigbeta-less1:
  assumes big: Big-Red-5-3  $\mu$  l
  shows bigbeta  $< 1$ 
proof -
  have *:  $\forall i \in \text{Step-class } \{\text{dboost-step}\}. 0 < \text{beta } i$ 
    using assms beta-gt0 big by blast
  have fin: finite (Step-class {dboost-step})
    using dboost-step-finite assms by blast
  show bigbeta  $< 1$ 

```

```

proof (cases dboost-star = {})
  case True
    then show ?thesis
      using assms  $\mu 01$  by (simp add: bigbeta-def)
  next
    case False
      then have gt0: card (dboost-star) > 0
        by (meson card-gt-0-iff dboost-star-subset fin finite-subset)
      have real (card (dboost-star)) =  $(\sum_{i \in \text{dboost-star}} 1)$ 
        by simp
      also have ... <  $(\sum_{i \in \text{dboost-star}} 1 / \text{beta } i)$ 
      proof (intro sum-strict-mono)
        show finite (dboost-star)
          using card-gt-0-iff gt0 by blast
        fix i
          assume  $i \in \text{dboost-star}$ 
          with assms  $\mu 01$  * dboost-star-subset beta-le
          show  $1 < 1 / \text{beta } i$ 
            by (force simp: Step-class-insert-NO-MATCH)
      qed (use False in auto)
      finally show ?thesis
        using False by (simp add: bigbeta-def Let-def divide-simps)
qed
qed

lemma bigbeta-le:
  assumes big: Big-Red-5-3  $\mu$  l
  shows bigbeta  $\leq \mu$ 
proof -
  have real (card (dboost-star)) =  $(\sum_{i \in \text{dboost-star}} 1)$ 
    by simp
  also have ...  $\leq (\sum_{i \in \text{dboost-star}} \mu / \text{beta } i)$ 
  proof (intro sum-mono)
    fix i
    assume  $i: i \in \text{dboost-star}$ 
    with beta-le dboost-star-subset have  $\text{beta } i \leq \mu$ 
      by (auto simp: Step-class-insert-NO-MATCH)
    with beta-gt0 assms show  $1 \leq \mu / \text{beta } i$ 
      by (smt (verit) dboost-star-subset divide-less-eq-1-pos i subset-iff)
  qed
  also have ... =  $\mu * (\sum_{i \in \text{dboost-star}} 1 / \text{beta } i)$ 
    by (simp add: sum-distrib-left)
  finally have real (card (dboost-star))  $\leq \mu * (\sum_{i \in \text{dboost-star}} 1 / \text{beta } i)$  .
  moreover have  $(\sum_{i \in \text{dboost-star}} 1 / \text{beta } i) \geq 0$ 
    by (simp add: beta-ge0 sum-nonneg)
  ultimately show ?thesis
    using  $\mu 01$  by (simp add: bigbeta-def Let-def divide-simps)
qed

```

end

6.2 Lemma 7.2

definition *Big-X-7-2* $\equiv \lambda \mu l. \text{nat } \lceil \text{real } l \text{ powr } (3/4) \rceil \geq 3 \wedge l > 1 / (1-\mu)$

establishing the size requirements for 7.11

lemma *Big-X-7-2*:

assumes $0 < \mu < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu_0.. \mu_1\} \longrightarrow \text{Big-X-7-2 } \mu l$

unfolding *Big-X-7-2-def eventually-conj-iff all-imp-conj-distrib eps-def*

apply (*simp add: eventually-conj-iff all-imp-conj-distrib*)

apply (*intro conjI strip eventually-all-geI1 [where L=1] eventually-all-ge-at-top*)

apply *real-asymp+*

by (*smt (verit, best) <\mu < 1> frac-le*)

definition *ok-fun-72* $\equiv \lambda \mu k. (\text{real } k / \ln 2) * \ln (1 - 1 / (k * (1-\mu)))$

lemma *ok-fun-72*:

assumes $\mu < 1$

shows *ok-fun-72* $\mu \in o(\text{real})$

using *assms unfolding ok-fun-72-def by real-asymp*

lemma *ok-fun-72-uniform*:

assumes $0 < \mu < 1$

assumes $e > 0$

shows $\forall^\infty k. \forall \mu. \mu_0 \leq \mu \wedge \mu \leq \mu_1 \longrightarrow |ok-fun-72 \mu k| / k \leq e$

proof (*intro eventually-all-geI1 [where L = Suc(nat[1/(1-\mu1)])]*)

show $\forall^\infty k. |ok-fun-72 \mu_1 k| / \text{real } k \leq e$

using *assms unfolding ok-fun-72-def by real-asymp*

next

fix $k \mu$

assume *le-e*: $|ok-fun-72 \mu_1 k| / \text{real } k \leq e$

and $\mu: \mu_0 \leq \mu \leq \mu_1$

and $k: \text{Suc}(\text{nat}[1/(1-\mu_1)]) \leq k$

with *assms* **have** $1 > 1 / (\text{real } k * (1 - \mu_1))$

by (*smt (verit, best) divide-less-eq divide-less-eq-1 less-eq-Suc-le natceiling-lessD*)

then **have** $*$: $1 > 1 / (\text{real } k * (1 - r))$ **if** $r \leq \mu_1$ **for** r

using *that assms k less-le-trans by fastforce*

have \dagger : $1 / (k * (1 - \mu)) \leq 1 / (k * (1 - \mu_1))$

using μ *assms by (simp add: divide-simps mult-less-0-iff)*

obtain $\mu < 1$ $k > 0$ **using** μk *assms by force*

then **have** $|ok-fun-72 \mu k| \leq |ok-fun-72 \mu_1 k|$

using $\mu *$ *assms \dagger*

by (*simp add: ok-fun-72-def abs-mult zero-less-mult-iff abs-of-neg divide-le-cancel*)

then **show** $|ok-fun-72 \mu k| / \text{real } k \leq e$

by (*smt (verit, best) le-e divide-right-mono of-nat-0-le-iff*)

qed

lemma (*in Book*) *X-7-2*:

```

defines  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ 
assumes big: Big-X-7-2  $\mu$  l
shows  $(\prod_{i \in \mathcal{R}} \text{card } (X\text{seq}(\text{Suc } i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } (\text{ok-fun-72 } \mu \text{ } k) * (1-\mu) \wedge \text{card } \mathcal{R}$ 
proof –
  define R where  $R \equiv \text{RN } k \text{ (nat } \lceil \text{real } l \text{ powr } (3/4) \rceil)$ 
  have l34-ge3:  $\text{nat } \lceil \text{real } l \text{ powr } (3/4) \rceil \geq 3$  and k-gt:  $k > 1 / (1-\mu)$ 
    using big l-le-k by (auto simp: Big-X-7-2-def)
  then obtain  $R > k$   $k \geq 2$ 
    using μ01 RN-gt1 R-def l-le-k
    by (smt (verit, best) divide-le-eq-1-pos fact-2 nat-le-real-less of-nat-fact)
  with k-gt μ01 have bigR:  $1-\mu > 1/R$ 
    by (smt (verit, best) less-imp-of-nat-less ln-div ln-le-cancel-iff zero-less-divide-iff)
  have  $*$ :  $1-\mu - 1/R \leq \text{card } (X\text{seq}(\text{Suc } i)) / \text{card } (X\text{seq } i)$ 
    if  $i \in \mathcal{R}$  for  $i$ 
  proof –
    let  $?NRX = \lambda i. \text{Neighbours Red } (\text{cvx } i) \cap X\text{seq } i$ 
    have nextX:  $X\text{seq } (\text{Suc } i) = ?NRX \text{ } i$  and nont:  $\neg \text{termination-condition } (X\text{seq } i)$ 
    using that by (auto simp: R-def step-kind-defs next-state-def split: prod.split)
    then have cardX:  $\text{card } (X\text{seq } i) > R$ 
      unfolding R-def by (meson not-less termination-condition-def)
    have  $1$ :  $\text{card } (?NRX \text{ } i) \geq (1-\mu) * \text{card } (X\text{seq } i) - 1$ 
      using that card-cvx-Neighbours μ01 by (simp add: R-def Step-class-def)
    have  $R \neq 0$ 
      using  $\langle k < R \rangle$  by linarith
    with cardX have  $(1-\mu) - 1/R \leq (1-\mu) - 1 / \text{card } (X\text{seq } i)$ 
      by (simp add: inverse-of-nat-le)
    also have  $\dots \leq \text{card } (X\text{seq}(\text{Suc } i)) / \text{card } (X\text{seq } i)$ 
      using cardX nextX 1 by (simp add: divide-simps)
    finally show ?thesis .
  qed
  have fin-red: finite  $\mathcal{R}$ 
    using red-step-finite by (auto simp: R-def)
  define  $t$  where  $t \equiv \text{card } \mathcal{R}$ 
  have  $t \geq 0$ 
    by (auto simp: t-def)
  have  $(1-\mu - 1/R) \wedge \text{card Red-steps} \leq (\prod_{i \in \text{Red-steps}} \text{card } (X\text{seq}(\text{Suc } i)) / \text{card } (X\text{seq } i))$ 
    if  $\text{Red-steps} \subseteq \mathcal{R}$  for Red-steps
    using finite-subset [OF that fin-red] that
  proof induction
    case empty
    then show ?case
      by auto
  next
    case (insert i Red-steps)
    then have  $i: i \in \mathcal{R}$ 
      by auto

```

have $((1-\mu) - 1/R) \wedge \text{card}(\text{insert } i \text{ Red-steps}) = ((1-\mu) - 1/R) * ((1-\mu) - 1/R) \wedge \text{card}(\text{Red-steps})$
by *(simp add: insert)*
also have $\dots \leq (\text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i)) * ((1-\mu) - 1/R) \wedge \text{card}(\text{Red-steps})$
using *bigR by (intro mult-right-mono * i) auto*
also have $\dots \leq (\text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i)) * (\prod i \in \text{Red-steps}. \text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i))$
using *insert by (intro mult-left-mono) auto*
also have $\dots = (\prod i \in \text{insert } i \text{ Red-steps}. \text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i))$
using *insert by simp*
finally show *?case .*
qed
then have $*$: $(1-\mu - 1/R) \wedge t \leq (\prod i \in \mathcal{R}. \text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i))$
using *t-def by blast*
— Asymptotic part of the argument
have $1-\mu - 1/k \leq 1-\mu - 1/R$
using *kn0 <k < R> by (simp add: inverse-of-nat-le)*
then have *ln-le*: $\ln(1-\mu - 1/k) \leq \ln(1-\mu - 1/R)$
using *μ01 k-gt <R>k> by (simp add: bigR divide-simps mult.commute less-le-trans)*
have *ok-fun-72* $\mu k * \ln 2 = k * \ln(1 - 1 / (k * (1-\mu)))$
by *(simp add: ok-fun-72-def)*
also have $\dots \leq t * \ln(1 - 1 / (k * (1-\mu)))$
proof *(intro mult-right-mono-neg)*
have *red-steps*: $\text{card } \mathcal{R} < k$
using *red-step-limit <0 < μ> by (auto simp: R-def)*
show *real* $t \leq \text{real } k$
using *nat-less-le red-steps by (simp add: t-def)*
show $\ln(1 - 1 / (k * (1-\mu))) \leq 0$
using *μ01 divide-less-eq k-gt ln-one-minus-pos-upper-bound by fastforce*
qed
also have $\dots = t * \ln((1-\mu - 1/k) / (1-\mu))$
using *<t ≥ 0> μ01 by (simp add: diff-divide-distrib)*
also have $\dots = t * (\ln(1-\mu - 1/k) - \ln(1-\mu))$
using *<t ≥ 0> μ01 k-gt kn0 ln-div by force*
also have $\dots \leq t * (\ln(1-\mu - 1/R) - \ln(1-\mu))$
by *(simp add: ln-le mult-left-mono)*
finally have *ok-fun-72* $\mu k * \ln 2 + t * \ln(1-\mu) \leq t * \ln(1-\mu - 1/R)$
by *(simp add: ring-distrib)*
then have *2 powr ok-fun-72* $\mu k * (1-\mu) \wedge t \leq (1-\mu - 1/R) \wedge t$
using *μ01 by (simp add: bigR ln-mult ln-powr ln-realpow flip: ln-le-cancel-iff)*
with $*$ **show** *?thesis*
by *(simp add: t-def)*
qed

6.3 Lemma 7.3

context *Book*

begin

definition $Bdelta \equiv \lambda \mu i. Bseq (Suc i) \setminus Bseq i$

lemma *card-Bdelta*: $card (Bdelta \mu i) = card (Bseq (Suc i)) - card (Bseq i)$
by (*simp add: Bseq-mono Bdelta-def card-Diff-subset finite-Bseq*)

lemma *card-Bseq-mono*: $card (Bseq (Suc i)) \geq card (Bseq i)$
by (*simp add: Bseq-Suc-subset card-mono finite-Bseq*)

lemma *card-Bseq-sum*: $card (Bseq i) = (\sum j < i. card (Bdelta \mu j))$

proof (*induction i*)

case 0

then show *?case*

by *auto*

next

case (*Suc i*)

with *card-Bseq-mono* **show** *?case*

unfolding *card-Bdelta sum.lessThan-Suc*

by (*smt (verit, del-insts) Nat.add-diff-assoc diff-add-inverse*)

qed

definition *get-blue-book* $\equiv \lambda i. let (X, Y, A, B) = stepper i in choose-blue-book (X, Y, A, B)$

Tracking changes to X and B. The sets are necessarily finite

lemma *Bdelta-bblue-step*:

assumes $i \in Step-class \{bblue-step\}$

shows $\exists S \subseteq Xseq i. Bdelta \mu i = S$

$\wedge card (Xseq (Suc i)) \geq (\mu \wedge card S) * card (Xseq i) / 2$

proof –

obtain $X Y A B S T$ **where** *step: stepper i = (X, Y, A, B)* **and** *bb: get-blue-book i = (S, T)*

and *valid: valid-state(X, Y, A, B)*

by (*metis surj-pair valid-state-stepper*)

moreover **have** *finite X*

by (*metis V-state-stepper finX step*)

ultimately **have** $*$: *stepper (Suc i) = (T, Y, A, B \cup S) \wedge good-blue-book X (S, T)*

and *Xeq: X = Xseq i*

using *assms choose-blue-book-works [of X S T Y A B]*

by (*simp-all add: step-kind-defs next-state-def valid-state-def get-blue-book-def*

choose-blue-book-works split: if-split-asm)

show *?thesis*

proof (*intro exI conjI*)

have $S \subseteq X$

proof (*intro choose-blue-book-subset [THEN conjunct1] <finite X>*)

show $(S, T) = choose-blue-book (X, Y, A, B)$

using *bb step* **by** (*simp add: get-blue-book-def Xseq-def*)

```

qed
then show  $S \subseteq Xseq\ i$ 
  using  $Xeq$  by force
have  $disjnt\ X\ B$ 
  using  $valid$  by (auto simp:  $valid-state-def\ disjoint-state-def$ )
then show  $Bdelta\ \mu\ i = S$ 
  using *  $step\ \langle S \subseteq X \rangle$  by (auto simp:  $Bdelta-def\ Bseq-def\ disjnt-iff$ )
show  $\mu \wedge card\ S * real\ (card\ (Xseq\ i)) / 2 \leq real\ (card\ (Xseq\ (Suc\ i)))$ 
  using * by (auto simp:  $Xseq-def\ good-blue-book-def\ step$ )
qed
qed

lemma  $Bdelta-dboost-step$ :
  assumes  $i \in Step-class\ \{dboost-step\}$ 
  shows  $\exists x \in Xseq\ i. Bdelta\ \mu\ i = \{x\}$ 
proof -
  obtain  $X\ Y\ A\ B$  where  $step: stepper\ i = (X, Y, A, B)$  and  $valid: valid-state(X, Y, A, B)$ 
    by (metis  $surj-pair\ valid-state-stepper$ )
  have  $cvx: choose-central-vx\ (X, Y, A, B) \in X$ 
    by (metis  $Step-class-insert\ Un-iff\ cvx-def\ cvx-in-Xseq\ assms\ step\ stepper-XYseq$ )
  then have  $\exists X'\ Y'. stepper\ (Suc\ i) = (X', Y', A, insert\ (choose-central-vx\ (X, Y, A, B))\ B)$ 
    using  $assms\ step$ 
    by (auto simp:  $step-kind-defs\ next-state-def\ split: if-split-asm$ )
  moreover have  $choose-central-vx\ (X, Y, A, B) \notin B$ 
    using  $valid\ cvx$  by (force simp:  $valid-state-def\ disjoint-state-def\ disjnt-iff$ )
  ultimately show ?thesis
    using  $step\ cvx$  by (auto simp:  $Bdelta-def\ Bseq-def\ disjnt-iff\ Xseq-def$ )
qed

lemma  $card-Bdelta-dboost-step$ :
  assumes  $i \in Step-class\ \{dboost-step\}$ 
  shows  $card\ (Bdelta\ \mu\ i) = 1$ 
  using  $Bdelta-dboost-step\ [OF\ assms]$  by force

lemma  $Bdelta-trivial-step$ :
  assumes  $i: i \in Step-class\ \{red-step, dreg-step, halted\}$ 
  shows  $Bdelta\ \mu\ i = \{\}$ 
  using  $assms$ 
  by (auto simp:  $step-kind-defs\ next-state-def\ Bdelta-def\ degree-reg-def\ split: if-split-asm\ prod.split$ )

end

definition  $ok-fun-73 \equiv \lambda k. - (real\ k\ powr\ (3/4))$ 

lemma  $ok-fun-73: ok-fun-73 \in o(real)$ 
  unfolding  $ok-fun-73-def$  by  $real-asymp$ 

```

lemma (in *Book*) *X-7-3*:
assumes *big*: *Big-Blue-4-1* μ *l*
defines $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$
defines $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
shows $(\prod i \in \mathcal{B}. \text{card } (X\text{seq}(\text{Suc } i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } (\text{ok-fun-73 } k) * \mu \wedge (l - \text{card } \mathcal{S})$
proof –
have [*simp*]: *finite* \mathcal{B} *finite* \mathcal{S} **and** *cardB*: $\text{card } \mathcal{B} \leq l \text{ powr } (3/4)$
using *assms bblue-step-limit big* **by** (*auto simp: B-def S-def*)
define *b* **where** $b \equiv \lambda i. \text{card } (B\text{delta } \mu i)$
obtain *i* **where** $\text{card } (B\text{seq } i) = \text{sum } b \mathcal{B} + \text{card } \mathcal{S}$
proof –
define *i* **where** $i = \text{Suc } (\text{Max } (\mathcal{B} \cup \mathcal{S}))$
define *TRIV* **where** $\text{TRIV} \equiv \text{Step-class } \{\text{red-step, dreg-step, halted}\} \cap \{..<i\}$
have [*simp*]: *finite* *TRIV*
by (*auto simp: TRIV-def*)
have *eq*: $\mathcal{B} \cup \mathcal{S} \cup \text{TRIV} = \{..<i\}$
proof
show $\mathcal{B} \cup \mathcal{S} \cup \text{TRIV} \subseteq \{..<i\}$
by (*auto simp: i-def TRIV-def less-Suc-eq-le*)
show $\{..<i\} \subseteq \mathcal{B} \cup \mathcal{S} \cup \text{TRIV}$
using *stepkind.exhaust* **by** (*auto simp: B-def S-def TRIV-def Step-class-def*)
qed
have *dis*: $\mathcal{B} \cap \mathcal{S} = \{\}$ $(\mathcal{B} \cup \mathcal{S}) \cap \text{TRIV} = \{\}$
by (*auto simp: B-def S-def TRIV-def Step-class-def*)
show *thesis*
proof
have $\text{card } (B\text{seq } i) = (\sum j \in \mathcal{B} \cup \mathcal{S} \cup \text{TRIV}. b j)$
using *card-Bseq-sum eq unfolding b-def* **by** *metis*
also have $\dots = (\sum j \in \mathcal{B}. b j) + (\sum j \in \mathcal{S}. b j) + (\sum j \in \text{TRIV}. b j)$
by (*simp add: sum-Un-nat dis*)
also have $\dots = \text{sum } b \mathcal{B} + \text{card } \mathcal{S}$
by (*simp add: b-def S-def card-Bdelta-dboost-step TRIV-def Bdelta-trivial-step*)
finally show $\text{card } (B\text{seq } i) = \text{sum } b \mathcal{B} + \text{card } \mathcal{S}$.
qed
qed
then have *sum-b-B*: $\text{sum } b \mathcal{B} \leq l - \text{card } \mathcal{S}$
by (*metis Bseq-less-l less-diff-conv nat-less-le*)
have *real* $(\text{card } \mathcal{B}) \leq \text{real } k \text{ powr } (3/4)$
using *cardB l-le-k*
by (*smt (verit, best) divide-nonneg-pos of-nat-0-le-iff of-nat-mono powr-mono2*)
then have $2 \text{ powr } (\text{ok-fun-73 } k) \leq (1/2) \wedge \text{card } \mathcal{B}$
by (*simp add: ok-fun-73-def powr-minus divide-simps flip: powr-realpow*)
then have $2 \text{ powr } (\text{ok-fun-73 } k) * \mu \wedge (l - \text{card } \mathcal{S}) \leq (1/2) \wedge \text{card } \mathcal{B} * \mu \wedge (l - \text{card } \mathcal{S})$
by (*simp add: $\mu 01$*)
also have $(1/2) \wedge \text{card } \mathcal{B} * \mu \wedge (l - \text{card } \mathcal{S}) \leq (1/2) \wedge \text{card } \mathcal{B} * \mu \wedge (\text{sum } b \mathcal{B})$
using *$\mu 01$ sum-b-B* **by** *simp*

also have $\dots = (\prod_{i \in \mathcal{B}} \mu \wedge b \ i \ / \ 2)$
by (*simp add: power-sum prod-dividef divide-simps*)
also have $\dots \leq (\prod_{i \in \mathcal{B}} \text{card} (Xseq (Suc \ i)) / \text{card} (Xseq \ i))$
proof (*rule prod-mono*)
fix $i :: \text{nat}$
assume $i \in \mathcal{B}$
then have $\neg \text{termination-condition} (Xseq \ i) (Yseq \ i)$
by (*simp add: B-def Step-class-def flip: step-non-terminating-iff*)
then have $\text{card} (Xseq \ i) \neq 0$
using *termination-condition-def by force*
with $\langle i \in \mathcal{B} \rangle \ \mu 01$ **show** $0 \leq \mu \wedge b \ i \ / \ 2 \wedge \mu \wedge b \ i \ / \ 2 \leq \text{card} (Xseq (Suc \ i)) / \text{card} (Xseq \ i)$
by (*force simp: b-def B-def divide-simps dest!: Bdelta-bblue-step*)
qed
finally show *?thesis .*
qed

6.4 Lemma 7.5

Small $o(k)$ bounds on summations for this section

This is the explicit upper bound for heights given just below (5) on page 9

definition *ok-fun-26* $\equiv \lambda k. \ 2 * \ln k / \text{eps } k$

definition *ok-fun-28* $\equiv \lambda k. \ -2 * \text{real } k \text{ powr } (7/8)$

lemma *ok-fun-26*: $ok\text{-fun-26} \in o(\text{real})$ **and** *ok-fun-28*: $ok\text{-fun-28} \in o(\text{real})$

unfolding *ok-fun-26-def ok-fun-28-def eps-def by real-asymp+*

definition

Big-X-7-5 \equiv

$\lambda \mu \ l. \ \text{Big-Blue-4-1 } \mu \ l \wedge \text{Big-Red-5-3 } \mu \ l \wedge \text{Big-Y-6-5-Bblue } l$

$\wedge (\forall k \geq l. \ \text{Big-height-upper-bound } k \wedge k \geq 16 \wedge (ok\text{-fun-26 } k - ok\text{-fun-28 } k \leq k))$

establishing the size requirements for 7.5

lemma *Big-X-7-5*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \ \forall \mu. \ \mu \in \{\mu 0 .. \mu 1\} \longrightarrow \text{Big-X-7-5 } \mu \ l$

proof –

have $ok: \forall^\infty l. \ ok\text{-fun-26 } l - ok\text{-fun-28 } l \leq l$

unfolding *eps-def ok-fun-26-def ok-fun-28-def by real-asymp*

show *?thesis*

using *assms Big-Y-6-5-Bblue Big-Red-5-3 Big-Blue-4-1*

unfolding *Big-X-7-5-def*

apply (*simp add: eventually-conj-iff all-imp-conj-distrib*)

apply (*intro conjI strip eventually-all-ge-at-top ok Big-height-upper-bound; real-asymp*)

```

done
qed

context Book
begin

lemma X-26-and-28:
  assumes big: Big-X-7-5  $\mu$  l
  defines  $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$ 
  defines  $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$ 
  defines  $\mathcal{H} \equiv \text{Step-class } \{\text{halted}\}$ 
  defines  $h \equiv \lambda i. \text{real } (\text{hgt } (\text{pseq } i))$ 
  obtains  $(\sum i \in \{..<\text{halted-point}\} \setminus \mathcal{D}. h(\text{Suc } i) - h(i-1)) \leq \text{ok-fun-26 } k$ 
     $\text{ok-fun-28 } k \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$ 
proof -
  define  $\mathcal{S}$  where  $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$ 
  have B-limit: Big-Blue-4-1  $\mu$  l and bigY65B: Big-Y-6-5-Bblue l
    and hub: Big-height-upper-bound k
    using big l-le-k by (auto simp: Big-X-7-5-def)
  have m-minimal:  $i \notin \mathcal{H} \longleftrightarrow i < \text{halted-point}$  for i
    unfolding  $\mathcal{H}$ -def using halted-point-minimal assms by blast
  have oddset:  $\{..<\text{halted-point}\} \setminus \mathcal{D} = \{i \in \{..<\text{halted-point}\}. \text{odd } i\}$ 
    using m-minimal step-odd step-even not-halted-even-dreg
    by (auto simp:  $\mathcal{D}$ -def  $\mathcal{H}$ -def Step-class-insert-NO-MATCH)
    — working on 28
  have ok-fun-28  $k \leq -2 * \varepsilon \text{ powr } (-1/2) * \text{card } \mathcal{B}$ 
proof -
  have  $k \text{ powr } (1/8) * \text{card } \mathcal{B} \leq k \text{ powr } (1/8) * l \text{ powr } (3/4)$ 
    using B-limit bblue-step-limit by (simp add:  $\mathcal{B}$ -def mult-left-mono)
  also have  $\dots \leq k \text{ powr } (1/8) * k \text{ powr } (3/4)$ 
    by (simp add: l-le-k mult-mono powr-mono2)
  also have  $\dots = k \text{ powr } (7/8)$ 
    by (simp flip: powr-add)
  finally show ?thesis
    by (simp add: eps-def powr-powr ok-fun-28-def)
qed
  also have  $\dots \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$ 
proof -
  have  $(\sum i \in \mathcal{B}. -2 * \varepsilon \text{ powr } (-1/2)) \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$ 
proof (rule sum-mono)
  fix i :: nat
  assume i:  $i \in \mathcal{B}$ 
  show  $-2 * \varepsilon \text{ powr } (-1/2) \leq h(\text{Suc } i) - h(i-1)$ 
    using bigY65B kn0 i Y-6-5-Bblue by (fastforce simp:  $\mathcal{B}$ -def h-def)
qed
  then show ?thesis
    by (simp add: mult.commute)
qed
  finally have 28:  $\text{ok-fun-28 } k \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$  .

```

have $(\sum i \in \{..<halted-point\} \setminus \mathcal{D}. h(Suc\ i) - h(i-1)) \leq h\ halted-point - h\ 0$
proof (*cases even halted-point*)
 case *False*
 have $hgt\ (pseq\ (halted-point - Suc\ 0)) \leq hgt\ (pseq\ halted-point)$
 using *Y-6-5-DegreeReg [of halted-point-1] False m-minimal not-halted-even-dreg odd-pos*
 by (*fastforce simp: H-def*)
 then have $h(halted-point - Suc\ 0) \leq h\ halted-point$
 using *h-def of-nat-mono by blast*
 with *False show ?thesis*
 by (*simp add: oddset sum-odds-odd*)
qed (*simp add: oddset sum-odds-even*)
also have $\dots \leq ok\ fun\ 26\ k$
proof –
 have $hgt\ (pseq\ i) \geq 1$ **for** i
 by (*simp add: Suc-leI hgt-gt0*)
 moreover have $hgt\ (pseq\ halted-point) \leq ok\ fun\ 26\ k$
 using *hub pee-le1 height-upper-bound unfolding ok-fun-26-def by blast*
 ultimately show *?thesis*
 by (*simp add: h-def*)
qed
finally have *26*: $(\sum i \in \{..<halted-point\} \setminus \mathcal{D}. h(Suc\ i) - h(i-1)) \leq ok\ fun\ 26\ k$
k .
 with *28 show ?thesis*
 using *that by blast*
qed

proposition *X-7-5*:

assumes $\mu: 0 < \mu < 1$
defines $\mathcal{S} \equiv Step\ class\ \{dboost\ step\}$ **and** $\mathcal{SS} \equiv dboost\ star$
assumes *big*: *Big-X-7-5* $\mu\ l$
shows $card\ (\mathcal{S} \setminus \mathcal{SS}) \leq 3 * \varepsilon\ powr\ (1/4) * k$
proof –
 define \mathcal{D} **where** $\mathcal{D} \equiv Step\ class\ \{dreg\ step\}$
 define \mathcal{R} **where** $\mathcal{R} \equiv Step\ class\ \{red\ step\}$
 define \mathcal{B} **where** $\mathcal{B} \equiv Step\ class\ \{bblue\ step\}$
 define h **where** $h \equiv \lambda i. real\ (hgt\ (pseq\ i))$
 obtain *26*: $(\sum i \in \{..<halted-point\} \setminus \mathcal{D}. h(Suc\ i) - h(i-1)) \leq ok\ fun\ 26\ k$
 and *28*: $ok\ fun\ 28\ k \leq (\sum i \in \mathcal{B}. h(Suc\ i) - h(i-1))$
 using *X-26-and-28 assms(1-3) big*
 unfolding *B-def D-def h-def Big-X-7-5-def by blast*
 have $\mathcal{SS}: \mathcal{SS} = \{i \in \mathcal{S}. h(Suc\ i) - h\ i \leq \varepsilon\ powr\ (-1/4)\}$ **and** $\mathcal{SS} \subseteq \mathcal{S}$
 by (*auto simp: SS-def S-def dboost-star-def h-def*)
 have *in-S*: $h(Suc\ i) - h\ i > \varepsilon\ powr\ (-1/4)$ **if** $i \in \mathcal{S} \setminus \mathcal{SS}$ **for** i
 using *that by (fastforce simp: SS)*
 have *B-limit*: *Big-Blue-4-1* $\mu\ l$
 and *bigR53*: *Big-Red-5-3* $\mu\ l$
 and *16*: $k \geq 16$
 and *ok-fun*: $ok\ fun\ 26\ k - ok\ fun\ 28\ k \leq k$

```

    using big l-le-k by (auto simp: Big-X-7-5-def)
  have [simp]: finite  $\mathcal{R}$  finite  $\mathcal{B}$  finite  $\mathcal{S}$ 
    using finite-components by (auto simp:  $\mathcal{R}$ -def  $\mathcal{B}$ -def  $\mathcal{S}$ -def)
  have [simp]:  $\mathcal{R} \cap \mathcal{S} = \{\}$   $\mathcal{B} \cap (\mathcal{R} \cup \mathcal{S}) = \{\}$ 
    by (auto simp:  $\mathcal{R}$ -def  $\mathcal{S}$ -def  $\mathcal{B}$ -def Step-class-def)

  obtain cardss: card  $\mathcal{SS} \leq$  card  $\mathcal{S}$  card  $(\mathcal{S} \setminus \mathcal{SS}) =$  card  $\mathcal{S} -$  card  $\mathcal{SS}$ 
    by (meson  $\langle \mathcal{SS} \subseteq \mathcal{S} \rangle$   $\langle$ finite  $\mathcal{S} \rangle$  card-Diff-subset card-mono infinite-super)
  have  $(\sum i \in \mathcal{S}. h(\text{Suc } i) - h(i-1)) \geq \varepsilon$  powr  $(-1/4) * \text{card } (\mathcal{S} \setminus \mathcal{SS})$ 
  proof -
    have  $(\sum i \in \mathcal{S} \setminus \mathcal{SS}. h(\text{Suc } i) - h(i-1)) \geq (\sum i \in \mathcal{S} \setminus \mathcal{SS}. \varepsilon \text{ powr } (-1/4))$ 
    proof (rule sum-mono)
      fix i :: nat
      assume i:  $i \in \mathcal{S} \setminus \mathcal{SS}$ 
      with i obtain  $i-1 \in \mathcal{D}$   $i > 0$ 
        using dreg-before-step1 dreg-before-gt0 by (fastforce simp:  $\mathcal{S}$ -def  $\mathcal{D}$ -def
        Step-class-insert-NO-MATCH)
      with i show  $\varepsilon \text{ powr } (-1/4) \leq h(\text{Suc } i) - h(i-1)$ 
        using in-S[of i] Y-6-5-DegreeReg[of i-1] by (simp add:  $\mathcal{D}$ -def h-def)
    qed
    moreover
    have  $(\sum i \in \mathcal{SS}. h(\text{Suc } i) - h(i-1)) \geq 0$ 
    proof (intro sum-nonneg)
      show  $\bigwedge i. i \in \mathcal{SS} \implies 0 \leq h(\text{Suc } i) - h(i-1)$ 
        using Y-6-4-dbooSt  $\mu$  bigR53 by (auto simp: h-def  $\mathcal{SS}$   $\mathcal{S}$ -def hgt-mono)
    qed
    ultimately show ?thesis
      by (simp add: mult.commute sum.subset-diff [OF  $\langle \mathcal{SS} \subseteq \mathcal{S} \rangle$   $\langle$ finite  $\mathcal{S} \rangle$ ])
  qed
  moreover
  have  $(\sum i \in \mathcal{R}. h(\text{Suc } i) - h(i-1)) \geq (\sum i \in \mathcal{R}. -2)$ 
  proof (rule sum-mono)
    fix i :: nat
    assume i:  $i \in \mathcal{R}$ 
    with i obtain  $i-1 \in \mathcal{D}$   $i > 0$ 
      using dreg-before-step1 dreg-before-gt0
      by (fastforce simp:  $\mathcal{R}$ -def  $\mathcal{D}$ -def Step-class-insert-NO-MATCH)
    with i have hgt (pseq (i-1)) - 2  $\leq$  hgt (pseq (Suc i))
      using Y-6-5-Red[of i] 16 Y-6-5-DegreeReg[of i-1]
      by (fastforce simp: algebra-simps  $\mathcal{R}$ -def  $\mathcal{D}$ -def)
    then show  $-2 \leq h(\text{Suc } i) - h(i-1)$ 
      unfolding h-def by linarith
  qed
  ultimately have 27:  $(\sum i \in \mathcal{R} \cup \mathcal{S}. h(\text{Suc } i) - h(i-1)) \geq \varepsilon \text{ powr } (-1/4) * \text{card } (\mathcal{S} \setminus \mathcal{SS}) - 2 * \text{card } \mathcal{R}$ 
    by (simp add: sum.union-disjoint)

  have ok-fun-28  $k + (\varepsilon \text{ powr } (-1/4) * \text{card } (\mathcal{S} \setminus \mathcal{SS}) - 2 * \text{card } \mathcal{R}) \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1)) + (\sum i \in \mathcal{R} \cup \mathcal{S}. h(\text{Suc } i) - h(i-1))$ 

```

using 27 28 **by** *simp*
also have $\dots = (\sum i \in \mathcal{B} \cup (\mathcal{R} \cup \mathcal{S}). h(\text{Suc } i) - h(i-1))$
by (*simp add: sum.union-disjoint*)
also have $\dots = (\sum i \in \{..<\text{halted-point}\} \setminus \mathcal{D}. h(\text{Suc } i) - h(i-1))$
proof –
have $i \in \mathcal{B} \cup (\mathcal{R} \cup \mathcal{S})$ **if** $i < \text{halted-point}$ $i \notin \mathcal{D}$ **for** i
using *that unfolding D-def B-def R-def S-def*
using *Step-class-cases halted-point-minimal by auto*
moreover
have $i \in \{..<\text{halted-point}\} \setminus \mathcal{D}$ **if** $i \in \mathcal{B} \cup (\mathcal{R} \cup \mathcal{S})$ **for** i
using *halted-point-minimal' that by (force simp: D-def B-def R-def S-def Step-class-def)*
ultimately have $\mathcal{B} \cup (\mathcal{R} \cup \mathcal{S}) = \{..<\text{halted-point}\} \setminus \mathcal{D}$
by *auto*
then show *?thesis*
by *simp*
qed
finally have $ok\text{-fun-28 } k + (\varepsilon \text{ powr } (-1/4) * \text{card } (\mathcal{S} \setminus \mathcal{S}\mathcal{S}) - \text{real } (2 * \text{card } \mathcal{R}))$
 $\leq ok\text{-fun-26 } k$
using 26 **by** *simp*
then have $\text{real } (\text{card } (\mathcal{S} \setminus \mathcal{S}\mathcal{S})) \leq (ok\text{-fun-26 } k - ok\text{-fun-28 } k + 2 * \text{card } \mathcal{R}) * \varepsilon \text{ powr } (1/4)$
using *eps-gt0 by (simp add: powr-minus field-simps del: div-add div-mult-self3)*
moreover have $\text{card } \mathcal{R} < k$
using *red-step-limit μ unfolding R-def by blast*
ultimately have $\text{card } (\mathcal{S} \setminus \mathcal{S}\mathcal{S}) \leq (k + 2 * k) * \varepsilon \text{ powr } (1/4)$
by (*smt (verit, best) of-nat-add mult-2 mult-right-mono nat-less-real-le ok-fun powr-ge-zero*)
then show *?thesis*
by (*simp add: algebra-simps*)
qed
end

6.5 Lemma 7.4

definition

Big-X-7-4 $\equiv \lambda \mu l. \text{Big-X-7-5 } \mu l \wedge \text{Big-Red-5-3 } \mu l$

establishing the size requirements for 7.4

lemma *Big-X-7-4*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0 .. \mu 1\} \longrightarrow \text{Big-X-7-4 } \mu l$

using *assms Big-X-7-5 Big-Red-5-3*

unfolding *Big-X-7-4-def*

by (*simp add: eventually-conj-iff all-imp-conj-distrib*)

definition *ok-fun-74* $\equiv \lambda k. -6 * \text{eps } k \text{ powr } (1/4) * k * \ln k / \ln 2$

lemma *ok-fun-74*: $ok-fun-74 \in o(\text{real})$
unfolding *ok-fun-74-def eps-def* **by** *real-asymp*

context *Book*
begin

lemma *X-7-4*:
assumes *big*: *Big-X-7-4* μ *l*
defines $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
shows $(\prod_{i \in \mathcal{S}} \text{card } (Xseq (\text{Suc } i)) / \text{card } (Xseq i)) \geq 2 \text{ powr } ok-fun-74 \ k * \text{bigbeta} \wedge \text{card } \mathcal{S}$
proof –
define \mathcal{SS} **where** $\mathcal{SS} \equiv \text{dboost-star}$
then have *big53*: *Big-Red-5-3* μ *l* **and** *X75*: $\text{card } (\mathcal{S} \setminus \mathcal{SS}) \leq 3 * \varepsilon \text{ powr } (1/4)$
 $* k$
using $\mu 01$ *big* **by** (*auto simp: Big-X-7-4-def X-7-5 S-def SS-def*)
then have *R53*: $\text{pseq } (\text{Suc } i) \geq \text{pseq } i \wedge \text{beta } i \geq 1 / (\text{real } k)^2$ **and** *beta-gt0*: $0 < \text{beta } i$
if $i \in \mathcal{S}$ **for** i
using *that Red-5-3 beta-gt0* **by** (*auto simp: S-def*)
have *bigbeta01*: $\text{bigbeta} \in \{0 < .. < 1\}$
using *big53* *assms bigbeta-gt0 bigbeta-less1* **by force**
have $\mathcal{SS} \subseteq \mathcal{S}$
unfolding *SS-def S-def dboost-star-def* **by auto**
then obtain [*simp*]: *finite S finite SS*
by (*simp add: SS-def S-def finite-dboost-star*)
have *card-SSS*: $\text{card } \mathcal{SS} \leq \text{card } \mathcal{S}$
by (*metis SS-def S-def <finite S> card-mono dboost-star-subset*)
have β : $\text{beta } i = \text{card } (Xseq (\text{Suc } i)) / \text{card } (Xseq i)$ **if** $i \in \mathcal{S}$ **for** i
proof –
have $Xseq (\text{Suc } i) = \text{Neighbours Blue } (cvx i) \cap Xseq i$
using *that unfolding S-def*
by (*auto simp: step-kind-defs next-state-def split: prod.split*)
then show *?thesis*
by (*force simp: beta-eq*)

qed
then have $*$: $(\prod_{i \in \mathcal{S}} \text{card } (Xseq (\text{Suc } i)) / \text{card } (Xseq i)) = (\prod_{i \in \mathcal{S}} \text{beta } i)$
by force
have *prod-beta-gt0*: $\text{prod } (\text{beta}) \ S' > 0$ **if** $S' \subseteq \mathcal{S}$ **for** S'
using *beta-gt0 that*
by (*force simp: beta-ge0 intro: prod-pos*)
– bounding the immoderate steps
have $(\prod_{i \in \mathcal{S} \setminus \mathcal{SS}} 1 / \text{beta } i) \leq (\prod_{i \in \mathcal{S} \setminus \mathcal{SS}} \text{real } k \wedge 2)$
proof (*rule prod-mono*)
fix i
assume i : $i \in \mathcal{S} \setminus \mathcal{SS}$
with *R53 kn0 beta-ge0 [of i]* **show** $0 \leq 1 / \text{beta } i \wedge 1 / \text{beta } i \leq (\text{real } k)^2$
by (*force simp: R53 divide-simps mult.commute*)

qed

```

then have ( $\prod_{i \in \mathcal{S} \setminus \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ )  $\leq$  real  $k \wedge (2 * \text{card}(\mathcal{S} \setminus \mathcal{S}\mathcal{S}))$ 
  by (simp add: power-mult)
also have ... = real  $k \text{ powr } (2 * \text{card}(\mathcal{S} \setminus \mathcal{S}\mathcal{S}))$ 
  by (metis kn0 of-nat-0-less-iff powr-realpow)
also have ...  $\leq k \text{ powr } (2 * 3 * \varepsilon \text{ powr } (1/4) * k)$ 
  using X75 kn0 by (intro powr-mono; linarith)
also have ...  $\leq \text{exp } (6 * \varepsilon \text{ powr } (1/4) * k * \ln k)$ 
  by (simp add: powr-def)
also have ... = 2 powr -ok-fun-74 k
  by (simp add: ok-fun-74-def powr-def)
finally have ( $\prod_{i \in \mathcal{S} \setminus \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ )  $\leq 2 \text{ powr } -\text{ok-fun-74 } k$  .
then have A: ( $\prod_{i \in \mathcal{S} \setminus \mathcal{S}\mathcal{S}} \text{beta } i$ )  $\geq 2 \text{ powr } \text{ok-fun-74 } k$ 
  using prod-beta-gt0[of  $\mathcal{S} \setminus \mathcal{S}\mathcal{S}$ ]
  by (simp add: powr-minus prod-dividef mult.commute divide-simps)
— bounding the moderate steps
have ( $\prod_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ )  $\leq \text{bigbeta powr } (- (\text{card } \mathcal{S}\mathcal{S}))$ 
proof (cases  $\mathcal{S}\mathcal{S} = \{\}$ )
  case True
  with bigbeta01 show ?thesis
  by fastforce
next
  case False
  then have  $\text{card } \mathcal{S}\mathcal{S} > 0$ 
  using <finite  $\mathcal{S}\mathcal{S}$ > card-0-eq by blast
  have ( $\prod_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ ) powr (1 / card  $\mathcal{S}\mathcal{S}$ )  $\leq (\sum_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i / \text{card } \mathcal{S}\mathcal{S})$ 
proof (rule arith-geom-mean [OF <finite  $\mathcal{S}\mathcal{S}$ > < $\mathcal{S}\mathcal{S} \neq \{\}$ >])
  show  $\bigwedge i. i \in \mathcal{S}\mathcal{S} \implies 0 \leq 1 / \text{beta } i$ 
  by (simp add: beta-ge0)
qed
then have (( $\prod_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ ) powr (1 / card  $\mathcal{S}\mathcal{S}$ )) powr (card  $\mathcal{S}\mathcal{S}$ )
   $\leq (\sum_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i / \text{card } \mathcal{S}\mathcal{S})$  powr (card  $\mathcal{S}\mathcal{S}$ )
  using powr-mono2 by auto
with < $\mathcal{S}\mathcal{S} \neq \{\}$ >
have ( $\prod_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i$ )  $\leq (\sum_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i / \text{card } \mathcal{S}\mathcal{S})$  powr (card  $\mathcal{S}\mathcal{S}$ )
  by (simp add: powr-powr beta-ge0 prod-nonneg)
also have ...  $\leq (1 / (\text{card } \mathcal{S}\mathcal{S}) * (\sum_{i \in \mathcal{S}\mathcal{S}} 1 / \text{beta } i))$  powr (card  $\mathcal{S}\mathcal{S}$ )
  using <card  $\mathcal{S}\mathcal{S} > 0$ > by (simp add: field-simps sum-divide-distrib)
also have ...  $\leq \text{bigbeta powr } (- (\text{card } \mathcal{S}\mathcal{S}))$ 
  using < $\mathcal{S}\mathcal{S} \neq \{\}$ > <card  $\mathcal{S}\mathcal{S} > 0$ >
  by (simp add: bigbeta-def field-simps powr-minus powr-divide beta-ge0 sum-nonneg flip:  $\mathcal{S}\mathcal{S}$ -def)
finally show ?thesis .
qed
then have B: ( $\prod_{i \in \mathcal{S}\mathcal{S}} \text{beta } i$ )  $\geq \text{bigbeta powr } (\text{card } \mathcal{S}\mathcal{S})$ 
  using < $\mathcal{S}\mathcal{S} \subseteq \mathcal{S}$ > prod-beta-gt0[of  $\mathcal{S}\mathcal{S}$ ] bigbeta01
  by (simp add: powr-minus prod-dividef mult.commute divide-simps)
have 2 powr ok-fun-74 k * bigbeta powr card  $\mathcal{S} \leq 2 \text{ powr } \text{ok-fun-74 } k * \text{bigbeta powr card } \mathcal{S}\mathcal{S}$ 

```

using *bigbeta01 big53 card-SSS* **by** (*simp add: powr-mono'*)
also have $\dots \leq (\prod_{i \in \mathcal{S} \setminus \mathcal{SS}} \text{beta } i) * (\prod_{i \in \mathcal{SS}} \text{beta } i)$
using *beta-ge0* **by** (*intro mult-mono A B*) (*auto simp: prod-nonneg*)
also have $\dots = (\prod_{i \in \mathcal{S}} \text{beta } i)$
by (*metis* $\langle \mathcal{SS} \subseteq \mathcal{S} \rangle \langle \text{finite } \mathcal{S} \rangle \text{prod.subset-diff}$)
finally have $2^{\text{powr ok-fun-74 } k} * \text{bigbeta powr real } (\text{card } \mathcal{S}) \leq \text{prod } (\text{beta}) \mathcal{S}$.
with *bigbeta01* **show** *?thesis*
by (*simp add: * powr-realpow*)
qed

6.6 Observation 7.7

lemma *X-7-7:*

assumes $i: i \in \text{Step-class } \{\text{dreg-step}\}$
defines $q \equiv \varepsilon^{\text{powr } (-1/2)} * \text{alpha } (\text{hgt } (\text{pseq } i))$
shows $\text{pseq } (\text{Suc } i) - \text{pseq } i \geq \text{card } (X\text{seq } i \setminus X\text{seq } (\text{Suc } i)) / \text{card } (X\text{seq } (\text{Suc } i)) * q \wedge \text{card } (X\text{seq } (\text{Suc } i)) > 0$
proof –
have *finX: finite (Xseq i) for i*
using *finite-Xseq* **by** *blast*
define *Y* **where** $Y \equiv Y\text{seq}$
have $X\text{seq } (\text{Suc } i) = \{x \in X\text{seq } i. \text{red-dense } (Y \ i) (\text{red-density } (X\text{seq } i) (Y \ i))\}$
and $Y: Y (\text{Suc } i) = Y \ i$
using *i*
by (*simp-all add: step-kind-defs next-state-def X-degree-reg-def degree-reg-def Y-def split: if-split-asm prod.split-asm*)
then have $X\text{seq } (\text{Suc } i) = \{x \in X\text{seq } i. \text{card } (\text{Neighbours Red } x \cap Y \ i) \geq (\text{pseq } i - q) * \text{card } (Y \ i)\}$
by (*simp add: red-dense-def q-def pseq-def Y-def*)
have $X\text{sub}[simp]: X\text{seq } (\text{Suc } i) \subseteq X\text{seq } i$
using *Xseq-Suc-subset* **by** *blast*
then have *card-le: card (Xseq (Suc i)) ≤ card (Xseq i)*
by (*simp add: card-mono finX*)
have $[simp]: \text{disjnt } (X\text{seq } i) (Y \ i)$
using *Xseq-Yseq-disjnt Y-def* **by** *blast*
have $X\text{non0: card } (X\text{seq } i) > 0$ **and** $Y\text{non0: card } (Y \ i) > 0$
using *i* **by** (*simp-all add: Y-def Xseq-gt0 Yseq-gt0 Step-class-def*)
have $\text{alpha } (\text{hgt } (\text{pseq } i)) > 0$
by (*simp add: alpha-gt0 kn0 hgt-gt0*)
with *kn0* **have** $q > 0$
by (*smt (verit) q-def eps-gt0 mult-pos-pos powr-gt-zero*)
have $X\text{dif: } X\text{seq } i \setminus X\text{seq } (\text{Suc } i) = \{x \in X\text{seq } i. \text{card } (\text{Neighbours Red } x \cap Y \ i) < (\text{pseq } i - q) * \text{card } (Y \ i)\}$
using *Xseq* **by** *force*
have $\text{disYX: disjnt } (Y \ i) (X\text{seq } i \setminus X\text{seq } (\text{Suc } i))$
by (*metis Diff-subset* $\langle \text{disjnt } (X\text{seq } i) (Y \ i) \rangle \text{disjnt-subset2 disjnt-sym}$)
have $\text{edge-card Red } (Y \ i) (X\text{seq } i \setminus X\text{seq } (\text{Suc } i)) = (\sum x \in X\text{seq } i \setminus X\text{seq } (\text{Suc } i). \text{real } (\text{card } (\text{Neighbours Red } x \cap Y \ i)))$

using *edge-card-eq-sum-Neighbours* [*OF - - disYX*] *finX Red-E* **by** *simp*
also have $\dots \leq (\sum x \in Xseq\ i \setminus Xseq\ (Suc\ i). (pseq\ i - q) * card\ (Y\ i))$
by (*smt (verit, del-insts) Xdif mem-Collect-eq sum-mono*)
finally have $A: edge-card\ Red\ (Xseq\ i \setminus Xseq\ (Suc\ i))\ (Y\ i) \leq card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) * (pseq\ i - q) * card\ (Y\ i)$
by (*simp add: edge-card-commute*)
then have *False* **if** $Xseq\ (Suc\ i) = \{\}$
using $\langle q > 0 \rangle$ *Xnon0 Ynon0* **that** **by** (*simp add: edge-card-eq-pee Y-def mult-le-0-iff*)
then have *XSnon0*: $card\ (Xseq\ (Suc\ i)) > 0$
using *card-gt-0-iff finX* **by** *blast*
have $pseq\ i * card\ (Xseq\ i) * real\ (card\ (Y\ i)) - edge-card\ Red\ (Xseq\ (Suc\ i))\ (Y\ i)$
 $\leq card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) * (pseq\ i - q) * card\ (Y\ i)$
by (*metis A edge-card-eq-pee edge-card-mono Y-def Xsub <disjnt (Xseq i) (Y i)> edge-card-diff finX of-nat-diff*)
moreover have $real\ (card\ (Xseq\ (Suc\ i))) \leq real\ (card\ (Xseq\ i))$
using *Xsub* **by** (*simp add: card-le*)
ultimately have $\S: edge-card\ Red\ (Xseq\ (Suc\ i))\ (Y\ i) \geq pseq\ i * card\ (Xseq\ (Suc\ i)) * card\ (Y\ i) + card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) * q * card\ (Y\ i)$
using *Xnon0*
by (*smt (verit, del-insts) Xsub card-Diff-subset card-gt-0-iff card-le left-diff-distrib finite-subset mult-of-nat-commute of-nat-diff*)
have $edge-card\ Red\ (Xseq\ (Suc\ i))\ (Y\ i) / (card\ (Xseq\ (Suc\ i)) * card\ (Y\ i)) \geq pseq\ i + card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) * q / card\ (Xseq\ (Suc\ i))$
using *divide-right-mono* [*OF* \S , *of card (Xseq (Suc i)) * card (Y i)*] *XSnon0 Ynon0*
by (*simp add: add-divide-distrib split: if-split-asm*)
moreover have $pseq\ (Suc\ i) = real\ (edge-card\ Red\ (Xseq\ (Suc\ i))\ (Y\ i)) / (real\ (card\ (Y\ i)) * real\ (card\ (Xseq\ (Suc\ i))))$
using *Y* **by** (*simp add: pseq-def gen-density-def Y-def*)
ultimately show *?thesis*
by (*simp add: algebra-simps XSnon0*)

qed

end

6.7 Lemma 7.8

definition *Big-X-7-8* $\equiv \lambda k. k \geq 2 \wedge eps\ k\ powr\ (1/2) / k \geq 2 / k^2$

lemma *Big-X-7-8*: $\forall^\infty k. Big-X-7-8\ k$

unfolding *eps-def Big-X-7-8-def eventually-conj-iff eps-def*
by (*intro conjI; real-asymp*)

lemma (in *Book*) *X-7-8*:

assumes *big: Big-X-7-8 k*

and $i \in Step-class\ \{dreg-step\}$

shows $card\ (Xseq\ (Suc\ i)) \geq card\ (Xseq\ i) / k^2$

proof –

```

define  $q$  where  $q \equiv \varepsilon \text{ powr } (-1/2) * \text{alpha } (\text{hgt } (\text{pseq } i))$ 
have  $k > 0 \langle k \geq 2 \rangle$  using big by (auto simp: Big-X-7-8-def)
have  $2 / k^2 \leq \varepsilon \text{ powr } (1/2) / k$ 
  using big by (auto simp: Big-X-7-8-def)
also have  $\dots \leq q$ 
  using kn0 eps-gt0 Red-5-7a [of pseq i]
  by (simp add: q-def powr-minus divide-simps flip: powr-add)
finally have  $q \geq 2 / k^2$  .
define  $Y$  where  $Y \equiv Y\text{seq}$ 
have  $X\text{seq } (\text{Suc } i) = \{x \in X\text{seq } i. \text{red-dense } (Y \ i) (\text{red-density } (X\text{seq } i) (Y \ i))\}$ 
and  $Y: Y (\text{Suc } i) = Y \ i$ 
  using i
  by (simp-all add: step-kind-defs next-state-def X-degree-reg-def degree-reg-def
    Y-def split: if-split-asm prod.split-asm)
have  $X\text{Snon0}: \text{card } (X\text{seq } (\text{Suc } i)) > 0$ 
  using X-7-7 kn0 assms by simp
have  $\text{fin}X: \text{finite } (X\text{seq } i)$  for  $i$ 
  using finite-Xseq by blast
have  $X\text{sub}[simp]: X\text{seq } (\text{Suc } i) \subseteq X\text{seq } i$ 
  using Xseq-Suc-subset by blast
then have  $\text{card-le}: \text{card } (X\text{seq } (\text{Suc } i)) \leq \text{card } (X\text{seq } i)$ 
  by (simp add: card-mono finX)
have  $2 \leq (\text{real } k)^2$ 
  by (metis of-nat-numeral <2 ≤ k> of-nat-power-le-of-nat-cancel-iff self-le-ge2-pow)
then have  $2: 2 / (\text{real } k^2 + 2) \geq 1 / k^2$ 
  by (simp add: divide-simps)
have  $q * \text{card } (X\text{seq } i \setminus X\text{seq } (\text{Suc } i)) / \text{card } (X\text{seq } (\text{Suc } i)) \leq \text{pseq } (\text{Suc } i) -$ 
 $\text{pseq } i$ 
  using X-7-7 μ01 kn0 assms by (simp add: q-def mult-of-nat-commute)
also have  $\dots \leq 1$ 
  by (smt (verit) pee-ge0 pee-le1)
finally have  $q * \text{card } (X\text{seq } i \setminus X\text{seq } (\text{Suc } i)) \leq \text{card } (X\text{seq } (\text{Suc } i))$ 
  using XSnon0 by auto
with  $q \geq 2 / k^2$  have  $\text{card } (X\text{seq } (\text{Suc } i)) \geq (2 / k^2) * \text{card } (X\text{seq } i \setminus X\text{seq } (\text{Suc } i))$ 
  by (smt (verit, best) mult-right-mono of-nat-0-le-iff)
then have  $\text{card } (X\text{seq } (\text{Suc } i)) * (1 + 2/k^2) \geq (2/k^2) * \text{card } (X\text{seq } i)$ 
  by (simp add: card-Diff-subset finX card-le diff-divide-distrib field-simps)
then have  $\text{card } (X\text{seq } (\text{Suc } i)) \geq (2/(\text{real } k^2 + 2)) * \text{card } (X\text{seq } i)$ 
  using kn0 add-nonneg-nonneg[of real k^2 2]
  by (simp del: add-nonneg-nonneg add: divide-simps split: if-split-asm)
then show ?thesis
  using mult-right-mono [OF 2, of card (Xseq i)] by simp
qed

```

6.8 Lemma 7.9

definition *Big-X-7-9* $\equiv \lambda k. ((1 + \text{eps } k) \text{ powr } (\text{eps } k \text{ powr } (-1/4) + 1) - 1) / \text{eps } k \leq 2 * \text{eps } k \text{ powr } (-1/4)$

$\wedge k \geq 2 \wedge \text{eps } k \text{ powr } (1/2) / k \geq 2 / k^2$

lemma *Big-X-7-9*: $\forall^\infty k. \text{Big-X-7-9 } k$
unfolding *eps-def Big-X-7-9-def eventually-conj-iff eps-def*
by (*intro conjI; real-asymp*)

lemma *one-plus-powr-le*:

fixes *p::real*

assumes $0 \leq p \leq 1 \ x \geq 0$

shows $(1+x) \text{ powr } p - 1 \leq x * p$

proof –

define *f* **where** $f \equiv \lambda x. x * p - ((1+x) \text{ powr } p - 1)$

have $0 \leq f 0$

by (*simp add: f-def*)

also have $\dots \leq f x$

proof (*intro DERIV-nonneg-imp-nondecreasing[of concl: f] exI conjI assms*)

fix *y::real*

assume $y: 0 \leq y \leq x$

show (*f has-real-derivative p - (1+y)powr (p-1) * p*) (*at y*)

unfolding *f-def using assms y by* (*intro derivative-eq-intros | simp*)**+**

show $p - (1+y) \text{ powr } (p-1) * p \geq 0$

using *y assms less-eq-real-def powr-less-one by fastforce*

qed

finally show *?thesis*

by (*simp add: f-def*)

qed

lemma (*in Book*) *X-7-9*:

assumes *i: i ∈ Step-class {dreg-step} and big: Big-X-7-9 k*

defines *hp ≡ λi. hgt (pseq i)*

assumes $pseq\ i \geq p0$ **and** *hgt: hp (Suc i) ≤ hp i + ε powr (-1/4)*

shows $\text{card } (Xseq\ (Suc\ i)) \geq (1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card } (Xseq\ i)$

proof –

have *k: k ≥ 2 ε powr (1/2) / k ≥ 2 / k^2*

using *big by* (*auto simp: Big-X-7-9-def*)

let *?q = ε powr (-1/2) * alpha (hp i)*

have *k > 0 using k by auto*

have *Xsub[simp]: Xseq (Suc i) ⊆ Xseq i*

using *Xseq-Suc-subset by blast*

have *finX: finite (Xseq i) for i*

using *finite-Xseq by blast*

then have *card-le: card (Xseq (Suc i)) ≤ card (Xseq i)*

by (*simp add: card-mono finX*)

have *XNon0: card (Xseq (Suc i)) > 0*

using *X-7-7 <0 <k> i by blast*

have $\text{card } (Xseq\ i \setminus Xseq\ (Suc\ i)) / \text{card } (Xseq\ (Suc\ i)) * ?q \leq pseq\ (Suc\ i) - pseq\ i$

using *X-7-7 i k hp-def by auto*

also have $\dots \leq 2 * \varepsilon \text{ powr } (-1/4) * \text{alpha } (hp\ i)$

proof –
have *hgt-le*: $hp\ i \leq hp\ (Suc\ i)$
using *Y-6-5-DegreeReg* $\langle 0 < k \rangle\ i\ hp-def$ **by** *blast*
have *A*: $pseq\ (Suc\ i) \leq qfun\ (hp\ (Suc\ i))$
by (*simp add*: $\langle 0 < k \rangle\ hp-def\ hgt-works$)
have *B*: $qfun\ (hp\ i - 1) \leq pseq\ i$
using *hgt-Least* [*of hp i - 1 pseq i*] $\langle pseq\ i \geq p0 \rangle$ **by** (*force simp*: *hp-def*)
have $pseq\ (Suc\ i) - pseq\ i \leq qfun\ (hp\ (Suc\ i)) - qfun\ (hp\ i - 1)$
using *A B* **by** *auto*
also have $\dots = ((1 + \varepsilon) ^ (Suc\ (hp\ i - 1 + hp\ (Suc\ i)) - hp\ i) - (1 + \varepsilon) ^ (hp\ i - 1)) / k$
using *kn0 eps-gt0 hgt-le* $\langle pseq\ i \geq p0 \rangle$ *hgt-gt0* [*of k*]
by (*simp add*: *hp-def qfun-eq Suc-diff-eq-diff-pred hgt-gt0 diff-divide-distrib*)
also have $\dots = alpha\ (hp\ i) / \varepsilon * ((1 + \varepsilon) ^ (1 + hp\ (Suc\ i) - hp\ i) - 1)$
using *kn0 hgt-le hgt-gt0*
by (*simp add*: *hp-def alpha-eq right-diff-distrib flip: diff-divide-distrib power-add*)
also have $\dots \leq 2 * \varepsilon\ powr\ (-1/4) * alpha\ (hp\ i)$
proof –
have $((1 + \varepsilon) ^ (1 + hp\ (Suc\ i) - hp\ i) - 1) / \varepsilon \leq ((1 + \varepsilon) powr\ (\varepsilon\ powr\ (-1/4) + 1) - 1) / \varepsilon$
using *hgt eps-ge0 hgt-le powr-mono-both* **by** (*force simp flip: powr-realpow intro: divide-right-mono*)
also have $\dots \leq 2 * \varepsilon\ powr\ (-1/4)$
using *big* **by** (*meson Big-X-7-9-def*)
finally have $*$: $((1 + \varepsilon) ^ (1 + hp\ (Suc\ i) - hp\ i) - 1) / \varepsilon \leq 2 * \varepsilon\ powr\ (-1/4)$.
show *?thesis*
using *mult-left-mono* [*OF **, *of alpha (hp i)*]
by (*smt (verit) alpha-ge0 mult commute times-divide-eq-right*)
qed
finally show *?thesis* .
qed
finally have *29*: $card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) / card\ (Xseq\ (Suc\ i)) * ?q \leq 2 * \varepsilon\ powr\ (-1/4) * alpha\ (hp\ i)$.
moreover have $alpha\ (hp\ i) > 0$
unfolding *hp-def*
by (*smt (verit, ccfv-SIG) eps-gt0* $\langle 0 < k \rangle$ *alpha-ge divide-le-0-iff hgt-gt0 of-nat-0-less-iff*)
ultimately have $card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) / card\ (Xseq\ (Suc\ i)) * \varepsilon\ powr\ (-1/2) \leq 2 * \varepsilon\ powr\ (-1/4)$
using *mult-le-cancel-right* **by** *fastforce*
then have $card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) / card\ (Xseq\ (Suc\ i)) \leq 2 * \varepsilon\ powr\ (-1/4) * \varepsilon\ powr\ (1/2)$
using $\langle 0 < k \rangle$ *eps-gt0*
by (*force simp: powr-minus divide-simps mult commute mult-less-0-iff*)
then have $card\ (Xseq\ i \setminus Xseq\ (Suc\ i)) \leq 2 * \varepsilon\ powr\ (1/4) * card\ (Xseq\ (Suc\ i))$
using *XNon0* **by** (*simp add: field-simps flip: powr-add*)
also have $\dots \leq 2 * \varepsilon\ powr\ (1/4) * card\ (Xseq\ i)$

by (*simp add: card-le mult-mono'*)
 finally show ?thesis
 by (*simp add: card-Diff-subset finX card-le algebra-simps*)
 qed

6.9 Lemma 7.10

definition *Big-X-7-10* $\equiv \lambda \mu l. \text{Big-X-7-5 } \mu l \wedge \text{Big-Red-5-3 } \mu l$

establishing the size requirements for 7.10

lemma *Big-X-7-10*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-X-7-10 } \mu l$

using *Big-X-7-10-def Big-X-7-4 Big-X-7-4-def assms* by force

lemma (in *Book*) *X-7-10*:

defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$

defines $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$

defines $h \equiv \lambda i. \text{real } (\text{hgt } (\text{pseq } i))$

defines $C \equiv \{i. h \ i \geq h \ (i-1) + \varepsilon \ \text{powr } (-1/4)\}$

assumes *big: Big-X-7-10* μl

shows $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) \leq 3 * \varepsilon \ \text{powr } (1/4) * k$

proof –

define \mathcal{D} where $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$

define \mathcal{B} where $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$

have *hub: Big-height-upper-bound* k

and *16: k* ≥ 16

and *ok-le-k: ok-fun-26* $k - \text{ok-fun-28 } k \leq k$

and *bigR53: Big-Red-5-3* μl

using *big l-le-k* by (*auto simp: Big-X-7-5-def Big-X-7-10-def*)

have $\mathcal{R} \cup \mathcal{S} \subseteq \{.. < \text{halted-point}\} \setminus \mathcal{D} \setminus \mathcal{B}$ and *BmD: B* $\subseteq \{.. < \text{halted-point}\} \setminus \mathcal{D}$

using *halted-point-minimal'*

by (*fastforce simp: R-def S-def D-def B-def Step-class-def*) +

then have *RS-eq: R* $\cup \mathcal{S} = \{.. < \text{halted-point}\} \setminus \mathcal{D} - \mathcal{B}$

using *halted-point-minimal Step-class-cases* by (*auto simp: R-def S-def D-def B-def*)

obtain *26: (* $\sum i \in \{.. < \text{halted-point}\} \setminus \mathcal{D}. h \ (\text{Suc } i) - h \ (i-1)) \leq \text{ok-fun-26 } k$

and *28: ok-fun-28* $k \leq (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$

using *X-26-and-28 big unfolding B-def D-def h-def Big-X-7-10-def* by blast

have $(\sum i \in \mathcal{R} \cup \mathcal{S}. h \ (\text{Suc } i) - h \ (i-1)) = (\sum i \in \{.. < \text{halted-point}\} \setminus \mathcal{D}. h \ (\text{Suc } i) - h \ (i-1)) - (\sum i \in \mathcal{B}. h(\text{Suc } i) - h(i-1))$

unfolding *RS-eq* by (*intro sum-diff BmD*) auto

also have $\dots \leq \text{ok-fun-26 } k - \text{ok-fun-28 } k$

using *26 28* by *linarith*

finally have $*$: $(\sum i \in \mathcal{R} \cup \mathcal{S}. h \ (\text{Suc } i) - h \ (i-1)) \leq \text{ok-fun-26 } k - \text{ok-fun-28 } k$

.

have [*simp*]: *finite R finite S*

using *finite-components* by (*auto simp: R-def S-def*)

```

have h-ge-0-if-S: h(Suc i) - h(i-1) ≥ 0 if i ∈ S for i
proof -
  have *: hgt (pseq i) ≤ hgt (pseq (Suc i))
    using bigR53 Y-6-5-dbooSt that unfolding S-def by blast
  obtain i-1 ∈ D i>0
    using that ⟨i∈S⟩ dreg-before-step1[of i] dreg-before-gt0[of i]
    by (force simp: S-def D-def Step-class-insert-NO-MATCH)
  then have hgt (pseq (i-1)) ≤ hgt (pseq i)
    using that kn0 by (metis Suc-diff-1 Y-6-5-DegreeReg D-def)
  with * show 0 ≤ h(Suc i) - h(i-1)
    using kn0 unfolding h-def by linarith
qed

have card ((RUS) ∩ C) * ε powr (-1/4) + real (card R) * (-2)
  = (∑ i ∈ RUS. if i∈C then ε powr (-1/4) else 0) + (∑ i ∈ RUS. if i∈R
then -2 else 0)
  by (simp add: Int-commute Int-left-commute flip: sum.inter-restrict)
also have ... = (∑ i ∈ RUS. (if i∈C then ε powr (-1/4) else 0) + (if i∈R
then -2 else 0))
  by (simp add: sum.distrib)
also have ... ≤ (∑ i ∈ RUS. h(Suc i) - h(i-1))
proof (rule sum-mono)
  fix i :: nat
  assume i: i ∈ RUS
  with i dreg-before-step1 dreg-before-gt0 have D: i-1 ∈ D i>0
    by (force simp: S-def R-def D-def dreg-before-step Step-class-def)+
  then have *: hgt (pseq (i-1)) ≤ hgt (pseq i)
    by (metis Suc-diff-1 Y-6-5-DegreeReg D-def)
  show (if i∈C then ε powr (-1/4) else 0) + (if i∈R then -2 else 0) ≤ h (Suc
i) - h (i-1)
  proof (cases i∈R)
    case True
    then have h i - 2 ≤ h (Suc i)
      using Y-6-5-Red[of i] 16 by (force simp: algebra-simps R-def h-def)
    with * True show ?thesis
      by (simp add: h-def C-def)
    next
    case False
    with i have i∈S by blast
    show ?thesis
    proof (cases i∈C)
      case True
      then have h (i - Suc 0) + ε powr (-1/4) ≤ h i
        by (simp add: C-def)
      then show ?thesis
        using * i ⟨i∉R⟩ kn0 bigR53 Y-6-5-dbooSt by (force simp: h-def S-def)
    qed (use ⟨i∉R⟩ ⟨i∈S⟩ h-ge-0-if-S in auto)
  qed
qed

```

also have $\dots \leq k$
using $*\text{ok-le-k}$
by *linarith*
finally have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) * \varepsilon \text{ powr } (-1/4) - 2 * \text{card } \mathcal{R} \leq k$
by *linarith*
moreover have $\text{card } \mathcal{R} \leq k$
by (*metis* $\mathcal{R}\text{-def nless-le red-step-limit}$)
ultimately have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) * \varepsilon \text{ powr } (-1/4) \leq 3 * k$
by *linarith*
with *eps-gt0* **show** *?thesis*
by (*simp add: powr-minus divide-simps mult.commute split: if-split-asm*)
qed

6.10 Lemma 7.11

definition *Big-X-7-11-inequalities* $\equiv \lambda k.$

$$\begin{aligned}
& \text{eps } k * \text{eps } k \text{ powr } (-1/4) \leq (1 + \text{eps } k) \wedge (2 * \text{nat } \lfloor \text{eps } k \text{ powr } \\
& (-1/4) \rfloor) - 1 \\
& \wedge k \geq 2 * \text{eps } k \text{ powr } (-1/2) * k \text{ powr } (3/4) \\
& \wedge ((1 + \text{eps } k) * (1 + \text{eps } k) \text{ powr } (2 * \text{eps } k \text{ powr } (-1/4))) \leq 2 \\
& \wedge (1 + \text{eps } k) \wedge (\text{nat } \lfloor 2 * \text{eps } k \text{ powr } (-1/4) \rfloor + \text{nat } \lfloor 2 * \text{eps } k \text{ powr } \\
& (-1/2) \rfloor - 1) \leq 2
\end{aligned}$$

definition *Big-X-7-11* \equiv

$$\begin{aligned}
& \lambda \mu l. \text{Big-X-7-5 } \mu l \wedge \text{Big-Red-5-3 } \mu l \wedge \text{Big-Y-6-5-Bblue } l \\
& \wedge (\forall k. l \leq k \longrightarrow \text{Big-X-7-11-inequalities } k)
\end{aligned}$$

establishing the size requirements for 7.11

lemma *Big-X-7-11*:

assumes $0 < \mu 0 \ \mu 1 < 1$
shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-X-7-11 } \mu l$
using *assms Big-Red-5-3 Big-X-7-5 Big-Y-6-5-Bblue*
unfolding *Big-X-7-11-def Big-X-7-11-inequalities-def eventually-conj-iff all-imp-conj-distrib*
eps-def
apply (*simp add: eventually-conj-iff all-imp-conj-distrib*)
apply (*intro conjI strip eventually-all-geI0 eventually-all-ge-at-top; real-asymp*)
done

lemma (*in Book*) *X-7-11*:

defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$
defines $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
defines $C \equiv \{i. \text{pseq } i \geq \text{pseq } (i-1) + \varepsilon \text{ powr } (-1/4) * \text{alpha } 1 \wedge \text{pseq } (i-1) \leq p0\}$

assumes *big: Big-X-7-11* μl
shows $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) \leq 4 * \varepsilon \text{ powr } (1/4) * k$

proof –

define *qstar* **where** $qstar \equiv p0 + \varepsilon \text{ powr } (-1/4) * \text{alpha } 1$
define *pstar* **where** $pstar \equiv \lambda i. \text{min } (\text{pseq } i) \text{ qstar}$
define \mathcal{D} **where** $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$
define \mathcal{B} **where** $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$

have *big-x75*: *Big-X-7-5* μ *l*
and *711*: $\varepsilon * \varepsilon \text{ powr } (-1/4) \leq (1 + \varepsilon) \wedge (2 * \text{nat } \lfloor \varepsilon \text{ powr } (-1/4) \rfloor) - 1$
and *big34*: $k \geq 2 * \varepsilon \text{ powr } (-1/2) * k \text{ powr } (3/4)$
and *le2*: $((1 + \varepsilon) * (1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/4))) \leq 2$
 $(1 + \varepsilon) \wedge (\text{nat } \lfloor 2 * \varepsilon \text{ powr } (-1/4) \rfloor + \text{nat } \lfloor 2 * \varepsilon \text{ powr } (-1/2) \rfloor) - 1$
 ≤ 2
and *bigY65B*: *Big-Y-6-5-Bblue* *l*
and *R53*: $\bigwedge i. i \in \mathcal{S} \implies \text{pseq } (\text{Suc } i) \geq \text{pseq } i$
using *big l-le-k*
by (*auto simp: Red-5-3 Big-X-7-11-def Big-X-7-11-inequalities-def S-def*)
then have *Y-6-5-B*: $\bigwedge i. i \in \mathcal{B} \implies \text{hgt } (\text{pseq } (\text{Suc } i)) \geq \text{hgt } (\text{pseq } (i-1)) - 2$
 $* \varepsilon \text{ powr } (-1/2)$
using *bigY65B Y-6-5-Bblue unfolding B-def by blast*
have *big41*: *Big-Blue-4-1* μ *l*
and *hub*: *Big-height-upper-bound* *k*
and *16*: $k \geq 16$
and *ok-le-k*: $\text{ok-fun-26 } k - \text{ok-fun-28 } k \leq k$
using *big-x75 l-le-k by (auto simp: Big-X-7-5-def)*
have *oddset*: $\{..<\text{halted-point}\} \setminus \mathcal{D} = \{i \in \{..<\text{halted-point}\}. \text{ odd } i\}$
using *step-odd step-even not-halted-even-dreg halted-point-minimal by (auto simp: D-def)*
have [*simp*]: *finite* \mathcal{R} *finite* \mathcal{B} *finite* \mathcal{S}
using *finite-components by (auto simp: R-def B-def S-def)*
have [*simp*]: $\mathcal{R} \cap \mathcal{S} = \{\}$ **and** [*simp*]: $(\mathcal{R} \cup \mathcal{S}) \cap \mathcal{B} = \{\}$
by (*simp-all add: R-def S-def B-def Step-class-def disjoint-iff*)

have *hgt-qstar-le*: $\text{hgt } qstar \leq 2 * \varepsilon \text{ powr } (-1/4)$
proof (*intro real-hgt-Least*)
show $0 < 2 * \text{nat } \lfloor \varepsilon \text{ powr } (-1/4) \rfloor$
using *kn0 eps-gt0 by (simp add: eps-le1 powr-le1 powr-minus-divide)*
show $qstar \leq \text{qfun } (2 * \text{nat } \lfloor \varepsilon \text{ powr } (-1/4) \rfloor)$
using *kn0 711*
by (*simp add: qstar-def alpha-def qfun-eq divide-right-mono mult.commute*)
qed *auto*
then have $((1 + \varepsilon) * (1 + \varepsilon) \wedge \text{hgt } qstar) \leq ((1 + \varepsilon) * (1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/4)))$
by (*smt (verit) eps-ge0 mult-left-mono powr-mono powr-realpow*)
also have $((1 + \varepsilon) * (1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/4))) \leq 2$
using *le2 by simp*
finally have $(1 + \varepsilon) * (1 + \varepsilon) \wedge \text{hgt } qstar \leq 2$.
moreover have $\text{card } \mathcal{R} \leq k$
by (*simp add: R-def less-imp-le red-step-limit*)
ultimately have $\S: ((1 + \varepsilon) * (1 + \varepsilon) \wedge \text{hgt } qstar) * \text{card } \mathcal{R} \leq 2 * \text{real } k$
by (*intro mult-mono auto*)
have $-2 * \text{alpha } 1 * k \leq -\text{alpha } (\text{hgt } qstar + 2) * \text{card } \mathcal{R}$
using *mult-right-mono-neg [OF S, of -] eps-ge0*
by (*simp add: alpha-eq divide-simps mult-ac*)
also have $\dots \leq (\sum i \in \mathcal{R}. \text{pstar } (\text{Suc } i) - \text{pstar } i)$
proof –

```

{ fix i
  assume i ∈ R
  have - alpha (hgt qstar + 2) ≤ pstar (Suc i) - pstar i
  proof (cases hgt (pseq i) > hgt qstar + 2)
    case True
    then have hgt (pseq (Suc i)) > hgt qstar
      using Y-6-5-Red 16 ⟨i ∈ R⟩ by (force simp: R-def)
    then have pstar (Suc i) = pstar i
      using True hgt-mono' pstar-def by fastforce
    then show ?thesis
      by (simp add: alpha-ge0)
    next
    case False
    with ⟨i ∈ R⟩ show ?thesis
      unfolding pstar-def R-def
      by (smt (verit, del-insts) Y-6-4-Red alpha-ge0 alpha-mono hgt-gt0
linorder-not-less)
  qed
}
then show ?thesis
  by (smt (verit, ccfv-SIG) mult-of-nat-commute sum-constant sum-mono)
qed
finally have - 2 * alpha 1 * k ≤ (∑ i ∈ R. pstar (Suc i) - pstar i) .
moreover have 0 ≤ (∑ i ∈ S. pstar (Suc i) - pstar i)
  using R53 by (intro sum-nonneg) (force simp: pstar-def)
ultimately have RS-half: - 2 * alpha 1 * k ≤ (∑ i ∈ R ∪ S. pstar (Suc i) -
pstar i)
  by (simp add: sum.union-disjoint)

let ?e12 = ε powr (-1/2)
define h' where h' ≡ hgt qstar + nat ⌊2 * ?e12⌋
have - alpha 1 * k ≤ -2 * ?e12 * alpha 1 * k powr (3/4)
  using mult-right-mono-neg [OF big34, of - alpha 1] alpha-ge0 [of 1]
  by (simp add: mult-ac)
also have ... ≤ -?e12 * alpha (h') * card B
proof -
  have card B ≤ l powr (3/4)
    using big41 bblue-step-limit by (simp add: B-def)
  also have ... ≤ k powr (3/4)
    by (simp add: powr-mono2 l-le-k)
  finally have 1: card B ≤ k powr (3/4) .
  have alpha (h') ≤ alpha (nat ⌊2 * ε powr (-1/4)⌋ + nat ⌊2 * ?e12⌋)
  proof (rule alpha-mono)
    show h' ≤ nat ⌊2 * ε powr (-1/4)⌋ + nat ⌊2 * ?e12⌋
      using h'-def hgt-qstar-le le-nat-floor by auto
  qed (simp add: hgt-gt0 h'-def)
  also have ... ≤ 2 * alpha 1
  proof -
    have *: (1 + ε) ^ (nat ⌊2 * ε powr (-1/4)⌋ + nat ⌊2 * ?e12⌋ - 1) ≤ 2

```

```

    using le2 by simp
    have 1 ≤ 2 * ε powr (-1/4)
      by (smt (verit) hgt-qstar-le Suc-leI divide-minus-left hgt-gt0 numeral-nat(7)
real-of-nat-ge-one-iff)
    then show ?thesis
      using mult-right-mono [OF *, of ε] eps-ge0
      by (simp add: alpha-eq hgt-gt0 divide-right-mono mult.commute)
    qed
    finally have 2: 2 * alpha 1 ≥ alpha (h') .
    show ?thesis
      using mult-right-mono-neg [OF mult-mono [OF 1 2], of -?e12] alpha-ge0
by (simp add: mult-ac)
    qed
    also have ... ≤ (∑ i∈B. pstar (Suc i) - pstar (i-1))
    proof -
      { fix i
        assume i ∈ B
        have -?e12 * alpha (h') ≤ pstar (Suc i) - pstar (i-1)
        proof (cases hgt (pseq (i-1)) > hgt qstar + 2 * ?e12)
          case True
            then have hgt (pseq (Suc i)) > hgt qstar
              using Y-6-5-B <i ∈ B> by (force simp: R-def)
            then have pstar (i-1) = pstar (Suc i)
              unfolding pstar-def
              by (smt (verit) True hgt-mono' of-nat-less-iff powr-non-neg)
            then show ?thesis
              by (simp add: alpha-ge0)
          next
            case False
            then have hgt (pseq (i-1)) ≤ h'
              by (simp add: h'-def) linarith
            then have †: alpha (hgt (pseq (i-1))) ≤ alpha h'
              by (intro alpha-mono hgt-gt0)
            have pseq (Suc i) ≥ pseq (i-1) - ?e12 * alpha (hgt (pseq (i-1)))
              using Y-6-4-Bblue <i ∈ B> unfolding B-def by blast
            with mult-left-mono [OF †, of ?e12] show ?thesis
              unfolding pstar-def
              by (smt (verit) alpha-ge0 mult-minus-left powr-non-neg mult-le-0-iff)
          qed
        }
      then show ?thesis
        by (smt (verit, cfv-SIG) mult-of-nat-commute sum-constant sum-mono)
    qed
    finally have B: - alpha 1 * k ≤ (∑ i∈B. pstar (Suc i) - pstar (i-1)) .

    have ε powr (-1/4) * alpha 1 * card ((RUS) ∩ C) ≤ (∑ i∈RUS. if i ∈ C then
ε powr (-1/4) * alpha 1 else 0)
      by (simp add: flip: sum.inter-restrict)
    also have (∑ i∈RUS. if i ∈ C then ε powr (-1/4) * alpha 1 else 0) ≤

```

```

( $\sum i \in \mathcal{R} \cup \mathcal{S}. pstar\ i - pstar\ (i-1)$ )
proof (intro sum-mono)
  fix  $i$ 
  assume  $i: i \in \mathcal{R} \cup \mathcal{S}$ 
  then obtain  $i-1 \in \mathcal{D} \ i > 0$ 
    unfolding  $\mathcal{R}$ -def  $\mathcal{S}$ -def  $\mathcal{D}$ -def by (metis dreg-before-step1 dreg-before-gt0
Step-class-insert Un-iff)
  then have  $pseq\ (i-1) \leq pseq\ i$ 
    by (metis Suc-pred' Y-6-4-DegreeReg  $\mathcal{D}$ -def)
  then have  $pstar\ (i-1) \leq pstar\ i$ 
    by (fastforce simp: pstar-def)
  then show (if  $i \in C$  then  $\varepsilon\ powr\ (-1/4) * \alpha\ 1$  else 0)  $\leq pstar\ i - pstar\ (i-1)$ 
    using  $C$ -def pstar-def qstar-def by auto
  qed
  finally have  $\S: \varepsilon\ powr\ (-1/4) * \alpha\ 1 * card\ ((\mathcal{R} \cup \mathcal{S}) \cap C) \leq (\sum i \in \mathcal{R} \cup \mathcal{S}. pstar\ i - pstar\ (i-1))$  .

  have  $psplit: pstar\ (Suc\ i) - pstar\ (i-1) = (pstar\ (Suc\ i) - pstar\ i) + (pstar\ i - pstar\ (i-1))$  for  $i$ 
    by simp
  have  $RS: \varepsilon\ powr\ (-1/4) * \alpha\ 1 * card\ ((\mathcal{R} \cup \mathcal{S}) \cap C) + (-2 * \alpha\ 1 * k) \leq (\sum i \in \mathcal{R} \cup \mathcal{S}. pstar\ (Suc\ i) - pstar\ (i-1))$ 
    unfolding  $psplit\ sum.distrib$  using  $RS$ -half  $\S$  by linarith

  have  $k16: k\ powr\ (1/16) \leq k\ powr\ 1$ 
    using  $kn0$  by (intro powr-mono) auto

  have  $meq: \{..<halted-point\} \setminus \mathcal{D} = (\mathcal{R} \cup \mathcal{S}) \cup \mathcal{B}$ 
    using Step-class-cases halted-point-minimal' by (fastforce simp:  $\mathcal{R}$ -def  $\mathcal{S}$ -def  $\mathcal{D}$ -def  $\mathcal{B}$ -def Step-class-def)

  have  $(\varepsilon\ powr\ (-1/4) * \alpha\ 1 * card\ ((\mathcal{R} \cup \mathcal{S}) \cap C) + (-2 * \alpha\ 1 * k) + (-\alpha\ 1 * k) \leq (\sum i \in \mathcal{R} \cup \mathcal{S}. pstar\ (Suc\ i) - pstar\ (i-1)) + (\sum i \in \mathcal{B}. pstar\ (Suc\ i) - pstar\ (i-1))$ 
    using  $RS\ B$  by linarith
  also have  $\dots = (\sum i \in \{..<halted-point\} \setminus \mathcal{D}. pstar\ (Suc\ i) - pstar\ (i-1))$ 
    by (simp add: meq sum.union-disjoint)
  also have  $\dots \leq pstar\ halted-point - pstar\ 0$ 
  proof (cases even halted-point)
    case False
      have  $pseq\ (halted-point - Suc\ 0) \leq pseq\ halted-point$ 
        using Y-6-4-DegreeReg [of halted-point-1] False not-halted-even-dreg odd-pos

        by (auto simp: halted-point-minimal)
      then have  $pstar\ (halted-point - Suc\ 0) \leq pstar\ halted-point$ 
        by (simp add: pstar-def)
      with False show ?thesis

```

by (simp add: oddset sum-odds-odd)
 qed (simp add: oddset sum-odds-even)
 also have ... = $(\sum i < \text{halted-point}. \text{pstar}(\text{Suc } i) - \text{pstar } i)$
 by (simp add: sum-lessThan-telescope)
 also have ... = $\text{pstar } \text{halted-point} - \text{pstar } 0$
 by (simp add: sum-lessThan-telescope)
 also have ... $\leq \text{alpha } 1 * \varepsilon \text{ powr } (-1/4)$
 using alpha-ge0 by (simp add: mult.commute pee-eq-p0 pstar-def qstar-def)
 also have ... $\leq \text{alpha } 1 * k$
 using alpha-ge0 k16 by (intro powr-mono mult-left-mono) (auto simp: eps-def
 powr-powr)
 finally have $\varepsilon \text{ powr } (-1/4) * \text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) * \text{alpha } 1 \leq 4 * k * \text{alpha } 1$
 by (simp add: mult-ac)
 then have $\varepsilon \text{ powr } (-1/4) * \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C)) \leq 4 * k$
 using kn0 by (simp add: divide-simps alpha-eq eps-gt0)
 then show ?thesis
 using alpha-ge0[of 1] kn0 eps-gt0 by (simp add: powr-minus divide-simps
 mult-ac split: if-split-asm)
 qed

6.11 Lemma 7.12

definition *Big-X-7-12* \equiv

$\lambda \mu l. \text{Big-X-7-11 } \mu l \wedge \text{Big-X-7-10 } \mu l \wedge (\forall k. l \leq k \longrightarrow \text{Big-X-7-9 } k)$

establishing the size requirements for 7.12

lemma *Big-X-7-12*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-X-7-12 } \mu l$

using assms *Big-X-7-11 Big-X-7-10 Big-X-7-9*

unfolding *Big-X-7-12-def eventually-conj-iff*

apply (simp add: eventually-conj-iff all-imp-conj-distrib eventually-frequently-const-simps)

using eventually-all-ge-at-top by blast

lemma (in *Book*) *X-7-12*:

defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$

defines $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$

defines $C \equiv \{i. \text{card } (X\text{seq } i) < (1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card } (X\text{seq } (i-1))\}$

assumes big: *Big-X-7-12* μl

shows $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C) \leq 7 * \varepsilon \text{ powr } (1/4) * k$

proof –

define \mathcal{D} where $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$

have big-711: *Big-X-7-11* μl and big-710: *Big-X-7-10* μl

using big by (auto simp: *Big-X-7-12-def*)

have [simp]: finite \mathcal{R} finite \mathcal{S}

using finite-components by (auto simp: \mathcal{R} -def \mathcal{S} -def)

— now the conditions for Lemmas 7.10 and 7.11

define $C10$ where $C10 \equiv \{i. \text{hgt } (p\text{seq } i) \geq \text{hgt } (p\text{seq } (i-1)) + \varepsilon \text{ powr } (-1/4)\}$

define $C11$ where $C11 \equiv \{i. p\text{seq } i \geq p\text{seq } (i-1) + \varepsilon \text{ powr } (-1/4) * \text{alpha } 1 \wedge p\text{seq } (i-1) \leq p0\}$

```

have (( $\mathcal{R} \cup \mathcal{S}$ )  $\cap$   $C \cap \{i. \text{pseq } (i-1) \leq p0\}$ )  $\subseteq$  (( $\mathcal{R} \cup \mathcal{S}$ )  $\cap$   $C11$ )
proof
  fix i
  assume i:  $i \in ((\mathcal{R} \cup \mathcal{S}) \cap C \cap \{i. \text{pseq } (i-1) \leq p0\})$ 
  then have iRS:  $i \in \mathcal{R} \cup \mathcal{S}$  and iC:  $i \in C$ 
  by auto
  then obtain i1:  $i-1 \in \mathcal{D}$  i>0
  unfolding  $\mathcal{R}$ -def  $\mathcal{S}$ -def  $\mathcal{D}$ -def by (metis Step-class-insert Un-iff dreg-before-step1
dreg-before-gt0)
  then have 77:  $\text{card } (X\text{seq } (i-1) \setminus X\text{seq } i) / \text{card } (X\text{seq } i) * (\varepsilon \text{ powr } (-1/2))$ 
*  $\text{alpha } (\text{hgt } (\text{pseq } (i-1)))$ 
     $\leq \text{pseq } i - \text{pseq } (i-1)$ 
  by (metis Suc-diff-1 X-7-7  $\mathcal{D}$ -def)
  have card-Xm1:  $\text{card } (X\text{seq } (i-1)) = \text{card } (X\text{seq } i) + \text{card } (X\text{seq } (i-1) \setminus X\text{seq } i)$ 
  by (metis Xseq-antimono add-diff-inverse-nat card-Diff-subset card-mono
diff-le-self
finite-Xseq linorder-not-less)
  have card (Xseq i) > 0
  by (metis Step-class-insert card-Xseq-pos  $\mathcal{R}$ -def  $\mathcal{S}$ -def iRS)
  have card (Xseq (i-1)) > 0
  using C-def iC less-irrefl by fastforce
  moreover have  $2 * (\text{card } (X\text{seq } (i-1)) * \varepsilon \text{ powr } (1/4)) < \text{card } (X\text{seq } (i-1))$ 
\  $X\text{seq } i$ 
  using iC card-Xm1 by (simp add: algebra-simps C-def)
  moreover have  $\text{card } (X\text{seq } i) \leq 2 * \text{card } (X\text{seq } (i-1))$ 
  using card-Xm1 by linarith
  ultimately have  $\varepsilon \text{ powr } (1/4) \leq \text{card } (X\text{seq } (i-1) \setminus X\text{seq } i) / \text{card } (X\text{seq } (i-1))$ 
  by (simp add: divide-simps mult.commute)
  moreover have  $\text{real } (\text{card } (X\text{seq } i)) \leq \text{card } (X\text{seq } (i-1))$ 
  using card-Xm1 by linarith
  ultimately have 1:  $\varepsilon \text{ powr } (1/4) \leq \text{card } (X\text{seq } (i-1) \setminus X\text{seq } i) / \text{card } (X\text{seq } i)$ 
  by (smt (verit) <0 < card (Xseq i)> frac-le of-nat-0-le-iff of-nat-0-less-iff)
  have  $\varepsilon \text{ powr } (-1/4) * \text{alpha } 1$ 
     $\leq \text{card } (X\text{seq } (i-1) \setminus X\text{seq } i) / \text{card } (X\text{seq } i) * (\varepsilon \text{ powr } (-1/2)) * \text{alpha } 1$ 
  using alpha-ge0 mult-right-mono [OF 1, of  $\varepsilon \text{ powr } (-1/2) * \text{alpha } 1$ ]
  by (simp add: mult-ac flip: powr-add)
  also have  $\dots \leq \text{card } (X\text{seq } (i-1) \setminus X\text{seq } i) / \text{card } (X\text{seq } i) * (\varepsilon \text{ powr } (-1/2))$ 
*  $\text{alpha } (\text{hgt } (\text{pseq } (i-1)))$ 
  by (intro mult-left-mono alpha-mono) (auto simp: Suc-leI hgt-gt0)
  also have  $\dots \leq \text{pseq } i - \text{pseq } (i-1)$ 
  using 77 by simp
  finally have  $\varepsilon \text{ powr } (-1/4) * \text{alpha } 1 \leq \text{pseq } i - \text{pseq } (i-1)$  .
  with i show  $i \in (\mathcal{R} \cup \mathcal{S}) \cap C11$ 
  by (simp add: C11-def)
qed
then have  $\text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C \cap \{i. \text{pseq } (i-1) \leq p0\})) \leq \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S})$ 

```

$\cap C11))$
by (*simp add: card-mono*)
also have $\dots \leq 4 * \varepsilon \text{ powr } (1/4) * k$
using *X-7-11 big-711* **by** (*simp add: \mathcal{R} -def \mathcal{S} -def C11-def Step-class-insert-NO-MATCH*)
finally have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C \cap \{i. \text{pseq } (i-1) \leq p0\}) \leq 4 * \varepsilon \text{ powr } (1/4) * k$.
moreover
have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C \setminus \{i. \text{pseq } (i-1) \leq p0\}) \leq 3 * \varepsilon \text{ powr } (1/4) * k$
proof –
have *Big-X-7-9 k*
using *Big-X-7-12-def big l-le-k* **by** *presburger*
then have *X79: card (Xseq (Suc i)) $\geq (1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card } (Xseq$*
i)
if $i \in \text{Step-class } \{\text{dreg-step}\}$ **and** $\text{pseq } i \geq p0$
and $\text{hgt } (\text{pseq } (\text{Suc } i)) \leq \text{hgt } (\text{pseq } i) + \varepsilon \text{ powr } (-1/4)$ **for** i
using *X-7-9 that* **by** *blast*
have $(\mathcal{R} \cup \mathcal{S}) \cap C \setminus \{i. \text{pseq } (i-1) \leq p0\} \subseteq (\mathcal{R} \cup \mathcal{S}) \cap C10$
unfolding *C10-def C-def*
proof *clarify*
fix i
assume $i \in \mathcal{R} \cup \mathcal{S}$
and $\S: \text{card } (Xseq\ i) < (1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card } (Xseq\ (i-1)) \neg \text{pseq}$
 $(i-1) \leq p0$
then obtain $i-1 \in \mathcal{D}$ $i > 0$
unfolding *D-def \mathcal{R} -def \mathcal{S} -def*
by (*metis dreg-before-step1 dreg-before-gt0 Step-class-Un Un-iff insert-is-Un*)
with *X79 \S* **show** $\text{hgt } (\text{pseq } (i - 1)) + \varepsilon \text{ powr } (-1/4) \leq \text{hgt } (\text{pseq } i)$
by (*force simp: D-def*)
qed
then have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C \setminus \{i. \text{pseq } (i-1) \leq p0\}) \leq \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap$
 $C10))$
by (*simp add: card-mono*)
also have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C10) \leq 3 * \varepsilon \text{ powr } (1/4) * k$
unfolding *\mathcal{R} -def \mathcal{S} -def C10-def* **by** (*intro X-7-10 assms big-710*)
finally show *?thesis*.
qed
moreover
have $\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C)$
 $= \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C \cap \{i. \text{pseq } (i-1) \leq p0\})) + \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap$
 $C \setminus \{i. \text{pseq } (i-1) \leq p0\}))$
by (*metis card-Int-Diff of-nat-add <finite \mathcal{R} > <finite \mathcal{S} > finite-Int infinite-Un*)
ultimately show *?thesis*
by *linarith*
qed

6.12 Lemma 7.6

definition *Big-X-7-6* \equiv

$$\lambda \mu l. \text{Big-Blue-4-1 } \mu\ l \wedge \text{Big-X-7-12 } \mu\ l \wedge (\forall k. k \geq l \longrightarrow \text{Big-X-7-8 } k \wedge 1 - 2$$

* $\text{eps } k \text{ powr } (1/4) > 0$)

lemma *Big-X-7-6*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-X-7-6 } \mu \ l$

using *assms Big-Blue-4-1 Big-X-7-8 Big-X-7-12*

unfolding *Big-X-7-6-def eps-def*

apply (*simp add: eventually-conj-iff all-imp-conj-distrib eventually-all-ge-at-top*)

apply (*intro conjI strip eventually-all-geI0 eventually-all-ge-at-top; real-asymp*)

done

definition *ok-fun-76* \equiv

$\lambda k. ((1 + 2 * \text{real } k) * \ln (1 - 2 * \text{eps } k \text{ powr } (1/4)))$

$- (k \text{ powr } (3/4) + 7 * \text{eps } k \text{ powr } (1/4) * k + 1) * (2 * \ln k) / \ln 2$

lemma *ok-fun-76*: $\text{ok-fun-76} \in o(\text{real})$

unfolding *eps-def ok-fun-76-def* **by** *real-asymp*

lemma (in *Book*) *X-7-6*:

assumes *big*: *Big-X-7-6* $\mu \ l$

defines $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$

shows $(\prod_{i \in \mathcal{D}} \text{card}(X\text{seq}(\text{Suc } i)) / \text{card}(X\text{seq } i)) \geq 2 \text{ powr } \text{ok-fun-76 } k$

proof –

define \mathcal{R} **where** $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$

define \mathcal{B} **where** $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$

define \mathcal{S} **where** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$

define C **where** $C \equiv \{i. \text{card}(X\text{seq } i) < (1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card}(X\text{seq } (i-1))\}$

define C' **where** $C' \equiv \text{Suc } -' C$

have *big41*: *Big-Blue-4-1* $\mu \ l$

and *712*: $\text{card}((\mathcal{R} \cup \mathcal{S}) \cap C) \leq 7 * \varepsilon \text{ powr } (1/4) * k$

using *big X-7-12 l-le-k* **by** (*auto simp: Big-X-7-6-def R-def S-def C-def*)

have [*simp*]: *finite D finite R finite B finite S*

using *finite-components* **by** (*auto simp: D-def R-def B-def S-def*)

have $\text{card } \mathcal{R} < k$

using *R-def assms red-step-limit* **by** *blast+*

have $\text{card } \mathcal{B} \leq l \text{ powr } (3/4)$

using *big41 bblue-step-limit* **by** (*auto simp: B-def*)

then have $\text{card}(\mathcal{B} \cap C) \leq l \text{ powr } (3/4)$

using *card-mono [OF - Int-lower1]* **by** (*smt (verit) <finite B> of-nat-mono*)

also have $\dots \leq k \text{ powr } (3/4)$

by (*simp add: l-le-k powr-mono2*)

finally have *Bk-34*: $\text{card}(\mathcal{B} \cap C) \leq k \text{ powr } (3/4)$.

have *less-l*: $\text{card } \mathcal{B} + \text{card } \mathcal{S} < l$

using *bblue-dboost-step-limit big41* **by** (*auto simp: B-def S-def*)

have [*simp*]: $(\mathcal{B} \cup (\mathcal{R} \cup \mathcal{S})) \cap \{\text{halted-point}\} = \{\} \ \mathcal{R} \cap \mathcal{S} = \{\} \ \mathcal{B} \cap (\mathcal{R} \cup \mathcal{S}) =$

```

{}
    halted-point  $\notin$   $\mathcal{B}$  halted-point  $\notin$   $\mathcal{R}$  halted-point  $\notin$   $\mathcal{S}$ 
     $\mathcal{B} \cap C \cap (\mathcal{R} \cap C \cup \mathcal{S} \cap C) = \{\}$  for  $C$ 
using halted-point-minimal' by (force simp:  $\mathcal{B}$ -def  $\mathcal{R}$ -def  $\mathcal{S}$ -def Step-class-def)+

have Big-X-7-8  $k$  and one-minus-gt0:  $1 - 2 * \epsilon \text{ powr } (1/4) > 0$ 
    using big l-le-k by (auto simp: Big-X-7-6-def)
then have X78:  $\text{card } (X\text{seq } (\text{Suc } i)) \geq \text{card } (X\text{seq } i) / k^2$  if  $i \in \mathcal{D}$  for  $i$ 
    using X-7-8 that by (force simp:  $\mathcal{D}$ -def)

let ?DC =  $\lambda k. k \text{ powr } (3/4) + 7 * \epsilon k \text{ powr } (1/4) * k + 1$ 
have dc-pos:  $?DC k > 0$  for  $k$ 
    by (smt (verit) of-nat-less-0-iff powr-ge-zero zero-le-mult-iff)
have X-pos:  $\text{card } (X\text{seq } i) > 0$  if  $i \in \mathcal{D}$  for  $i$ 
proof -
    have  $\text{card } (X\text{seq } (\text{Suc } i)) > 0$ 
        using that X-7-7 kn0 unfolding  $\mathcal{D}$ -def by blast
    then show ?thesis
        by (metis Xseq-Suc-subset card-mono finite-Xseq gr0I leD)
qed
have ok-fun-76  $k \leq \log 2 ((1 / (\text{real } k)^2) \text{ powr } ?DC k * (1 - 2 * \epsilon \text{ powr } (1/4)))$ 
 $^{\wedge} (k + l + 1)$ 
    unfolding ok-fun-76-def log-def
    using kn0 l-le-k one-minus-gt0
    by (simp add: ln-mult ln-div ln-realpow divide-right-mono mult-le-cancel-right
flip: power-Suc mult.assoc)
    then have  $2 \text{ powr } ok\text{-fun-76 } k \leq (1 / (\text{real } k)^2) \text{ powr } ?DC k * (1 - 2 * \epsilon \text{ powr } (1/4))$ 
 $^{\wedge} (k + l + 1)$ 
        using powr-eq-iff kn0 one-minus-gt0 by (simp add: le-log-iff)
    also have  $\dots \leq (1 / (\text{real } k)^2) \text{ powr } \text{card } (\mathcal{D} \cap C') * (1 - 2 * \epsilon \text{ powr } (1/4))$ 
 $^{\wedge} \text{card } (\mathcal{D} \setminus C')$ 
proof (intro mult-mono powr-mono')
    have  $\text{Suc } i \in \mathcal{R}$  if  $i \in \mathcal{D}$   $\text{Suc } i \neq \text{halted-point}$   $\text{Suc } i \notin \mathcal{B}$   $\text{Suc } i \notin \mathcal{S}$  for  $i$ 
proof -
    have  $\text{Suc } i \notin \mathcal{D}$ 
        by (metis  $\mathcal{D}$ -def  $\langle i \in \mathcal{D} \rangle$  even-Suc step-even)
    moreover
    have stepper-kind  $i \neq \text{halted}$ 
        using  $\mathcal{D}$ -def  $\langle i \in \mathcal{D} \rangle$  Step-class-def by force
    ultimately show  $\text{Suc } i \in \mathcal{R}$ 
        using that halted-point-minimal' halted-point-minimal Step-class-cases
Suc-lessI
 $\mathcal{B}$ -def  $\mathcal{D}$ -def  $\mathcal{R}$ -def  $\mathcal{S}$ -def by blast
qed
then have  $\text{Suc } i \in \mathcal{D} \subseteq \mathcal{B} \cup (\mathcal{R} \cup \mathcal{S}) \cup \{\text{halted-point}\}$ 
    by auto
then have ifD:  $\text{Suc } i \in \mathcal{B} \vee \text{Suc } i \in \mathcal{R} \vee \text{Suc } i \in \mathcal{S} \vee \text{Suc } i = \text{halted-point}$  if
 $i \in \mathcal{D}$  for  $i$ 
    using that by force

```

then have $\text{card } \mathcal{D} \leq \text{card } (\mathcal{B} \cup (\mathcal{R} \cup \mathcal{S}) \cup \{\text{halted-point}\})$
by (*intro card-inj-on-le [of Suc]*) *auto*
also have $\dots = \text{card } \mathcal{B} + \text{card } \mathcal{R} + \text{card } \mathcal{S} + 1$
by (*simp add: card-Un-disjoint card-insert-if*)
also have $\dots \leq k + l + 1$
using $\langle \text{card } \mathcal{R} < k \rangle$ *less-l* **by** *linarith*
finally have *card-D*: $\text{card } \mathcal{D} \leq k + l + 1$.

have $(1 - 2 * \varepsilon \text{ powr } (1/4)) * \text{card } (Xseq\ 0) \leq 1 * \text{real } (\text{card } (Xseq\ 0))$
by (*intro mult-right-mono; force*)
then have $0 \notin C$
by (*force simp: C-def*)
then have *C-eq-C'*: $C = \text{Suc } ' C'$
using *nat.exhaust* **by** (*auto simp: C'-def set-eq-iff image-iff*)
have $\text{card } (\mathcal{D} \cap C') \leq \text{real } (\text{card } ((\mathcal{B} \cup (\mathcal{R} \cup \mathcal{S}) \cup \{\text{halted-point}\}) \cap C))$
using *ifD*
by (*intro of-nat-mono card-inj-on-le [of Suc]*) (*force simp: Int-insert-left C-eq-C'*)
also have $\dots \leq \text{card } (\mathcal{B} \cap C) + \text{real } (\text{card } ((\mathcal{R} \cup \mathcal{S}) \cap C)) + 1$
by (*simp add: Int-insert-left Int-Un-distrib2 card-Un-disjoint card-insert-if*)
also have $\dots \leq ?DC\ k$
using *Bk-34 712* **by** *force*
finally show $\text{card } (\mathcal{D} \cap C') \leq ?DC\ k$.
have $\text{card } (\mathcal{D} \setminus C') \leq \text{card } \mathcal{D}$
using $\langle \text{finite } \mathcal{D} \rangle$ **by** (*simp add: card-mono*)
then show $(1 - 2 * \varepsilon \text{ powr } (1/4)) ^ (k+l+1) \leq (1 - 2 * \varepsilon \text{ powr } (1/4)) ^ \text{card } (\mathcal{D} \setminus C')$
by (*smt (verit) card-D add-leD2 one-minus-gt0 power-decreasing powr-ge-zero*)
qed (*use one-minus-gt0 kn0 in auto*)
also have $\dots = (\prod i \in \mathcal{D}. \text{if } i \in C' \text{ then } 1 / \text{real } k ^ 2 \text{ else } 1 - 2 * \varepsilon \text{ powr } (1/4))$
by (*simp add: kn0 powr-realpow prod.If-cases Diff-eq*)
also have $\dots \leq (\prod i \in \mathcal{D}. \text{card } (Xseq\ (\text{Suc } i)) / \text{card } (Xseq\ i))$
using *X-pos X78 one-minus-gt0 kn0* **by** (*simp add: divide-simps C'-def C-def prod-mono*)
finally show *?thesis* .
qed

6.13 Lemma 7.1

definition *Big-X-7-1* \equiv

$$\lambda \mu l. \text{Big-Blue-4-1 } \mu\ l \wedge \text{Big-X-7-2 } \mu\ l \wedge \text{Big-X-7-4 } \mu\ l \wedge \text{Big-X-7-6 } \mu\ l$$

establishing the size requirements for 7.11

lemma *Big-X-7-1*:

assumes $0 < \mu 0$ $\mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0 .. \mu 1\} \longrightarrow \text{Big-X-7-1 } \mu\ l$

unfolding *Big-X-7-1-def*

using *assms Big-Blue-4-1 Big-X-7-2 Big-X-7-4 Big-X-7-6*

by (*simp add: eventually-conj-iff all-imp-conj-distrib*)

definition *ok-fun-71* $\equiv \lambda\mu k. \text{ok-fun-72 } \mu k + \text{ok-fun-73 } k + \text{ok-fun-74 } k + \text{ok-fun-76 } k$

lemma *ok-fun-71*:

assumes $0 < \mu < 1$
shows *ok-fun-71* $\mu \in o(\text{real})$
using *ok-fun-72 ok-fun-73 ok-fun-74 ok-fun-76*
by (*simp add: assms ok-fun-71-def sum-in-smallo*)

lemma (in *Book*) *X-7-1*:

assumes *big*: *Big-X-7-1* μl
defines $\mathcal{D} \equiv \text{Step-class } \{\text{dreg-step}\}$
defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ **and** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
shows $\text{card } (X\text{seq halted-point}) \geq$
 $2 \text{ powr } \text{ok-fun-71 } \mu k * \mu^l * (1-\mu)^\wedge \text{card } \mathcal{R} * (\text{bigbeta} / \mu)^\wedge \text{card } \mathcal{S} * \text{card } X0$

proof –

define \mathcal{B} **where** $\mathcal{B} \equiv \text{Step-class } \{\text{bblue-step}\}$
have *72*: *Big-X-7-2* μl **and** *74*: *Big-X-7-4* μl
and *76*: *Big-X-7-6* μl
and *big41*: *Big-Blue-4-1* μl
using *big* **by** (*auto simp: Big-X-7-1-def*)
then have [*simp*]: $\text{finite } \mathcal{R} \text{ finite } \mathcal{B} \text{ finite } \mathcal{S} \text{ finite } \mathcal{D}$
 $\mathcal{R} \cap \mathcal{B} = \{\} \mathcal{S} \cap \mathcal{D} = \{\} (\mathcal{R} \cup \mathcal{B}) \cap (\mathcal{S} \cup \mathcal{D}) = \{\}$
using *finite-components* **by** (*auto simp: R-def B-def S-def D-def Step-class-def*)
have *BS-le-l*: $\text{card } \mathcal{B} + \text{card } \mathcal{S} < l$
using *big41 bblue-dboost-step-limit* **by** (*auto simp: S-def B-def*)

have *R*: $(\prod i \in \mathcal{R}. \text{card } (X\text{seq}(Suc i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } (\text{ok-fun-72 } \mu k)$
 $* (1-\mu)^\wedge \text{card } \mathcal{R}$
unfolding *R-def* **using** *72 X-7-2* **by** *meson*
have *B*: $(\prod i \in \mathcal{B}. \text{card } (X\text{seq}(Suc i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } (\text{ok-fun-73 } k) *$
 $\mu^\wedge (l - \text{card } \mathcal{S})$
unfolding *B-def S-def* **using** *big41 X-7-3* **by** *meson*
have *S*: $(\prod i \in \mathcal{S}. \text{card } (X\text{seq}(Suc i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } \text{ok-fun-74 } k *$
 $\text{bigbeta}^\wedge \text{card } \mathcal{S}$
unfolding *S-def* **using** *74 X-7-4* **by** *meson*
have *D*: $(\prod i \in \mathcal{D}. \text{card } (X\text{seq}(Suc i)) / \text{card } (X\text{seq } i)) \geq 2 \text{ powr } \text{ok-fun-76 } k$
unfolding *D-def* **using** *76 X-7-6* **by** *meson*
have *below-m*: $\mathcal{R} \cup \mathcal{B} \cup \mathcal{S} \cup \mathcal{D} = \{.. < \text{halted-point}\}$
using *assms* **by** (*auto simp: R-def B-def S-def D-def before-halted-eq Step-class-insert-NO-MATCH*)
have *X-nz*: $\bigwedge i. i < \text{halted-point} \implies \text{card } (X\text{seq } i) \neq 0$
using *assms below-halted-point-cardX* **by** *blast*
have *tele*: $\text{card } (X\text{seq halted-point}) = (\prod i < \text{halted-point}. \text{card } (X\text{seq}(Suc i)) /$
 $\text{card } (X\text{seq } i)) * \text{card } (X\text{seq } 0)$
proof (*cases halted-point=0*)
case *False*
with *X-nz prod-lessThan-telescope-mult* [**where** $f = \lambda i. \text{real } (\text{card } (X\text{seq } i))$]

```

    show ?thesis by simp
  qed auto
  have X0-nz: card (Xseq 0) > 0
    by (simp add: card-XY0)
  have 2 powr ok-fun-71  $\mu k * \mu^l * (1-\mu)^{\text{card } \mathcal{R}} * (\text{bigbeta} / \mu)^{\text{card } \mathcal{S}}$ 
     $\leq 2 \text{ powr ok-fun-71 } \mu k * \mu^{\text{card } \mathcal{S}} * (1-\mu)^{\text{card } \mathcal{R}} * (\text{bigbeta}^{\text{card } \mathcal{S}})$ 
    using  $\mu 01$  BS-le-l by (simp add: power-diff power-divide)
  also have ...  $\leq (\prod_{i \in \mathcal{R} \cup \mathcal{B} \cup \mathcal{S} \cup \mathcal{D}} \text{card } (Xseq(\text{Suc } i)) / \text{card } (Xseq i))$ 
  proof -
    have  $(\prod_{i \in (\mathcal{R} \cup \mathcal{B}) \cup (\mathcal{S} \cup \mathcal{D})} \text{card } (Xseq(\text{Suc } i)) / \text{card } (Xseq i))$ 
       $\geq ((2 \text{ powr } (ok-fun-72) \mu k) * (1-\mu)^{\text{card } \mathcal{R}} * (2 \text{ powr } (ok-fun-73) k) * \mu^{\text{card } \mathcal{S}})$ 
       $* ((2 \text{ powr } (ok-fun-74) k * \text{bigbeta}^{\text{card } \mathcal{S}}) * (2 \text{ powr } (ok-fun-76) k))$ 
    using  $\mu 01$  by (auto simp: R B S D prod.union-disjoint prod-nonneg bigbeta-ge0 intro!: mult-mono)
    then show ?thesis
      by (simp add: Un-assoc mult-ac powr-add ok-fun-71-def)
  qed
  also have ...  $\leq (\prod_{i < \text{halted-point}} \text{card } (Xseq(\text{Suc } i)) / \text{card } (Xseq i))$ 
    using below-m by auto
  finally show ?thesis
    using X0-nz  $\mu 01$  unfolding tele by (simp add: divide-simps)
  qed
end

```

7 The Zigzag Lemma

theory Zigzag imports Bounding-X

begin

7.1 Lemma 8.1 (the actual Zigzag Lemma)

definition *Big-ZZ-8-2* $\equiv \lambda k. (1 + \text{eps } k \text{ powr } (1/2)) \geq (1 + \text{eps } k) \text{ powr } (\text{eps } k \text{ powr } (-1/4))$

An inequality that pops up in the proof of (39)

definition *Big39* $\equiv \lambda k. 1/2 \leq (1 + \text{eps } k) \text{ powr } (-2 * \text{eps } k \text{ powr } (-1/2))$

Two inequalities that pops up in the proof of (42)

definition *Big42a* $\equiv \lambda k. (1 + \text{eps } k)^2 / (1 - \text{eps } k \text{ powr } (1/2)) \leq 1 + 2 * k \text{ powr } (-1/16)$

definition *Big42b* $\equiv \lambda k. 2 * k \text{ powr } (-1/16) * k$
 $+ (1 + 2 * \ln k / \text{eps } k + 2 * k \text{ powr } (7/8)) / (1 - \text{eps } k \text{ powr } (1/2))$
 $\leq \text{real } k \text{ powr } (19/20)$

definition *Big-ZZ-8-1* \equiv
 $\lambda \mu l. \text{Big-Blue-4-1 } \mu l \wedge \text{Big-Red-5-1 } \mu l \wedge \text{Big-Red-5-3 } \mu l \wedge \text{Big-Y-6-5-Bblue}$
 l
 $\wedge (\forall k. k \geq l \longrightarrow \text{Big-height-upper-bound } k \wedge \text{Big-ZZ-8-2 } k \wedge k \geq 16 \wedge \text{Big39}$
 k
 $\wedge \text{Big42a } k \wedge \text{Big42b } k)$
 $(16::'a) \leq k$ is for *Y-6-5-Red*

lemma *Big-ZZ-8-1*:
assumes $0 < \mu 0 \ \mu 1 < 1$
shows $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-ZZ-8-1 } \mu l$
using *assms Big-Blue-4-1 Big-Red-5-1 Big-Red-5-3 Big-Y-6-5-Bblue*
unfolding *Big-ZZ-8-1-def Big-ZZ-8-2-def Big39-def Big42a-def Big42b-def*
eventually-conj-iff all-imp-conj-distrib eps-def
apply (*simp add: eventually-conj-iff eventually-frequently-const-simps*)
apply (*intro conjI strip eventually-all-ge-at-top Big-height-upper-bound; real-asymp*)
done

lemma (*in Book*) *ZZ-8-1*:
assumes *big: Big-ZZ-8-1* μl
defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$
defines $\text{sum-SS} \equiv (\sum_{i \in \text{dboost-star}}. (1 - \text{beta } i) / \text{beta } i)$
shows $\text{sum-SS} \leq \text{card } \mathcal{R} + k \text{ powr } (19/20)$
proof –
define *pp* **where** $pp \equiv \lambda i h. \text{if } h=1 \text{ then } \min (\text{pseq } i) (\text{qfun } 1)$
 $\text{else if } \text{pseq } i \leq \text{qfun } (h-1) \text{ then } \text{qfun } (h-1)$
 $\text{else if } \text{pseq } i \geq \text{qfun } h \text{ then } \text{qfun } h$
 $\text{else } \text{pseq } i$
define Δ **where** $\Delta \equiv \lambda i. \text{pseq } (\text{Suc } i) - \text{pseq } i$
define $\Delta\Delta$ **where** $\Delta\Delta \equiv \lambda i h. pp (\text{Suc } i) h - pp i h$
have *pp-eq*: $pp i h = (\text{if } h=1 \text{ then } \min (\text{pseq } i) (\text{qfun } 1)$
 $\text{else } \max (\text{qfun } (h-1)) (\min (\text{pseq } i) (\text{qfun } h)))$ **for** $i h$
using *qfun-mono [of h-1 h] by (auto simp: pp-def max-def)*

define *maxh* **where** $\text{maxh} \equiv \text{nat} \lfloor 2 * \ln k / \varepsilon \rfloor + 1$
have *maxh*: $\bigwedge \text{pseq}. \text{pseq} \leq 1 \implies \text{hgt } \text{pseq} \leq 2 * \ln k / \varepsilon$ **and** $k \geq 16$
using *big l-le-k by (auto simp: Big-ZZ-8-1-def height-upper-bound)*
then have $1 \leq 2 * \ln k / \varepsilon$
using *hgt-gt0 [of 1] by force*
then have $\text{maxh} > 1$
by (*simp add: maxh-def eps-gt0*)
have $\text{hgt } \text{pseq} < \text{maxh}$ **if** $\text{pseq} \leq 1$ **for** pseq
using *that kn0 maxh [of pseq] unfolding maxh-def by linarith*
then have *hgt-le-maxh*: $\text{hgt } (\text{pseq } i) < \text{maxh}$ **for** i
using *pee-le1 by auto*

have *pp-eq-hgt [simp]*: $pp i (\text{hgt } (\text{pseq } i)) = \text{pseq } i$ **for** i
using *hgt-less-imp-qfun-less [of hgt (pseq i) - 1 pseq i]*

```

using hgt-works [of pseq i] hgt-gt0 [of pseq i] kn0 pp-eq by force

have pp-less-hgt [simp]: pp i h = qfun h if 0 < h < hgt (pseq i) for h i
proof (cases h=1)
  case True
  then show ?thesis
  using hgt-less-imp-qfun-less pp-def that by auto
next
  case False
  with that show ?thesis
  using alpha-def alpha-ge0 hgt-less-imp-qfun-less pp-eq by force
qed

have pp-gt-hgt [simp]: pp i h = qfun (h-1) if h > hgt (pseq i) for h i
  using hgt-gt0 [of pseq i] kn0 that
  by (simp add: pp-def hgt-le-imp-qfun-ge)

have Δ0: Δ i ≥ 0 ↔ (∀ h > 0. Δ Δ i h ≥ 0) for i
proof (intro iffI strip)
  fix h::nat
  assume 0 ≤ Δ i 0 < h then show 0 ≤ Δ Δ i h
  using qfun-mono [of h-1 h] kn0 by (auto simp: Δ-def ΔΔ-def pp-def)
next
  assume ∀ h > 0. 0 ≤ Δ Δ i h
  then have pseq i ≤ pp (Suc i) (hgt (pseq i))
  unfolding ΔΔ-def
  by (smt (verit, best) hgt-gt0 pp-eq-hgt)
  then show 0 ≤ Δ i
  using hgt-less-imp-qfun-less [of hgt (pseq i) - 1 pseq i]
  using hgt-gt0 [of pseq i] kn0
  by (simp add: Δ-def pp-def split: if-split-asm)
qed

have sum-pp-aux: (∑ h=Suc 0..n. pp i h)
  = (if hgt (pseq i) ≤ n then pseq i + (∑ h=1..<n. qfun h) else
(∑ h=1..n. qfun h))
  if n > 0 for n i
  using that
proof (induction n)
  case (Suc n)
  show ?case
proof (cases n=0)
  case True
  then show ?thesis
  using kn0 hgt-Least [of 1 pseq i]
  by (simp add: pp-def hgt-le-imp-qfun-ge min-def)
next
  case False
  with Suc show ?thesis

```

```

    by (simp split: if-split-asm) (smt (verit) le-Suc-eq not-less-eq pp-eq-hgt
sum.head-if)
  qed
  qed auto
  have sum-pp: (∑ h=Suc 0..maxh. pp i h) = pseq i + (∑ h=1..<maxh. qfun h)
for i
  using <1 < maxh> by (simp add: hgt-le-maxh less-or-eq-imp-le sum-pp-aux)
  have 33: Δ i = (∑ h=1..maxh. ΔΔ i h) for i
  by (simp add: ΔΔ-def Δ-def sum-subtractf sum-pp)

  have (∑ i<halted-point. ΔΔ i h) = 0
  if ∧i. i<halted-point ⇒ h > hgt (pseq i) for h
  using that by (simp add: sum.neutral ΔΔ-def)
  then have B: (∑ i<halted-point. ΔΔ i h) = 0 if h ≥ maxh for h
  by (meson hgt-le-maxh le-simps le-trans not-less-eq that)
  have (∑ h=Suc 0..maxh. ∑ i<halted-point. ΔΔ i h / alpha h) ≤ (∑ h=Suc
0..maxh. 1)
  proof (intro sum-mono)
    fix h
    assume h ∈ {Suc 0..maxh}
    have (∑ i<halted-point. ΔΔ i h) ≤ alpha h
    using qfun-mono [of h-1 h] kn0
    unfolding ΔΔ-def alpha-def sum-lessThan-telescope [where f = λi. pp i h]
    by (auto simp: pp-def pee-eq-p0)
    then show (∑ i<halted-point. ΔΔ i h / alpha h) ≤ 1
    using alpha-ge0 [of h] by (simp add: divide-simps flip: sum-divide-distrib)
  qed
  also have ... ≤ 1 + 2 * ln k / ε
  using <maxh > 1> by (simp add: maxh-def)
  finally have 34: (∑ h=Suc 0..maxh. ∑ i<halted-point. ΔΔ i h / alpha h) ≤ 1
+ 2 * ln k / ε .

  define D where D ≡ Step-class {dreg-step}
  define B where B ≡ Step-class {bblue-step}
  define S where S ≡ Step-class {dboost-step}
  have dboost-star ⊆ S
  unfolding dboost-star-def S-def dboost-star-def by auto
  have BD-disj: B∩D = {} and disj: R∩B = {} S∩B = {} R∩D = {} S∩D =
{} R∩S = {}
  by (auto simp: D-def R-def B-def S-def Step-class-def)

  have [simp]: finite D finite B finite R finite S
  using finite-components assms
  by (auto simp: D-def B-def R-def S-def Step-class-insert-NO-MATCH)
  have card R < k
  using red-step-limit by (auto simp: R-def)

  have R52: pseq (Suc i) - pseq i ≥ (1 - ε) * ((1 - beta i) / beta i) * alpha (hgt
(pseq i))

```

and beta-gt0 : $\text{beta } i > 0$
and $R53$: $\text{pseq } (\text{Suc } i) \geq \text{pseq } i \wedge \text{beta } i \geq 1 / (\text{real } k)^2$
if $i \in \mathcal{S}$ **for** i
using big Red-5-2 **that** **by** (*auto simp: Big-ZZ-8-1-def Red-5-3 B-def S-def*)
have cardB : $\text{card } \mathcal{B} \leq l \text{ powr } (3/4)$ **and** bigY65B : *Big-Y-6-5-Bblue* l
using $\text{big bblue-step-limit}$ **by** (*auto simp: Big-ZZ-8-1-def B-def*)

have $\Delta\Delta\text{-ge0}$: $\Delta\Delta \ i \ h \geq 0$ **if** $i \in \mathcal{S}$ $h \geq 1$ **for** $i \ h$
using *that R53 [OF <i ∈ S>]* **by** (*fastforce simp: ΔΔ-def pp-eq*)
have $\Delta\Delta\text{-eq-0}$: $\Delta\Delta \ i \ h = 0$ **if** $\text{hgt } (\text{pseq } i) \leq \text{hgt } (\text{pseq } (\text{Suc } i))$ $\text{hgt } (\text{pseq } (\text{Suc } i)) < h$ **for** $h \ i$
using $\Delta\Delta\text{-def}$ **that** **by** *fastforce*
define oneminus **where** $\text{oneminus} \equiv 1 - \varepsilon \text{ powr } (1/2)$
have 35 : $\text{oneminus} * ((1 - \text{beta } i) / \text{beta } i)$
 $\leq (\sum_{h=1..maxh.} \Delta\Delta \ i \ h / \text{alpha } h)$ (**is** $?L \leq ?R$)
if $i \in \text{dboost-star}$ **for** i
proof –
have $i \in \mathcal{S}$
using $\langle \text{dboost-star} \subseteq \mathcal{S} \rangle$ **that** **by** *blast*
have $[\text{simp}]$: $\text{real } (\text{hgt } x - \text{Suc } 0) = \text{real } (\text{hgt } x) - 1$ **for** x
using hgt-gt0 $[\text{of } x]$ **by** *linarith*
have 36 : $(1 - \varepsilon) * ((1 - \text{beta } i) / \text{beta } i) \leq \Delta \ i / \text{alpha } (\text{hgt } (\text{pseq } i))$
using $R52 \ \text{alpha-gt0}$ $[\text{OF } \text{hgt-gt0}]$ beta-gt0 **that** $\langle \text{dboost-star} \subseteq \mathcal{S} \rangle$ **by** (*force simp: Δ-def divide-simps*)
have $k\text{-big}$: $(1 + \varepsilon \text{ powr } (1/2)) \geq (1 + \varepsilon) \text{ powr } (\varepsilon \text{ powr } (-1/4))$
using big l-le-k **by** (*auto simp: Big-ZZ-8-1-def Big-ZZ-8-2-def*)
have $*$: $\bigwedge x::\text{real. } x > 0 \implies (1 - x \text{ powr } (1/2)) * (1 + x \text{ powr } (1/2)) = 1 - x$
by (*simp add: algebra-simps flip: powr-add*)
have $?L = (1 - \varepsilon) * ((1 - \text{beta } i) / \text{beta } i) / (1 + \varepsilon \text{ powr } (1/2))$
using beta-gt0 $[\text{OF } \langle i \in \mathcal{S} \rangle]$ eps-gt0 $k\text{-big}$
by (*force simp: oneminus-def divide-simps **)
also have $\dots \leq \Delta \ i / \text{alpha } (\text{hgt } (\text{pseq } i)) / (1 + \varepsilon \text{ powr } (1/2))$
by (*intro 36 divide-right-mono auto*)
also have $\dots \leq \Delta \ i / \text{alpha } (\text{hgt } (\text{pseq } i)) / (1 + \varepsilon) \text{ powr } (\text{real } (\text{hgt } (\text{pseq } (\text{Suc } i))) - \text{hgt } (\text{pseq } i))$
proof (*intro divide-left-mono mult-pos-pos*)
have $\text{real } (\text{hgt } (\text{pseq } (\text{Suc } i))) - \text{hgt } (\text{pseq } i) \leq \varepsilon \text{ powr } (-1/4)$
using *that by (simp add: dboost-star-def)*
then show $(1 + \varepsilon) \text{ powr } (\text{real } (\text{hgt } (\text{pseq } (\text{Suc } i))) - \text{real } (\text{hgt } (\text{pseq } i))) \leq 1 + \varepsilon \text{ powr } (1/2)$
using $k\text{-big}$ **by** (*smt (verit) eps-ge0 powr-mono*)
show $0 \leq \Delta \ i / \text{alpha } (\text{hgt } (\text{pseq } i))$
by (*simp add: Δ0 ΔΔ-ge0 <i ∈ S> alpha-ge0*)
show $0 < (1 + \varepsilon) \text{ powr } (\text{real } (\text{hgt } (\text{pseq } (\text{Suc } i))) - \text{real } (\text{hgt } (\text{pseq } i)))$
using eps-gt0 **by** *auto*
qed (*auto simp: add-strict-increasing*)
also have $\dots \leq \Delta \ i / \text{alpha } (\text{hgt } (\text{pseq } (\text{Suc } i)))$
proof –

```

    have alpha (hgt (pseq (Suc i))) ≤ alpha (hgt (pseq i)) * (1 + ε) powr (real
(hgt (pseq (Suc i))) - real (hgt (pseq i)))
      using eps-gt0 hgt-gt0
    by (simp add: alpha-eq divide-right-mono flip: powr-realpow powr-add)
  moreover have 0 ≤ Δ i
    by (simp add: Δ0 ΔΔ-ge0 <i ∈ S>)
  moreover have 0 < alpha (hgt (pseq (Suc i)))
    by (simp add: alpha-gt0 hgt-gt0 kn0)
  ultimately show ?thesis
    by (simp add: divide-left-mono)
qed
also have ... ≤ ?R
  unfolding 33 sum-divide-distrib
proof (intro sum-mono)
  fix h
  assume h: h ∈ {1..maxh}
  show ΔΔ i h / alpha (hgt (pseq (Suc i))) ≤ ΔΔ i h / alpha h
  proof (cases hgt (pseq i) ≤ hgt (pseq (Suc i)) ∧ hgt (pseq (Suc i)) < h)
    case False
    then consider hgt (pseq i) > hgt (pseq (Suc i)) | hgt (pseq (Suc i)) ≥ h
      by linarith
    then show ?thesis
  proof cases
    case 1
    then show ?thesis
      using R53 <i ∈ S> hgt-mono' kn0 by force
  next
    case 2
    have alpha h ≤ alpha (hgt (pseq (Suc i)))
      using 2 alpha-mono h by auto
    moreover have 0 ≤ ΔΔ i h
      using ΔΔ-ge0 <i ∈ S> h by presburger
    moreover have 0 < alpha h
      using h kn0 by (simp add: alpha-gt0 hgt-gt0)
    ultimately show ?thesis
      by (simp add: divide-left-mono)
  qed
  qed (auto simp: ΔΔ-eq-0)
qed
finally show ?thesis .
qed
— now we are able to prove claim 8.2
have oneminus * sum-SS = (∑ i∈dboost-star. oneminus * ((1 - beta i) / beta
i))
  using sum-distrib-left sum-SS-def by blast
also have ... ≤ (∑ i∈dboost-star. ∑ h=1..maxh. ΔΔ i h / alpha h)
  by (intro sum-mono 35)
also have ... = (∑ h=1..maxh. ∑ i∈dboost-star. ΔΔ i h / alpha h)
  using sum.swap by fastforce

```

also have $\dots \leq (\sum h=1..maxh. \sum i \in \mathcal{S}. \Delta\Delta i h / \alpha h)$
by (*intro sum-mono sum-mono2*) (*auto simp: <dboost-star $\subseteq \mathcal{S}$ > $\Delta\Delta$ -ge0*
alpha-ge0)
finally have 82: *oneminus * sum-SS*
 $\leq (\sum h=1..maxh. \sum i \in \mathcal{S}. \Delta\Delta i h / \alpha h)$.
— leading onto claim 8.3
have $\Delta\alpha$: $-1 \leq \Delta i / \alpha$ (*hgt (pseq i)*) **if** $i \in \mathcal{R}$ **for** i
using *Y-6-4-Red [of i] <i $\in \mathcal{R}$ >*
unfolding Δ -def \mathcal{R} -def
by (*smt (verit, best) hgt-gt0 alpha-gt0 divide-minus-left less-divide-eq-1-pos*)

have $(\sum i \in \mathcal{R}. - (1 + \varepsilon)^2) \leq (\sum i \in \mathcal{R}. \sum h=1..maxh. \Delta\Delta i h / \alpha h)$
proof (*intro sum-mono*)
fix $i :: nat$
assume $i \in \mathcal{R}$
show $-(1 + \varepsilon)^2 \leq (\sum h = 1..maxh. \Delta\Delta i h / \alpha h)$
proof (*cases $\Delta i < 0$*)
case *True*
have $(1 + \varepsilon)^2 * -1 \leq (1 + \varepsilon)^2 * (\Delta i / \alpha (hgt (pseq i)))$
using $\Delta\alpha$
by (*smt (verit, best) power2-less-0 <i $\in \mathcal{R}$ > mult-le-cancel-left2 mult-minus-right*)
also have $\dots \leq (\sum h = 1..maxh. \Delta\Delta i h / \alpha h)$
proof —
have *le0*: $\Delta\Delta i h \leq 0$ **for** h
using *True* **by** (*auto simp: $\Delta\Delta$ -def Δ -def pp-eq*)
have *eq0*: $\Delta\Delta i h = 0$ **if** $1 \leq h < hgt (pseq i) - 2$ **for** h
proof —
have $hgt (pseq i) - 2 \leq hgt (pseq (Suc i))$
using *Y-6-5-Red <16 $\leq k$ > <i $\in \mathcal{R}$ >* **unfolding** \mathcal{R} -def **by** *blast*
then show *?thesis*
using *that pp-less-hgt[of h]* **by** (*auto simp: $\Delta\Delta$ -def pp-def*)
qed
show *?thesis*
unfolding 33 *sum-distrib-left sum-divide-distrib*
proof (*intro sum-mono*)
fix $h :: nat$
assume $h \in \{1..maxh\}$
then have $1 \leq h < hgt (pseq i) - 2$ **by** *auto*
show $(1 + \varepsilon)^2 * (\Delta\Delta i h / \alpha (hgt (pseq i))) \leq \Delta\Delta i h / \alpha h$
proof (*cases h < hgt (pseq i) - 2*)
case *True*
then show *?thesis*
using $<1 \leq h>$ *eq0* **by** *force*
next
case *False*
have $*(1 + \varepsilon) \wedge (hgt (pseq i) - Suc 0) \leq (1 + \varepsilon)^2 * (1 + \varepsilon) \wedge (h -$
Suc 0)
using *False eps-ge0* **unfolding** *power-add [symmetric]*
by (*intro power-increasing*) *auto*

```

      have **:  $(1 + \varepsilon)^2 * \alpha h \geq \alpha (hgt (pseq i))$ 
      using <math>1 \leq h</math> mult-left-mono [OF *, of  $\varepsilon$ ] eps-ge0
      by (simp add: alpha-eq hgt-gt0 mult-ac divide-right-mono)
      show ?thesis
      using le0 alpha-gt0 <math>h \geq 1</math> hgt-gt0 mult-left-mono-neg [OF **, of  $\Delta\Delta$ 
i h]
      by (simp add: divide-simps mult-ac)
    qed
  qed
  qed
  finally show ?thesis
  by linarith
next
case False
then have  $\Delta\Delta i h \geq 0$  for h
  using  $\Delta\Delta$ -def  $\Delta$ -def pp-eq by auto
then have  $(\sum h = 1..maxh. \Delta\Delta i h / \alpha h) \geq 0$ 
  by (simp add: alpha-ge0 sum-nonneg)
then show ?thesis
  by (smt (verit, ccfv-SIG) sum-power2-ge-zero)
qed
qed
then have 83:  $-(1 + \varepsilon)^2 * \text{card } \mathcal{R} \leq (\sum h=1..maxh. \sum i \in \mathcal{R}. \Delta\Delta i h / \alpha h)$ 
  by (simp add: mult.commute sum.swap [of -  $\mathcal{R}$ ])

```

— now to tackle claim 8.4

```

have  $\Delta 0$ :  $\Delta i \geq 0$  if  $i \in \mathcal{D}$  for i
  using Y-6-4-DegreeReg that unfolding  $\mathcal{D}$ -def  $\Delta$ -def by auto

have 39:  $-2 * \varepsilon \text{ powr}(-1/2) \leq (\sum h = 1..maxh. (\Delta\Delta (i-1) h + \Delta\Delta i h) / \alpha h)$  (is ?L  $\leq$  ?R)
  if  $i \in \mathcal{B}$  for i
proof -
  have odd i
    using step-odd that by (force simp: Step-class-insert-NO-MATCH  $\mathcal{B}$ -def)
  then have  $i > 0$ 
    using odd-pos by auto
  show ?thesis
  proof (cases  $\Delta (i-1) + \Delta i \geq 0$ )
  case True
  with <math>i > 0</math> have  $\Delta\Delta (i-1) h + \Delta\Delta i h \geq 0$  if  $h \geq 1$  for h
    by (fastforce simp:  $\Delta\Delta$ -def  $\Delta$ -def pp-eq)
  then have  $(\sum h = 1..maxh. (\Delta\Delta (i-1) h + \Delta\Delta i h) / \alpha h) \geq 0$ 
    by (force simp: alpha-ge0 intro: sum-nonneg)
  then show ?thesis
    by (smt (verit, ccfv-SIG) powr-ge-zero)
  next

```

```

case False
then have  $\Delta\Delta\text{-le0}$ :  $\Delta\Delta (i-1) h + \Delta\Delta i h \leq 0$  if  $h \geq 1$  for  $h$ 
by (smt (verit, best) One-nat-def  $\Delta\Delta\text{-def}$   $\Delta\text{-def}$   $\langle\text{odd } i\rangle$  odd-Suc-minus-one
pp-eq)
have  $hge$ :  $hgt (pseq (Suc i)) \geq hgt (pseq (i-1)) - 2 * \varepsilon \text{ powr } (-1/2)$ 
using bigY65B that Y-6-5-Bblue by (fastforce simp:  $\mathcal{B}\text{-def}$ )
have  $\Delta\Delta 0$ :  $\Delta\Delta (i-1) h + \Delta\Delta i h = 0$  if  $0 < h < hgt (pseq (i-1)) - 2 * \varepsilon \text{ powr } (-1/2)$  for  $h$ 
using  $\langle\text{odd } i\rangle$  that hge unfolding  $\Delta\Delta\text{-def}$  One-nat-def
by (smt (verit) of-nat-less-iff odd-Suc-minus-one powr-non-neg pp-less-hgt)
have big39:  $1/2 \leq (1 + \varepsilon) \text{ powr } (-2 * \varepsilon \text{ powr } (-1/2))$ 
using big l-le-k by (auto simp: Big-ZZ-8-1-def Big39-def)
have  $?L * \alpha (hgt (pseq (i-1))) * (1 + \varepsilon) \text{ powr } (-2 * \varepsilon \text{ powr } (-1/2))$ 
 $\leq -(\varepsilon \text{ powr } (-1/2)) * \alpha (hgt (pseq (i-1)))$ 
using mult-left-mono-neg [OF big39, of  $-(\varepsilon \text{ powr } (-1/2)) * \alpha (hgt$ 
(pseq (i-1))) / 2]
using alpha-ge0 [of hgt (pseq (i-1))] eps-ge0
by (simp add: mult-ac)
also have  $\dots \leq \Delta (i-1) + \Delta i$ 
proof -
have  $pseq (Suc i) \geq pseq (i-1) - (\varepsilon \text{ powr } (-1/2)) * \alpha (hgt (pseq$ 
(i-1)))
using Y-6-4-Bblue that  $\mathcal{B}\text{-def}$  by blast
with  $\langle i > 0 \rangle$  show ?thesis
by (simp add:  $\Delta\text{-def}$ )
qed
finally have  $?L * \alpha (hgt (pseq (i-1))) * (1 + \varepsilon) \text{ powr } (-2 * \varepsilon \text{ powr } (-1/2)) \leq \Delta (i-1) + \Delta i$  .
then have  $?L \leq (1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/2)) * (\Delta (i-1) + \Delta i) / \alpha (hgt (pseq (i-1)))$ 
using alpha-ge0 [of hgt (pseq (i-1))] eps-ge0
by (simp add: powr-minus divide-simps mult-ac)
also have  $\dots \leq ?R$ 
proof -
have  $(1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/2)) * (\Delta\Delta (i - Suc 0) h + \Delta\Delta i h) / \alpha (hgt (pseq (i - Suc 0)))$ 
 $\leq (\Delta\Delta (i - Suc 0) h + \Delta\Delta i h) / \alpha h$ 
if  $h$ :  $Suc 0 \leq h$   $h \leq \text{max } h$  for  $h$ 
proof (cases  $h < hgt (pseq (i-1)) - 2 * \varepsilon \text{ powr } (-1/2)$ )
case False
then have  $hgt (pseq (i-1)) - 1 \leq 2 * \varepsilon \text{ powr } (-1/2) + (h - 1)$ 
using hgt-gt0 by (simp add: nat-less-real-le)
then have  $*$ :  $(1 + \varepsilon) \text{ powr } (2 * \varepsilon \text{ powr } (-1/2)) / \alpha (hgt (pseq (i-1))) \geq 1 / \alpha h$ 
using that eps-gt0 kn0 hgt-gt0
by (simp add: alpha-eq divide-simps flip: powr-realpow powr-add)
show ?thesis
using mult-left-mono-neg [OF *  $\Delta\Delta\text{-le0}$ ] that by (simp add: Groups.mult-ac)

```

```

    qed (use h  $\Delta\Delta 0$  in auto)
  then show ?thesis
    by (force simp: 33 sum-distrib-left sum-divide-distrib simp flip: sum.distrib
intro: sum-mono)
  qed
  finally show ?thesis .
  qed
  qed

have B34: card  $\mathcal{B} \leq k \text{ powr } (3/4)$ 
  by (smt (verit) card $\mathcal{B}$  l-le-k of-nat-0-le-iff of-nat-mono powr-mono2 zero-le-divide-iff)
have  $-2 * k \text{ powr } (7/8) \leq -2 * \varepsilon \text{ powr } (-1/2) * k \text{ powr } (3/4)$ 
  by (simp add: eps-def powr-powr flip: powr-add)
also have  $\dots \leq -2 * \varepsilon \text{ powr } (-1/2) * \text{card } \mathcal{B}$ 
  using B34 by (intro mult-left-mono-neg powr-mono2) auto
also have  $\dots = (\sum_{i \in \mathcal{B}} -2 * \varepsilon \text{ powr } (-1/2))$ 
  by simp
also have  $\dots \leq (\sum_{h=1..maxh} \sum_{i \in \mathcal{B}} (\Delta\Delta (i-1) h + \Delta\Delta i h) / \text{alpha } h)$ 
  unfolding sum.swap [of  $\mathcal{B}$ ] by (intro sum-mono 39)
also have  $\dots \leq (\sum_{h=1..maxh} \sum_{i \in \mathcal{B} \cup \mathcal{D}} \Delta\Delta i h / \text{alpha } h)$ 
  proof (intro sum-mono)
    fix h
    assume  $h \in \{1..maxh\}$ 
    have  $\mathcal{B} \subseteq \{0<..\}$ 
    using odd-pos [OF step-odd] by (auto simp:  $\mathcal{B}$ -def Step-class-insert-NO-MATCH)
    with inj-on-diff-nat [of  $\mathcal{B}$  1] have inj-pred: inj-on  $(\lambda i. i - \text{Suc } 0) \mathcal{B}$ 
      by (simp add: Suc-leI subset-eq)
    have  $(\sum_{i \in \mathcal{B}} \Delta\Delta (i - \text{Suc } 0) h) = (\sum_{i \in (\lambda i. i-1) \text{ ' } \mathcal{B}} \Delta\Delta i h)$ 
      by (simp add: sum.reindex [OF inj-pred])
    also have  $\dots \leq (\sum_{i \in \mathcal{D}} \Delta\Delta i h)$ 
    proof (intro sum-mono2)
      show  $(\lambda i. i - 1) \text{ ' } \mathcal{B} \subseteq \mathcal{D}$ 
      by (force simp:  $\mathcal{D}$ -def  $\mathcal{B}$ -def Step-class-insert-NO-MATCH intro: dreg-before-step')
      show  $0 \leq \Delta\Delta i h$  if  $i \in \mathcal{D} \setminus (\lambda i. i - 1) \text{ ' } \mathcal{B}$  for i
      using that  $\Delta 0$   $\Delta\Delta$ -def  $\Delta$ -def pp-eq by fastforce
    qed auto
    finally have  $(\sum_{i \in \mathcal{B}} \Delta\Delta (i - \text{Suc } 0) h) \leq (\sum_{i \in \mathcal{D}} \Delta\Delta i h)$  .
    with alpha-ge0 [of h]
    show  $(\sum_{i \in \mathcal{B}} (\Delta\Delta (i - 1) h + \Delta\Delta i h) / \text{alpha } h) \leq (\sum_{i \in \mathcal{B} \cup \mathcal{D}} \Delta\Delta i h / \text{alpha } h)$ 
      by (simp add: BD-disj divide-right-mono sum.distrib sum.union-disjoint flip:
sum-divide-distrib)
    qed
  finally have 84:  $-2 * k \text{ powr } (7/8) \leq (\sum_{h=1..maxh} \sum_{i \in \mathcal{B} \cup \mathcal{D}} \Delta\Delta i h / \text{alpha } h)$  .

have m-eq:  $\{..<\text{halted-point}\} = \mathcal{R} \cup \mathcal{S} \cup (\mathcal{B} \cup \mathcal{D})$ 
  using before-halted-eq by (auto simp:  $\mathcal{B}$ -def  $\mathcal{D}$ -def  $\mathcal{S}$ -def  $\mathcal{R}$ -def Step-class-insert-NO-MATCH)

```

have $-(1 + \varepsilon)^2 * \text{real}(\text{card } \mathcal{R})$
 $+ \text{oneminus} * \text{sum-SS}$
 $- 2 * \text{real } k \text{ powr } (7/8) \leq (\sum h = \text{Suc } 0..maxh. \sum i \in \mathcal{R}. \Delta\Delta i h / \text{alpha } h)$
 $+ (\sum h = \text{Suc } 0..maxh. \sum i \in \mathcal{S}. \Delta\Delta i h / \text{alpha } h)$
 $+ (\sum h = \text{Suc } 0..maxh. \sum i \in \mathcal{B} \cup \mathcal{D}. \Delta\Delta i h / \text{alpha } h)$
using 82 83 84 **by** *simp*
also have $\dots = (\sum h = \text{Suc } 0..maxh. \sum i \in \mathcal{R} \cup \mathcal{S} \cup (\mathcal{B} \cup \mathcal{D}). \Delta\Delta i h / \text{alpha } h)$
by (*simp add: sum.distrib disj sum.union-disjoint Int-Un-distrib Int-Un-distrib2*)
also have $\dots \leq 1 + 2 * \ln(\text{real } k) / \varepsilon$
using 34 **by** (*simp add: m-eq*)
finally
have 41: $\text{oneminus} * \text{sum-SS} - (1 + \varepsilon)^2 * \text{card } \mathcal{R} - 2 * k \text{ powr } (7/8)$
 $\leq 1 + 2 * \ln k / \varepsilon$
by *simp*
have big42: $(1 + \varepsilon)^2 / \text{oneminus} \leq 1 + 2 * k \text{ powr } (-1/16)$
 $2 * k \text{ powr } (-1/16) * k$
 $+ (1 + 2 * \ln k / \varepsilon + 2 * k \text{ powr } (7/8)) / \text{oneminus}$
 $\leq \text{real } k \text{ powr } (19/20)$
using big-l-le-k **by** (*auto simp: Big-ZZ-8-1-def Big42a-def Big42b-def oneminus-def*)
have $\text{oneminus} > 0$
using $\langle 16 \leq k \rangle$ *eps-gt0 eps-less1 powr01-less-one* **by** (*auto simp: oneminus-def*)
with 41 **have** *sum-SS*
 $\leq (1 + 2 * \ln k / \varepsilon + (1 + \varepsilon)^2 * \text{card } \mathcal{R} + 2 * k \text{ powr } (7/8)) / \text{oneminus}$
by (*simp add: mult-ac pos-le-divide-eq diff-le-eq*)
also have $\dots \leq \text{card } \mathcal{R} * (((1 + \varepsilon)^2) / \text{oneminus})$
 $+ (1 + 2 * \ln k / \varepsilon + 2 * k \text{ powr } (7/8)) / \text{oneminus}$
by (*simp add: field-simps add-divide-distrib*)
also have $\dots \leq \text{card } \mathcal{R} * (1 + 2 * k \text{ powr } (-1/16))$
 $+ (1 + 2 * \ln k / \varepsilon + 2 * k \text{ powr } (7/8)) / \text{oneminus}$
using big42 $\langle \text{oneminus} > 0 \rangle$ **by** (*intro add-mono mult-mono auto*)
also have $\dots \leq \text{card } \mathcal{R} + 2 * k \text{ powr } (-1/16) * k$
 $+ (1 + 2 * \ln k / \varepsilon + 2 * k \text{ powr } (7/8)) / \text{oneminus}$
using $\langle \text{card } \mathcal{R} < k \rangle$ **by** (*intro add-mono mult-mono auto simp: algebra-simps*)
also have $\dots \leq \text{real}(\text{card } \mathcal{R}) + \text{real } k \text{ powr } (19/20)$
using big42 **by** *force*
finally show *?thesis* .
qed

7.2 Lemma 8.5

An inequality that pops up in the proof of (39)

definition *inequality85* $\equiv \lambda k. 3 * \text{eps } k \text{ powr } (1/4) * k \leq k \text{ powr } (19/20)$

definition *Big-ZZ-8-5* \equiv

$\lambda \mu l. \text{Big-X-7-5 } \mu l \wedge \text{Big-ZZ-8-1 } \mu l \wedge \text{Big-Red-5-3 } \mu l$
 $\wedge (\forall k \geq l. \text{inequality85 } k)$

lemma *Big-ZZ-8-5*:

```

assumes  $0 < \mu_0 \ \mu_1 < 1$ 
shows  $\forall^\infty l. \forall \mu. \mu \in \{\mu_0.. \mu_1\} \longrightarrow \text{Big-ZZ-8-5 } \mu \ l$ 
using assms Big-Red-5-3 Big-X-7-5 Big-ZZ-8-1
unfolding Big-ZZ-8-5-def inequality85-def eps-def
apply (simp add: eventually-conj-iff all-imp-conj-distrib)
apply (intro conjI strip eventually-all-ge-at-top; real-asymp)
done

lemma (in Book) ZZ-8-5:
assumes big: Big-ZZ-8-5  $\mu \ l$ 
defines  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$  and  $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$ 
shows  $\text{card } \mathcal{S} \leq (\text{bigbeta} / (1 - \text{bigbeta})) * \text{card } \mathcal{R}$ 
 $+ (2 / (1 - \mu)) * k \text{ powr } (19/20)$ 
proof -
have [simp]: finite  $\mathcal{S}$ 
by (simp add: S-def)
moreover have dboost-star  $\subseteq \mathcal{S}$ 
by (auto simp: dboost-star-def S-def)
ultimately have real ( $\text{card } \mathcal{S} - \text{card } \text{dboost-star} = \text{card } (\mathcal{S} \setminus \text{dboost-star})$ )
by (metis card-Diff-subset card-mono finite-subset of-nat-diff)
also have  $\dots \leq 3 * \epsilon \text{ powr } (1/4) * k$ 
using  $\mu_01$  big X-7-5 by (auto simp: Big-ZZ-8-5-def dboost-star-def S-def)
also have  $\dots \leq k \text{ powr } (19/20)$ 
using big l-le-k by (auto simp: Big-ZZ-8-5-def inequality85-def)
finally have  $*$ :  $\text{real } (\text{card } \mathcal{S}) - \text{card } \text{dboost-star} \leq k \text{ powr } (19/20)$  .
have bigbeta-lt1:  $\text{bigbeta} < 1$  and bigbeta-gt0:  $0 < \text{bigbeta}$  and beta-gt0:  $\bigwedge i. i \in \mathcal{S} \implies \text{beta } i > 0$ 
using bigbeta-ge0 big by (auto simp: Big-ZZ-8-5-def S-def beta-gt0 bigbeta-gt0 bigbeta-less1)
then have ge0:  $\text{bigbeta} / (1 - \text{bigbeta}) \geq 0$ 
by auto
show ?thesis
proof (cases dboost-star = {})
case True
with  $*$  have  $\text{card } \mathcal{S} \leq k \text{ powr } (19/20)$ 
by simp
also have  $\dots \leq (2 / (1 - \mu)) * k \text{ powr } (19/20)$ 
using  $\mu_01$  kn0 by (simp add: divide-simps)
finally show ?thesis
by (smt (verit, ccfv-SIG) mult-nonneg-nonneg of-nat-0-le-iff ge0)
next
case False
have bb-le:  $\text{bigbeta} \leq \mu$ 
using big bigbeta-le by (auto simp: Big-ZZ-8-5-def)
have  $(\text{card } \mathcal{S} - k \text{ powr } (19/20)) / \text{bigbeta} \leq \text{card } \text{dboost-star} / \text{bigbeta}$ 
by (smt (verit) * bigbeta-ge0 divide-right-mono)
also have  $\dots = (\sum_{i \in \text{dboost-star}} 1 / \text{beta } i)$ 
proof (cases card dboost-star = 0)
case False

```

```

then show ?thesis
  by (simp add: bigbeta-def Let-def inverse-eq-divide)
qed (simp add: False card-eq-0-iff)
also have ... ≤ real(card dboost-star) + card  $\mathcal{R}$  + k powr (19/20)
proof -
  have ( $\sum_{i \in \text{dboost-star}} (1 - \text{beta } i) / \text{beta } i$ )
    ≤ real (card  $\mathcal{R}$ ) + k powr (19/20)
    using ZZ-8-1 big unfolding Big-ZZ-8-5-def  $\mathcal{R}$ -def by blast
  moreover have ( $\sum_{i \in \text{dboost-star}} \text{beta } i / \text{beta } i$ ) = ( $\sum_{i \in \text{dboost-star}} 1$ )
    using <dboost-star ⊆  $\mathcal{S}$ > beta-gt0 by (intro sum.cong) force+
  ultimately show ?thesis
    by (simp add: field-simps diff-divide-distrib sum-subtractf)
qed
also have ... ≤ real(card  $\mathcal{S}$ ) + card  $\mathcal{R}$  + k powr (19/20)
  by (simp add: <dboost-star ⊆  $\mathcal{S}$ > card-mono)
finally have (card  $\mathcal{S}$  - k powr (19/20)) / bigbeta ≤ real (card  $\mathcal{S}$ ) + card  $\mathcal{R}$ 
+ k powr (19/20) .
  then have card  $\mathcal{S}$  - k powr (19/20) ≤ (real (card  $\mathcal{S}$ ) + card  $\mathcal{R}$  + k powr
(19/20)) * bigbeta
    using bigbeta-gt0 by (simp add: field-simps)
  then have card  $\mathcal{S}$  * (1 - bigbeta) ≤ bigbeta * card  $\mathcal{R}$  + (1 + bigbeta) * k
powr (19/20)
    by (simp add: algebra-simps)
  then have card  $\mathcal{S}$  ≤ (bigbeta * card  $\mathcal{R}$  + (1 + bigbeta) * k powr (19/20)) /
(1 - bigbeta)
    using bigbeta-lt1 by (simp add: field-simps)
  also have ... = (bigbeta / (1 - bigbeta)) * card  $\mathcal{R}$ 
+ ((1 + bigbeta) / (1 - bigbeta)) * k powr (19/20)
    using bigbeta-gt0 bigbeta-lt1 by (simp add: divide-simps)
  also have ... ≤ (bigbeta / (1 - bigbeta)) * card  $\mathcal{R}$  + (2 / (1- $\mu$ )) * k powr
(19/20)
    using  $\mu$ 01 bb-le by (intro add-mono order-refl mult-right-mono frac-le) auto
  finally show ?thesis .
qed
qed

```

7.3 Lemma 8.6

For some reason this was harder than it should have been. It does require a further small limit argument.

definition *Big-ZZ-8-6* ≡

$$\lambda \mu l. \text{Big-ZZ-8-5 } \mu l \wedge (\forall k \geq l. 2 / (1 - \mu) * k \text{ powr } (19/20) < k \text{ powr } (39/40))$$

lemma *Big-ZZ-8-6*:

assumes $0 < \mu < 1$

shows $\forall^\infty l. \forall \mu. \mu \in \{\mu_0.. \mu_1\} \longrightarrow \text{Big-ZZ-8-6 } \mu l$

using *assms Big-ZZ-8-5*

unfolding *Big-ZZ-8-6-def*

apply (simp add: eventually-conj-iff all-imp-conj-distrib)

apply (*intro conjI strip eventually-all-ge-at-top eventually-all-geI1* [**where** $L=1$])

apply *real-asymp*
by (*smt (verit, ccfv-SIG) frac-le powr-ge-zero*)

lemma (*in Book*) *ZZ-8-6*:
assumes *big: Big-ZZ-8-6* μ l
defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ **and** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
and $a \equiv 2 / (1 - \mu)$
assumes *s-ge: card* $\mathcal{S} \geq k \text{ powr } (39/40)$
shows $\text{bigbeta} \geq (1 - a * k \text{ powr } (-1/40)) * (\text{card } \mathcal{S} / (\text{card } \mathcal{S} + \text{card } \mathcal{R}))$
proof –
have *bigbeta-lt1: bigbeta* < 1 **and** *bigbeta-gt0: 0* $<$ *bigbeta*
using *bigbeta-ge0 big*
by (*auto simp: Big-ZZ-8-6-def Big-ZZ-8-5-def bigbeta-less1 bigbeta-gt0* *S-def*)
have $a > 0$
using $\mu 01$ **by** (*simp add: a-def*)
have *s-gt-a: a* $* k \text{ powr } (19/20) <$ *card* \mathcal{S}
and *85: card* $\mathcal{S} \leq (\text{bigbeta} / (1 - \text{bigbeta})) * \text{card } \mathcal{R} + a * k \text{ powr } (19/20)$
using *big l-le-k assms*
unfolding *R-def S-def a-def Big-ZZ-8-6-def* **by** (*fastforce intro: ZZ-8-5*)
then have *t-non0: card* $\mathcal{R} \neq 0$ — *seemingly not provable without our assumption*
using *mult-eq-0-iff* **by** *fastforce*
then have $(\text{card } \mathcal{S} - a * k \text{ powr } (19/20)) / \text{card } \mathcal{R} \leq \text{bigbeta} / (1 - \text{bigbeta})$
using *85 bigbeta-gt0 bigbeta-lt1 t-non0* **by** (*simp add: pos-divide-le-eq*)
then have $\text{bigbeta} \geq (1 - \text{bigbeta}) * (\text{card } \mathcal{S} - a * k \text{ powr } (19/20)) / \text{card } \mathcal{R}$
by (*smt (verit, ccfv-threshold) bigbeta-lt1 mult.commute le-divide-eq times-divide-eq-left*)
then have $*$: $\text{bigbeta} * (\text{card } \mathcal{R} + \text{card } \mathcal{S} - a * k \text{ powr } (19/20)) \geq \text{card } \mathcal{S} - a * k \text{ powr } (19/20)$
using *t-non0* **by** (*simp add: field-simps*)
have $(1 - a * k \text{ powr } (-1/40)) * \text{card } \mathcal{S} \leq \text{card } \mathcal{S} - a * k \text{ powr } (19/20)$
using *s-ge kn0 <a>0> t-non0* **by** (*simp add: powr-minus field-simps flip: powr-add*)
then have $(1 - a * k \text{ powr } (-1/40)) * (\text{card } \mathcal{S} / (\text{card } \mathcal{S} + \text{card } \mathcal{R}))$
 $\leq (\text{card } \mathcal{S} - a * k \text{ powr } (19/20)) / (\text{card } \mathcal{S} + \text{card } \mathcal{R})$
by (*force simp: divide-right-mono*)
also have $\dots \leq (\text{card } \mathcal{S} - a * k \text{ powr } (19/20)) / (\text{card } \mathcal{R} + \text{card } \mathcal{S} - a * k \text{ powr } (19/20))$
using *s-gt-a <a>0> t-non0* **by** (*intro divide-left-mono*) *auto*
also have $\dots \leq \text{bigbeta}$
using $*$ *s-gt-a*
by (*simp add: divide-simps split: if-split-asm*)
finally show *?thesis* .
qed
end

8 An exponential improvement far from the diagonal

```
theory Far-From-Diagonal
  imports Zigzag Stirling-Formula.Stirling-Formula
```

```
begin
```

8.1 An asymptotic form for binomial coefficients via Stirling's formula

From Appendix D.3, page 56

```
lemma const-smallo-real: ( $\lambda n. x$ )  $\in o(\text{real})$ 
  by real-asymp
```

```
lemma o-real-shift:
  assumes  $f \in o(\text{real})$ 
  shows ( $\lambda i. f(i+j)$ )  $\in o(\text{real})$ 
  unfolding smallo-def
proof clarify
  fix  $c :: \text{real}$ 
  assume ( $0 :: \text{real}$ )  $< c$ 
  then have *:  $\forall_F i$  in sequentially.  $\text{norm } (f i) \leq c/2 * \text{norm } i$ 
    using assms half-gt-zero landau-o.smallD by blast
  have  $\forall_F i$  in sequentially.  $\text{norm } (f (i + j)) \leq c/2 * \text{norm } (i + j)$ 
    using eventually-all-ge-at-top [OF *]
    by (metis (mono-tags, lifting) eventually-sequentially le-add1)
  then have  $\forall_F i$  in sequentially.  $i \geq j \longrightarrow \text{norm } (f (i + j)) \leq c * \text{norm } i$ 
    apply eventually-elim
    apply clarsimp
    by (smt (verit, best)  $<0 < c$ ) mult-left-mono nat-distrib(2) of-nat-mono)
  then show  $\forall_F i$  in sequentially.  $\text{norm } (f (i + j)) \leq c * \text{norm } i$ 
    using eventually-mp by fastforce
```

```
qed
```

```
lemma tendsto-zero-imp-o1:
  fixes  $a :: \text{nat} \Rightarrow \text{real}$ 
  assumes  $a \longrightarrow 0$ 
  shows  $a \in o(1)$ 
proof -
  have  $\forall_F n$  in sequentially.  $|a n| \leq c$  if  $c > 0$  for  $c$ 
    using assms order-tendstoD(2) tendsto-rabs-zero-iff eventually-sequentially less-eq-real-def
  that
    by metis
  then show ?thesis
    by (auto simp: smallo-def)
qed
```

8.2 Fact D.3 from the Appendix

And hence, Fact 9.4

definition $stir \equiv \lambda n. fact\ n / (sqrt\ (2*pi*n) * (n / exp\ 1) ^ n) - 1$

Generalised to the reals to allow derivatives

definition $stirG \equiv \lambda n. Gamma\ (n+1) / (sqrt\ (2*pi*n) * (n / exp\ 1) powr\ n) - 1$

lemma $stir\ eq\ stirG: n > 0 \implies stir\ n = stirG\ (real\ n)$
by (*simp add: stirG-def stir-def add.commute powr-realpow Gamma-fact*)

lemma $stir\ ge\ 0: n > 0 \implies stir\ n \geq 0$
using *fact-bounds[of n]* **by** (*simp add: stir-def*)

lemma $stir\ to\ 0: stir \longrightarrow 0$
using *fact-asymp-equiv* **by** (*simp add: asymp-equiv-def stir-def LIM-zero*)

lemma $stir\ o1: stir \in o(1)$
using *stir-to-0 tendsto-zero-imp-o1* **by** *presburger*

lemma $fact\ eq\ stir\ times: n \neq 0 \implies fact\ n = (1 + stir\ n) * (sqrt\ (2*pi*n) * (n / exp\ 1) ^ n)$
by (*simp add: stir-def*)

definition $logstir \equiv \lambda n. if\ n=0\ then\ 0\ else\ log\ 2\ ((1 + stir\ n) * sqrt\ (2*pi*n))$

lemma $logstir\ o\ real: logstir \in o(real)$

proof –

have $\forall^\infty n. 0 < n \longrightarrow |log\ 2\ ((1 + stir\ n) * sqrt\ (2*pi*n))| \leq c * real\ n$ **if** $c > 0$
for c

proof –

have $\forall^\infty n. 2\ powr\ (c*n) / sqrt\ (2*pi*n) \geq c+1$

using *that* **by** *real-asymp*

moreover **have** $\forall^\infty n. |stir\ n| \leq c$

using *stir-o1* **that** **by** (*auto simp: smallo-def*)

ultimately **have** $\forall^\infty n. ((1 + stir\ n) * sqrt\ (2*pi*n)) \leq 2\ powr\ (c * n)$

proof *eventually-elim*

fix n

assume $c1: c+1 \leq 2\ powr\ (c * n) / sqrt\ (2*pi*n)$ **and** $lec: |stir\ n| \leq c$

then **have** $stir\ n \leq c$

by *auto*

then **show** $(1 + stir\ n) * sqrt\ (2*pi*n) \leq 2\ powr\ (c*n)$

using *mult-right-mono [OF c1, of sqrt (2*pi*n)] lec*

by (*smt (verit, ccfv-SIG) c1 mult-right-mono nonzero-eq-divide-eq pos-prod-le powr-gt-zero*)

qed

then **show** *?thesis*

proof (*eventually-elim, clarify*)

```

fix n
assume n: (1 + stir n) * sqrt (2 * pi * n) ≤ 2 powr (c * n)
  and n > 0
have (1 + stir n) * sqrt (2 * pi * real n) ≥ 1
  using stir-ge0 <0 < n> mult-ge1-I pi-ge-two by auto
with n show |log 2 ((1 + stir n) * sqrt (2 * pi * n))| ≤ c * n
  by (simp add: abs-if le-powr-iff)
qed
qed
then show ?thesis
  by (auto simp: smallo-def logstir-def)
qed

```

```

lemma logfact-eq-stir-times:
  fact n = 2 powr (logstir n) * (n / exp 1) ^ n
proof -
  have 1 + stir n > 0 if n ≠ 0
  using that by (simp add: stir-def)
  then show ?thesis
  by (simp add: logstir-def fact-eq-stir-times)
qed

```

```

lemma mono-G:
  defines G ≡ (λx::real. Gamma (x + 1) / (x / exp 1) powr x)
  shows mono-on {0<..} G
  unfolding monotone-on-def
proof (intro strip)
  fix x y::real
  assume x: x ∈ {0<..} x ≤ y
  define GD where GD ≡ λu::real. Gamma(u+1) * (Digamma(u+1) - ln(u))
  / (u / exp 1) powr u
  have *: ∃ D. (G has-real-derivative D) (at u) ∧ D > 0 if 0 < u for u
  proof (intro exI conjI)
    show (G has-real-derivative GD u) (at u)
    unfolding G-def GD-def
    using that
    by (force intro!: derivative-eq-intros has-real-derivative-powr' simp: ln-div
  pos-prod-lt field-simps)
    show GD u > 0
    using that by (auto simp: GD-def Digamma-plus-1-gt-ln) — Thank you,
  Manuel!
  qed
  show G x ≤ G y
  using x DERIV-pos-imp-increasing [OF - *] by (force simp: less-eq-real-def)
qed

```

```

lemma mono-logstir: mono logstir
  unfolding monotone-on-def
proof (intro strip)

```

```

fix i j::nat
assume i ≤ j
show logstir i ≤ logstir j
proof (cases j=0)
  case True
    with ⟨i ≤ j⟩ show ?thesis
    by auto
  next
    case False
      with pi-ge-two have 1 * 1 ≤ 2 * pi * j
      by (intro mult-mono) auto
      with False stir-ge0 [of j] have *: 1 * 1 ≤ (1 + stir j) * sqrt (2 * pi * real j)
      by (intro mult-mono) auto
      with ⟨i ≤ j⟩ mono-G show ?thesis
      by (auto simp: logstir-def stir-eq-stirG stirG-def monotone-on-def)
qed
qed

```

definition ok-fun-94 $\equiv \lambda k. - \logstir k$

lemma ok-fun-94: ok-fun-94 $\in o(\text{real})$
unfolding ok-fun-94-def
using logstir-o-real **by** simp

lemma fact-9-4:
assumes l: 0 < l l ≤ k
defines $\gamma \equiv l / (\text{real } k + \text{real } l)$
shows k+l choose l ≥ 2 powr ok-fun-94 k * γ powr (-l) * (1- γ) powr (-k)
proof -
have *: ok-fun-94 k ≤ logstir (k+l) - (logstir k + logstir l)
using mono-logstir **by** (auto simp: ok-fun-94-def monotone-def)
have 2 powr ok-fun-94 k * γ powr (- real l) * (1- γ) powr (- real k)
= (2 powr ok-fun-94 k) * (k+l) powr(k+l) / (k powr k * l powr l)
by (simp add: γ -def powr-minus powr-add powr-divide divide-simps)
also have ... ≤ (2 powr (logstir (k+l))) / (2 powr (logstir k) * 2 powr (logstir l))
* (k+l) powr (k+l) / (k powr k * l powr l)
by (smt (verit, del-insts) * divide-right-mono mult-less-0-iff mult-right-mono
powr-add powr-diff powr-ge-zero powr-mono)
also have ... = fact(k+l) / (fact k * fact l)
using l **by** (simp add: logfact-eq-stir-times powr-add divide-simps flip: powr-realpow)
also have ... = real (k+l choose l)
by (simp add: binomial-fact)
finally show ?thesis .
qed

8.3 Fact D.2

For Fact 9.6

lemma D2:
fixes $k\ l$
assumes $t: 0 < t \leq k$
defines $\gamma \equiv l / (\text{real } k + \text{real } l)$
shows $(k+l-t \text{ choose } l) \leq \exp(-\gamma * (t-1)^2 / (2*k)) * (k / (k+l))^{t-1} * (k+l \text{ choose } l)$
proof –
have $(k+l-t \text{ choose } l) * \text{inverse } (k+l \text{ choose } l) = (\prod_{i < t}. (k-i) / (k+l-i))$
using $\langle t \leq k \rangle$
proof (*induction t*)
case (*Suc t*)
then have $t \leq k$
by *simp*
have $(k+l-t) * (k+l - \text{Suc } t \text{ choose } l) = (k-t) * (k+l-t \text{ choose } l)$
by (*metis binomial-absorb-comp diff-Suc-eq-diff-pred diff-add-inverse2 diff-commute*)
with *Suc.IH* [*symmetric*] *Suc(2)* **show** *?case*
by (*simp add: field-simps flip: of-nat-mult of-nat-diff*)
qed auto
also have $\dots = (\text{real } k / (k+l))^{t-1} * (\prod_{i < t}. 1 - \text{real } i * \text{real } l / (\text{real } k * (k+l-i)))$
proof –
have $1 - i * \text{real } l / (\text{real } k * (k+l-i)) = ((k-i)/(k+l-i)) * ((k+l) / k)$
if $i < t$ **for** i
using *that* $\langle t \leq k \rangle$ **by** (*simp add: divide-simps*) *argo*
then have $*$: $(\prod_{i < t}. 1 - \text{real } i * \text{real } l / (\text{real } k * (k+l-i))) = (\prod_{i < t}. ((k-i)/(k+l-i)) * ((k+l) / k))$
by *auto*
show *?thesis*
unfolding $*$ *prod.distrib* **by** (*simp add: power-divide*)
qed
also have $\dots \leq (\text{real } k / (k+l))^{t-1} * \exp(-(\sum_{i < t}. \text{real } i * \text{real } l / (\text{real } k * (k+l-i))))$
proof (*intro mult-left-mono*)
have $\text{real } i * \text{real } l / (\text{real } k * \text{real } (k+l-i)) \leq 1$
if $i < t$ **for** i
using *that* $\langle t \leq k \rangle$ **by** (*simp add: divide-simps mult-mono*)
moreover have $1 - i * l / (k * \text{real } (k+l-i)) \leq \exp(- (i * \text{real } l / (k * (k + \text{real } l))))$ (*is - ≤ ?R*)
if $i < t$ **for** i
proof –
have $\exp(- (i * l / (k * \text{real } (k+l-i)))) \leq ?R$
using *that* $\langle t \leq k \rangle$ **by** (*simp add: frac-le-eq divide-le-0-iff mult-mono*)
with *exp-minus-ge* **show** *?thesis*
by (*smt (verit, best)*)
qed
ultimately show $(\prod_{i < t}. 1 - i * \text{real } l / (k * \text{real } (k+l-i))) \leq \exp(-(\sum_{i < t}. i * \text{real } l / (k * \text{real } (k+l-i))))$
by (*force simp: exp-sum simp flip: sum-negf intro!: prod-mono*)
qed auto

finally have 1: $(k+l-t \text{ choose } l) * \text{inverse } (k+l \text{ choose } l)$
 $\leq (\text{real } k / (k+l)) ^t * \text{exp } (- (\sum_{i<t} i * \gamma / k))$
by (*simp add: γ -def mult.commute*)
have **: $\gamma * (t-1)^2 / (2*k) \leq (\sum_{i<t} i * \gamma / k)$
proof –
have g: $(\sum_{i<t} \text{real } i) = \text{real } (t*(t-1)) / 2$
by (*induction t*) (*auto simp: algebra-simps eval-nat-numeral*)
have $\gamma * (t-1)^2 / (2*k) \leq \text{real}(t*(t-1)) / 2 * \gamma/k$
by (*simp add: field-simps eval-nat-numeral divide-right-mono mult-mono*
 γ -def)
also have ... = $(\sum_{i<t} i * \gamma / k)$
unfolding g [*symmetric*] **by** (*simp add: sum-distrib-right sum-divide-distrib*)
finally show ?thesis .
qed
have 0: $0 \leq \text{real } (k + l \text{ choose } l)$
by *simp*
have *: $(k+l-t \text{ choose } l) \leq (k / (k+l)) ^t * \text{exp } (- (\sum_{i<t} i * \gamma / k)) * (k+l$
choose l)
using *order-trans [OF - mult-right-mono [OF 1 0]]*
by (*simp add: less-eq-real-def*)
also have ... $\leq (k / (k+l)) ^t * \text{exp } (- \gamma * (t-1)^2 / (2*k)) * (k+l \text{ choose } l)$
using ** **by** (*intro mult-mono*) *auto*
also have ... $\leq \text{exp } (- \gamma * (t-1)^2 / (2 * \text{real } k)) * (k / (k+l)) ^t * (k+l$
choose l)
by (*simp add: mult-ac*)
finally show ?thesis
using t **by** *simp*
qed

Statement borrowed from Bhavik; no $o(k)$ function

corollary *Far-9-6*:

fixes $k \ l$
assumes $t: 0 < t \leq k$
defines $\gamma \equiv l / (k + \text{real } l)$
shows $\text{exp } (-1) * (1-\gamma) \text{ powr } (- \text{real } t) * \text{exp } (\gamma * (\text{real } t)^2 / \text{real}(2*k)) * (k-t+l \text{ choose } l) \leq (k+l \text{ choose } l)$
proof –
have kkl: $k / (k + \text{real } l) = 1 - \gamma \ k+l-t = k-t+l$
using t **by** (*auto simp: γ -def divide-simps*)
have [*simp*]: $t + t \leq \text{Suc } (t * t)$
using t
by (*metis One-nat-def Suc-leI mult-2 mult-right-mono nle-le not-less-eq-eq numeral-2-eq-2 mult-1-right*)
have $0 \leq \gamma \ \gamma < 1$
using t **by** (*auto simp: γ -def*)
then have $\gamma * (\text{real } t * 2) \leq \gamma + \text{real } k * 2$
using t **by** (*smt (verit, best) mult-less-cancel-right2 of-nat-0-less-iff of-nat-mono*)
then have *: $\gamma * t^2 / (2*k) - 1 \leq \gamma * (t-1)^2 / (2*k)$
using t

apply (*simp add: power2-eq-square pos-divide-le-eq divide-simps*)
apply (*simp add: algebra-simps*)
done
then have *: $\exp(-1) * \exp(\gamma * t^2 / (2*k)) \leq \exp(\gamma * (t-1)^2 / (2*k))$
by (*metis exp-add exp-le-cancel-iff uminus-add-conv-diff*)
have 1: $\exp(\gamma * (t-1)^2 / (2*k)) * (k+l-t \text{ choose } l) \leq (k / (k+l))^t * (k+l \text{ choose } l)$
using *mult-right-mono* [*OF D2* [*OF t*], *of exp* ($\gamma * (t-1)^2 / (2*k)$) *l*] *t*
by (*simp add: γ -def exp-minus field-simps*)
have 2: $(k / (k+l)) \text{ powr } (- \text{ real } t) * \exp(\gamma * (t-1)^2 / (2*k)) * (k+l-t \text{ choose } l) \leq (k+l \text{ choose } l)$
using *mult-right-mono* [*OF 1*, *of* ($1-\gamma$) *powr* ($- \text{ real } t$)] *t*
by (*simp add: powr-minus γ -def powr-realpow mult-ac divide-simps*)
then have 3: $(1-\gamma) \text{ powr } (- \text{ real } t) * \exp(\gamma * (t-1)^2 / (2*k)) * (k-t+l \text{ choose } l) \leq (k+l \text{ choose } l)$
by (*simp add: kkl*)
show ?thesis
apply (*rule order-trans* [*OF - 3*])
using * *less-eq-real-def* **by** *fastforce*
qed

8.4 Lemma 9.3

definition *ok-fun-93g* $\equiv \lambda\gamma k. (\text{nat } \lceil k \text{ powr } (3/4) \rceil) * \log 2 k - (\text{ok-fun-71 } \gamma k + \text{ok-fun-94 } k) + 1$

lemma *ok-fun-93g*:

assumes $0 < \gamma$ $\gamma < 1$
shows *ok-fun-93g* $\gamma \in o(\text{real})$

proof –

have $(\lambda k. (\text{nat } \lceil k \text{ powr } (3/4) \rceil) * \log 2 k) \in o(\text{real})$
by *real-asymp*

then show ?thesis

unfolding *ok-fun-93g-def*

by (*intro ok-fun-71* [*OF assms*] *ok-fun-94 sum-in-smallo const-smallo-real*)

qed

definition *ok-fun-93h* $\equiv \lambda\gamma k. (2 / (1-\gamma)) * k \text{ powr } (19/20) * (\ln \gamma + 2 * \ln k) + \text{ok-fun-93g } \gamma k * \ln 2$

lemma *ok-fun-93h*:

assumes $0 < \gamma$ $\gamma < 1$
shows *ok-fun-93h* $\gamma \in o(\text{real})$

proof –

have $(\lambda k. (2 / (1-\gamma)) * k \text{ powr } (19/20) * (\ln \gamma + 2 * \ln k)) \in o(\text{real})$
by *real-asymp*

then show ?thesis

unfolding *ok-fun-93h-def* **by** (*metis* (*mono-tags*) *ok-fun-93g assms sum-in-smallo(1) cmult-in-smallo-iff'*)

qed

lemma *ok-fun-93h-uniform*:

assumes $\mu01$: $0 < \mu0$ $\mu1 < 1$

assumes $e > 0$

shows $\forall^\infty k. \forall \mu. \mu \in \{\mu0.. \mu1\} \longrightarrow |ok-fun-93h \ \mu \ k| / k \leq e$

proof –

define f where $f \equiv \lambda k. ok-fun-73 \ k + ok-fun-74 \ k + ok-fun-76 \ k + ok-fun-94 \ k$

define g where $g \equiv \lambda \mu \ k. 2 * real \ k \ powr \ (19/20) * (\ln \ \mu + 2 * \ln \ k) / (1 - \mu)$

have g : $\forall^\infty k. \forall \mu. \mu0 \leq \mu \wedge \mu \leq \mu1 \longrightarrow |g \ \mu \ k| / k \leq e$ if $e > 0$ for e

proof (intro eventually-all-geI1 [where $L = nat \lceil 1 / \mu0 \rceil$])

show $\forall^\infty k. |g \ \mu1 \ k| / real \ k \leq e$

using *assms that unfolding g-def by real-asymp*

next

fix $k \ \mu$

assume $le-e$: $|g \ \mu1 \ k| / k \leq e$ and μ : $\mu0 \leq \mu \leq \mu1$ and k : $nat \lceil 1/\mu0 \rceil \leq k$

then have $k > 0$

using *assms gr0I by force*

have $ln-k$: $\ln \ k \geq \ln \ (1/\mu0)$

using $k < 0 < \mu0 >$ *ln-mono by fastforce*

with $\mu \ \mu01$

have $|\ln \ \mu + 2 * \ln \ (real \ k)| \leq |\ln \ \mu1 + 2 * \ln \ (real \ k)|$

by (*smt (verit) ln-div ln-mono ln-one*)

with $\mu \ k < \mu1 < 1 >$

have $|g \ \mu \ k| \leq |g \ \mu1 \ k|$

by (*simp add: g-def abs-mult frac-le mult-mono*)

then show $|g \ \mu \ k| / real \ k \leq e$

by (*smt (verit, best) divide-right-mono le-e of-nat-less-0-iff*)

qed

have $eq93$: $ok-fun-93h \ \mu \ k = g \ \mu \ k +$

$\lceil k \ powr \ (3/4) \rceil * \ln \ k - ((ok-fun-72 \ \mu \ k + f \ k) - 1) * \ln \ 2$ for $\mu \ k$

by (*simp add: ok-fun-93h-def g-def ok-fun-71-def ok-fun-93g-def f-def log-def field-simps*)

have $ln2$: $\ln \ 2 \geq (0::real)$

by *simp*

have $le93$: $|ok-fun-93h \ \mu \ k|$

$\leq |g \ \mu \ k| + |\lceil k \ powr \ (3/4) \rceil * \ln \ k| + (|ok-fun-72 \ \mu \ k| + |f \ k| + 1) * \ln \ 2$

for $\mu \ k$

unfolding $eq93$

by (*smt (verit, best) mult.commute ln-gt-zero-iff mult-le-cancel-left-pos mult-minus-left*)

define $e5$ where $e5 \equiv e/5$

have $e5 > 0$

by (*simp add: <e>0> e5-def*)

then have A : $\forall^\infty k. \forall \mu. \mu \in \{\mu0.. \mu1\} \longrightarrow |g \ \mu \ k| / k \leq e5$

using g by *simp*

have B : $\forall^\infty k. |\lceil k \ powr \ (3/4) \rceil * \ln \ k| / k \leq e5$

using $<0 < e5 >$ by *real-asymp*

have C : $\forall^\infty k. \forall \mu. \mu \in \{\mu0.. \mu1\} \longrightarrow |ok-fun-72 \ \mu \ k| * \ln \ 2 / k \leq e5$

```

    using ln2 assms ok-fun-72-uniform[OF  $\mu 01$ , of  $e5 / \ln 2$ ]  $\langle e5 > 0 \rangle$ 
    by (simp add: divide-simps)
  have  $f \in o(\text{real})$ 
    by (simp add: f-def ok-fun-73 ok-fun-74 ok-fun-76 ok-fun-94 sum-in-smallo(1))
  then have  $D: \forall^\infty k. |f k| * \ln 2 / k \leq e5$ 
    using  $\langle e5 > 0 \rangle \ln 2$ 
    by (force simp: smallo-def field-simps eventually-at-top-dense dest!: spec [where
 $x=e5 / \ln 2$ ])
  have  $E: \forall^\infty k. \ln 2 / k \leq e5$ 
    using  $\langle e5 > 0 \rangle \ln 2$  by real-asymp
  have  $\forall^\infty k. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow |ok\text{-fun-93h } \mu k| / \text{real } k \leq e5 + e5 + e5 + e5 + e5$ 
    using  $A B C D E$ 
    apply eventually-elim
    by (fastforce simp: add-divide-distrib distrib-right
        intro!: order-trans [OF divide-right-mono [OF le93]])
  then show ?thesis
    by (simp add: e5-def)
qed

```

```

context  $P0\text{-min}$ 
begin

```

definition $Big\text{-Far-9-3} \equiv$

```

 $\lambda \mu l. Big\text{-ZZ-8-5 } \mu l \wedge Big\text{-X-7-1 } \mu l \wedge Big\text{-Y-6-2 } \mu l \wedge Big\text{-Red-5-3 } \mu l$ 
 $\wedge (\forall k \geq l. p0\text{-min} - 3 * eps k > 1/k \wedge k \geq 2$ 
 $\wedge |ok\text{-fun-93h } \mu k| / (\mu * (1 + 1 / (exp 1 * (1-\mu)))) / k \leq 0.667 -$ 
 $2/3)$ 

```

lemma $Big\text{-Far-9-3}$:

```

assumes  $0 < \mu 0$   $\mu 0 \leq \mu 1$   $\mu 1 < 1$ 
shows  $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow Big\text{-Far-9-3 } \mu l$ 

```

proof –

```

define  $d$  where  $d \equiv \lambda \mu::\text{real}. \mu * (1 + 1 / (exp 1 * (1-\mu)))$ 

```

```

have  $d \mu 0 > 0$ 

```

```

using assms by (auto simp: d-def divide-simps add-pos-pos)

```

```

then have  $dgt: d \mu \geq d \mu 0$  if  $\mu \in \{\mu 0.. \mu 1\}$  for  $\mu$ 

```

```

using that assms by (auto simp: d-def frac-le mult-mono)

```

```

define  $e::\text{real}$  where  $e \equiv 0.667 - 2/3$ 

```

```

have  $e > 0$ 

```

```

by (simp add: e-def)

```

```

have  $*$ :  $\forall^\infty l. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow (\forall k \geq l. |ok\text{-fun-93h } \mu k| / d \mu) / k \leq e$ 

```

proof –

```

have  $\forall^\infty l. \forall k \geq l. (\forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow |ok\text{-fun-93h } \mu k| / k \leq d \mu 0 * e)$ 

```

```

using mult-pos-pos[OF  $\langle d \mu 0 > 0 \rangle \langle e > 0 \rangle$ ] assms

```

```

using ok-fun-93h-uniform eventually-all-ge-at-top

```

```

by blast

```

```

then show ?thesis

```

```

apply eventually-elim

```

```

    using dgt <0 < d μ0> <0 < e>
    by (auto simp: mult-ac divide-simps mult-less-0-iff zero-less-mult-iff split:
if-split-asm)
      (smt (verit) mult-less-cancel-left nat-neq-iff of-nat-0-le-iff)
  qed
  with p0-min show ?thesis
  unfolding Big-Far-9-3-def eps-def d-def e-def
  using assms Big-ZZ-8-5 Big-X-7-1 Big-Y-6-2 Big-Red-5-3
  apply (simp add: eventually-conj-iff all-imp-conj-distrib)
  apply (intro conjI strip eventually-all-ge-at-top; real-asymp)
  done
qed

end

```

```

lemma (λk. (nat ⌈real k powr (3/4)⌉) * log 2 k) ∈ o(real)
  by real-asymp

```

```

lemma RN34-le-2powr-ok:
  fixes l k::nat
  assumes l ≤ k 0 < k
  defines l34 ≡ nat ⌈real l powr (3/4)⌉
  shows RN k l34 ≤ 2 powr (⌈k powr (3/4)⌉ * log 2 k)
proof -
  have §: ⌈l powr (3/4)⌉ ≤ ⌈k powr (3/4)⌉
    by (simp add: assms(1) ceiling-mono powr-mono2)
  have RN k l34 ≤ k powr (l34 - 1)
    — Bhavik’s off-diagonal Ramsey upper bound; can’t use (2::'a)k + l34
    using RN-le-argpower' <k>0> powr-realpow by auto
  also have ... ≤ k powr l34
    using <k>0> powr-mono by force
  also have ... ≤ 2 powr (l34 * log 2 k)
    by (smt (verit, best) mult.commute <k>0> of-nat-0-less-iff powr-log-cancel
powr-powr)
  also have ... ≤ 2 powr (⌈real k powr (3/4)⌉ * log 2 k)
    unfolding l34-def
  proof (intro powr-mono powr-mono2 mult-mono ceiling-mono of-nat-mono nat-mono
<l ≤ k>)
    show 0 ≤ real-of-int ⌈k powr (3/4)⌉
      by (meson le-of-int-ceiling order.trans powr-ge-zero)
    qed (use assms § in auto)
  finally show ?thesis .
qed

```

Here n really refers to the cardinality of V , so actually nV

```

lemma (in Book') Far-9-3:
  defines δ ≡ min (1/200) (γ/20)
  defines ℛ ≡ Step-class {red-step}
  defines t ≡ card ℛ

```

assumes $\gamma15$: $\gamma \leq 1/5$ **and** $p0$: $p0 \geq 1/4$
and nge : $n \geq \exp(-\delta * \text{real } k) * (k+l \text{ choose } l)$
and $X0ge$: $\text{card } X0 \geq n/2$
— Because $n / 2 \leq \text{real } (\text{card } X0)$ makes the proof harder
assumes big : $Big\text{-Far-9-3 } \gamma \ l$
shows $t \geq 2*k / 3$
proof –
define \mathcal{S} **where** $\mathcal{S} \equiv \text{Step-class } \{dboost\text{-step}\}$
have $k \geq 2$ **and** $big85$: $Big\text{-ZZ-8-5 } \gamma \ l$ **and** $big71$: $Big\text{-X-7-1 } \gamma \ l$
and $big62$: $Big\text{-Y-6-2 } \gamma \ l$ **and** $big53$: $Big\text{-Red-5-3 } \gamma \ l$
using $big \ l\text{-le-}k$ **by** $(\text{auto simp: } Big\text{-Far-9-3-def})$
define $l34$ **where** $l34 \equiv \text{nat } \lceil \text{real } l \text{ powr } (3/4) \rceil$
have $l34 > 0$
using $l34\text{-def } ln0$ **by** fastforce
have $\gamma01$: $0 < \gamma \ \gamma < 1$
using $ln0 \ l\text{-le-}k$ **by** $(\text{auto simp: } \gamma\text{-def})$
then have $bigbeta01$: $0 < bigbeta \ bigbeta < 1$
using $big53 \ \text{assms } bigbeta\text{-gt}0 \ bigbeta\text{-less}1$ **by** $(\text{auto simp: } bigbeta\text{-def})$
have $one\text{-minus}$: $1-\gamma = \text{real } k / (\text{real } k + \text{real } l)$
using $ln0$ **by** $(\text{simp add: } \gamma\text{-def } \text{divide-simps})$
have $t < k$
using red-step-limit **by** $(\text{auto simp: } \mathcal{R}\text{-def } t\text{-def})$
have f : $2 \text{ powr } ok\text{-fun-94 } k * \gamma \ \text{powr } (-\text{real } l) * (1-\gamma) \ \text{powr } (-\text{real } k)$
 $\leq k+l \ \text{choose } l$
unfolding $\gamma\text{-def}$ **using** $\text{fact-9-4 } l\text{-le-}k \ ln0$ **by** blast
have $\text{powr-combine-right}$: $x \ \text{powr } a * (x \ \text{powr } b * y) = x \ \text{powr } (a+b) * y$ **for** x
 $y \ a \ b :: \text{real}$
by $(\text{simp add: } \text{powr-add})$
have $(2 \ \text{powr } ok\text{-fun-71 } \gamma \ k * 2 \ \text{powr } ok\text{-fun-94 } k) * (bigbeta/\gamma) \wedge \text{card } \mathcal{S} * (\exp$
 $(-\delta*k) * (1-\gamma) \ \text{powr } (-\text{real } k + t) / 2)$
 $\leq 2 \ \text{powr } ok\text{-fun-71 } \gamma \ k * \gamma^l * (1-\gamma) \wedge t * (bigbeta/\gamma) \wedge \text{card } \mathcal{S} * (\exp$
 $(-\delta*k) * (k+l \ \text{choose } l) / 2)$
using $\gamma01 \ \langle 0 < bigbeta \rangle \ \text{mult-right-mono}$ [$OF \ f$, of $2 \ \text{powr } ok\text{-fun-71 } \gamma \ k * \gamma^l$
 $* (1-\gamma) \wedge t * (bigbeta/\gamma) \wedge \text{card } \mathcal{S} * (\exp(-\delta*k)) / 2$]
by $(\text{simp add: } \text{mult-ac } \text{zero-le-mult-iff } \text{powr-minus } \text{powr-diff } \text{divide-simps } \text{powr-realpow})$
also have $\dots \leq 2 \ \text{powr } ok\text{-fun-71 } \gamma \ k * \gamma^l * (1-\gamma) \wedge t * (bigbeta/\gamma) \wedge \text{card}$
 $\mathcal{S} * \text{card } X0$
proof $(\text{intro } \text{mult-left-mono } \text{order-refl})$
show $\exp(-\delta * k) * \text{real } (k+l \ \text{choose } l) / 2 \leq \text{real } (\text{card } X0)$
using $X0ge \ nge$ **by** force
show $0 \leq 2 \ \text{powr } ok\text{-fun-71 } \gamma \ k * \gamma^l * (1-\gamma) \wedge t * (bigbeta/\gamma) \wedge \text{card } \mathcal{S}$
using $\gamma01 \ bigbeta\text{-ge}0$ **by** $(\text{force simp: } bigbeta\text{-def})$
qed
also have $\dots \leq \text{card } (Xseq \ \text{halted-point})$
unfolding $\mathcal{R}\text{-def } \mathcal{S}\text{-def } t\text{-def}$ **using** big
by $(\text{intro } X\text{-7-1}) \ (\text{auto simp: } Big\text{-Far-9-3-def})$
also have $\dots \leq RN \ k \ l34$
proof –
have $p0 - 3 * \varepsilon > 1/k$ **and** $pseq \ \text{halted-point} \geq p0 - 3 * \varepsilon$

```

    using l-le-k big p0-ge Y-6-2-halted by (auto simp: Big-Far-9-3-def  $\gamma$ -def)
  then show ?thesis
    using halted-point-halted  $\gamma$ 01
      by (fastforce simp: step-terminating-iff termination-condition-def pseq-def
l34-def)
  qed
  also have ...  $\leq 2 \text{ powr } (\lceil k \text{ powr } (3/4) \rceil * \log 2 k)$ 
    using RN34-le-2powr-ok l34-def l-le-k ln0 by blast
  finally have  $2 \text{ powr } (\text{ok-fun-71 } \gamma k + \text{ok-fun-94 } k) * (\text{bigbeta}/\gamma) ^ \text{card } \mathcal{S}$ 
    *  $\exp(-\delta*k) * (1-\gamma) \text{ powr } (- \text{real } k + t) / 2$ 
     $\leq 2 \text{ powr } (\lceil k \text{ powr } (3/4) \rceil * \log 2 k)$ 
    by (simp add: powr-add)
  then have le-2-powr-g:  $\exp(-\delta*k) * (1-\gamma) \text{ powr } (- \text{real } k + t) * (\text{bigbeta}/\gamma)$ 
 $^ \text{card } \mathcal{S}$ 
     $\leq 2 \text{ powr } \text{ok-fun-93g } \gamma k$ 
    using  $\langle k \geq 2 \rangle$  by (simp add: ok-fun-93g-def field-simps powr-add powr-diff flip:
powr-realpow)

  let ? $\xi$  =  $\text{bigbeta} * t / (1-\gamma) + (2 / (1-\gamma)) * k \text{ powr } (19/20)$ 
  have bigbeta-le:  $\text{bigbeta} \leq \gamma$  and bigbeta-ge:  $\text{bigbeta} \geq 1 / (\text{real } k)^2$ 
    using bigbeta-def  $\gamma$ 01 big53 bigbeta-le bigbeta-ge-square by blast+

  define  $\varphi$  where  $\varphi \equiv \lambda u. (u / (1-\gamma)) * \ln(\gamma/u)$  — finding the maximum via
derivatives
  have ln-eq:  $\ln(\gamma / (\gamma / \exp 1)) / (1-\gamma) = 1/(1-\gamma)$ 
    using  $\gamma$ 01 by simp
  have  $\varphi$ :  $\varphi(\gamma / \exp 1) \geq \varphi \text{ bigbeta}$ 
  proof (cases  $\gamma / \exp 1 \leq \text{bigbeta}$ ) — Could perhaps avoid case analysis via
2nd derivatives
    case True
      show ?thesis
      proof (intro DERIV-nonpos-imp-nonincreasing [where  $f = \varphi$ ])
        fix  $x$ 
        assume  $x: \gamma / \exp 1 \leq x \leq \text{bigbeta}$ 
        with  $\gamma$ 01 have  $x > 0$ 
          by (smt (verit, best) divide-pos-pos exp-gt-zero)
        with  $\gamma$ 01  $x$  have  $\ln(\gamma/x) / (1-\gamma) - 1 / (1-\gamma) \leq 0$ 
          by (smt (verit, ccfv-SIG) divide-pos-pos exp-gt-zero frac-le ln-eq ln-mono)
        with  $x \langle x > 0 \rangle$   $\gamma$ 01 show  $\exists D. (\varphi \text{ has-real-derivative } D) (at x) \wedge D \leq 0$ 
          unfolding  $\varphi$ -def by (intro exI conjI derivative-eq-intros | force)+
      qed (simp add: True)
    next
      case False
      show ?thesis
      proof (intro DERIV-nonneg-imp-nondecreasing [where  $f = \varphi$ ])
        fix  $x$ 
        assume  $x: \text{bigbeta} \leq x \leq \gamma / \exp 1$ 
        with bigbeta01  $\gamma$ 01 have  $x > 0$  by linarith
        with  $\gamma$ 01  $x$  have  $\ln(\gamma/x) / (1-\gamma) - 1 / (1-\gamma) \geq 0$ 

```

```

    by (smt (verit, best) frac-le ln-eq ln-mono zero-less-divide-iff)
  with  $x <x>0$   $\gamma 01$  show  $\exists D. (\varphi \text{ has-real-derivative } D) (at\ x) \wedge D \geq 0$ 
    unfolding  $\varphi$ -def
    by (intro exI conjI derivative-eq-intros | force)+
  qed (use False in force)
qed

define c where  $c \equiv \lambda x::real. 1 + 1 / (exp\ 1 * (1-x))$ 
have mono-c: mono-on  $\{0 <..<1\}$  c
  by (auto simp: monotone-on-def c-def field-simps)
have cgt0:  $c\ x > 0$  if  $x <1$  for x
  using that by (simp add: add-pos-nonneg c-def)

have card S  $\leq bigbeta * t / (1-bigbeta) + (2 / (1-\gamma)) * k\ powr\ (19/20)$ 
  using ZZ-8-5 [OF big85] by (auto simp: R-def S-def t-def)
also have ...  $\leq ?\xi$ 
  using bigbeta-le by (simp add:  $\gamma 01$  bigbeta-ge0 frac-le)
finally have card S  $\leq ?\xi$  .
with bigbeta-le bigbeta01 have  $?\xi * \ln (bigbeta/\gamma) \leq card\ S * \ln (bigbeta/\gamma)$ 
  by (simp add: mult-right-mono-neg)
then have  $-?\xi * \ln (\gamma/bigbeta) \leq card\ S * \ln (bigbeta/\gamma)$ 
  using bigbeta01  $\gamma 01$  by (smt (verit) ln-div minus-mult-minus)
then have  $\gamma * (real\ k - t) - \delta*k - ?\xi * \ln (\gamma/bigbeta) \leq \gamma * (real\ k - t) -$ 
 $\delta*k + card\ S * \ln (bigbeta/\gamma)$ 
  by linarith
also have ...  $\leq (t - real\ k) * \ln (1-\gamma) - \delta*k + card\ S * \ln (bigbeta/\gamma)$ 
  using  $\langle t < k \rangle \gamma 01$  mult-right-mono [OF ln-add-one-self-le-self2 [of  $-\gamma$ ], of real
k - t]
  by (simp add: algebra-simps)
also have ...  $= \ln (exp (-\delta*k) * (1-\gamma) powr (- real\ k + t) * (bigbeta/\gamma) ^$ 
card S)
  using  $\gamma 01$  bigbeta01 by (simp add: ln-mult ln-div ln-realpow)
also have ...  $\leq \ln (2\ powr\ ok-fun-93g\ \gamma\ k)$ 
  using le-2-powr-g  $\gamma 01$  bigbeta01 by (simp del: ln-powr)
also have ...  $= ok-fun-93g\ \gamma\ k * \ln\ 2$ 
  by auto
finally have  $\gamma * (real\ k - t) - \delta*k - ?\xi * \ln (\gamma/bigbeta) \leq ok-fun-93g\ \gamma\ k *$ 
 $\ln\ 2$  .
then have  $\gamma * (real\ k - t) \leq ?\xi * \ln (\gamma/bigbeta) + \delta*k + ok-fun-93g\ \gamma\ k * \ln\ 2$ 
  by simp
also have ...  $\leq (bigbeta * t / (1-\gamma)) * \ln (\gamma/bigbeta) + \delta*k + ok-fun-93h\ \gamma\ k$ 
proof -
  have  $\gamma/bigbeta \leq \gamma * (real\ k)^2$ 
    using kn0 bigbeta-le bigbeta-ge  $\langle bigbeta > 0 \rangle$  by (simp add: field-simps)
  then have X:  $\ln (\gamma/bigbeta) \leq \ln\ \gamma + 2 * \ln\ k$ 
    using  $\langle bigbeta > 0 \rangle \langle \gamma > 0 \rangle$  kn0
    by (metis ln-mult-pos ln-realpow of-nat-numeral of-nat-zero-less-power-iff
divide-pos-pos ln-mono)
  show ?thesis

```

```

    using mult-right-mono [OF X, of 2 * k pour (19/20) / (1-γ)] ⟨γ<1⟩
    by (simp add: ok-fun-93h-def algebra-simps)
qed
also have ... ≤ ((γ / exp 1) * t / (1-γ)) + δ*k + ok-fun-93h γ k
    using γ01 mult-right-mono [OF φ, of t] by (simp add: φ-def mult-ac)
finally have γ * (real k - t) ≤ ((γ / exp 1) * t / (1-γ)) + δ*k + ok-fun-93h
γ k .
then have (γ-δ) * k - ok-fun-93h γ k ≤ t * γ * c γ
    by (simp add: c-def algebra-simps)
then have ((γ-δ) * k - ok-fun-93h γ k) / (γ * c γ) ≤ t
    using γ01 cgt0 by (simp add: pos-divide-le-eq)
then have *: t ≥ (1-δ / γ) * inverse (c γ) * k - ok-fun-93h γ k / (γ * c γ)
    using γ01 cgt0[of γ] by (simp add: divide-simps)
define f47 where f47 ≡ λx. (1 - 1/(200*x)) * inverse (c x)
have concave-on {1/10..1/5} f47
    unfolding f47-def
proof (intro concave-on-mul)
    show concave-on {1/10..1/5} (λx. 1 - 1/(200*x))
    proof (intro f''-le0-imp-concave)
        fix x::real
        assume x ∈ {1/10..1/5}
        then have x01: 0 < x < 1 by auto
        show ((λx. (1 - 1/(200*x))) has-real-derivative 1/(200*x^2)) (at x)
            using x01 by (intro derivative-eq-intros | force simp: eval-nat-numeral)+
        show ((λx. 1/(200*x^2)) has-real-derivative -1/(100*x^3)) (at x)
            using x01 by (intro derivative-eq-intros | force simp: eval-nat-numeral)+
        show -1/(100*x^3) ≤ 0
            using x01 by (simp add: divide-simps)
    qed auto
    show concave-on {1/10..1/5} (λx. inverse (c x))
    proof (intro f''-le0-imp-concave)
        fix x::real
        assume x ∈ {1/10..1/5}
        then have x01: 0 < x < 1 by auto
        have swap: u * (x-1) = (-u) * (1-x) for u
            by (metis minus-diff-eq minus-mult-commute)
        have §: exp 1 * (x - 1) < 0
            using x01 by (meson exp-gt-zero less-iff-diff-less-0 mult-less-0-iff)
        then have non0: 1 + 1 / (exp 1 * (1-x)) ≠ 0
            using x01 by (smt (verit) exp-gt-zero mult-pos-pos zero-less-divide-iff)
        let ?f1 = λx. -exp 1 / (- 1 + exp 1 * (- 1 + x))^2
        let ?f2 = λx. 2*exp(1)^2/(-1 + exp(1)*(-1 + x))^3
        show ((λx. inverse (c x)) has-real-derivative ?f1 x) (at x)
            unfolding c-def power2-eq-square
            using x01 § non0
            apply (intro exI conjI derivative-eq-intros | force)+
            apply (simp add: divide-simps square-eq-iff swap)
            done
        show (?f1 has-real-derivative ?f2 x) (at x)

```

```

    using x01 §
    by (intro derivative-eq-intros | force simp: divide-simps eval-nat-numeral)+
    show ?f2 (x::real) ≤ 0
    using x01 § by (simp add: divide-simps)
  qed auto
  show mono-on {(1::real)/10..1/5} (λx. 1 - 1 / (200 * x))
    by (auto simp: monotone-on-def frac-le)
  show monotone-on {1/10..1/5} (≤) (λx y. y ≤ x) (λx. inverse (c x))
    using mono-c cgt0 by (auto simp: monotone-on-def divide-simps)
  qed (auto simp: c-def)
  moreover have f47(1/10) > 0.667
    unfolding f47-def c-def by (approximation 15)
  moreover have f47(1/5) > 0.667
    unfolding f47-def c-def by (approximation 15)
  ultimately have 47: f47 x > 0.667 if x ∈ {1/10..1/5} for x
    using concave-on-ge-min that by fastforce

define f48 where f48 ≡ λx. (1 - 1/20) * inverse (c x)
have 48: f48 x > 0.667 if x ∈ {0<..<1/10} for x
proof -
  have (0.667::real) < (1 - 1/20) * inverse(c(1/10))
    unfolding c-def by (approximation 15)
  also have ... ≤ f48 x
    using that unfolding f48-def c-def
  by (intro mult-mono le-imp-inverse-le add-mono divide-left-mono) (auto simp:
add-pos-pos)
  finally show ?thesis .
qed
define e::real where e ≡ 0.667 - 2/3
have BIGH: abs (ok-fun-93h γ k / (γ * c γ)) / k ≤ e
  using big l-le-k unfolding Big-Far-9-3-def all-imp-conj-distrib e-def [symmetric]
c-def
  by auto
consider γ ∈ {0<..<1/10} | γ ∈ {1/10..1/5}
  using δ-def ⟨γ ≤ 1/5⟩ γ01 by fastforce
then show ?thesis
proof cases
  case 1
  then have δγ: δ / γ = 1/20
    by (auto simp: δ-def)
  have (2/3::real) ≤ f48 γ - e
    using 48[OF 1] e-def by force
  also have ... ≤ (1-δ / γ) * inverse (c γ) - ok-fun-93h γ k / (γ * c γ) / k
    unfolding f48-def δγ using BIGH
  by (smt (verit, best) divide-nonneg-nonneg of-nat-0-le-iff zero-less-divide-iff)
  finally
  have A: 2/3 ≤ (1-δ / γ) * inverse (c γ) - ok-fun-93h γ k / (γ * c γ) / k .
  have real (2 * k) / 3 ≤ (1 - δ / γ) * inverse (c γ) * k - ok-fun-93h γ k /
(γ * c γ)

```

```

    using mult-left-mono [OF A, of k] cgt0 [of  $\gamma$ ]  $\gamma 01$  kn0
    by (simp add: divide-simps mult-ac)
  with * show ?thesis
    by linarith
next
case 2
then have  $\delta\gamma: \delta / \gamma = 1/(200*\gamma)$ 
  by (auto simp:  $\delta$ -def)
have  $(2/3::real) \leq f47 \gamma - e$ 
  using 47[OF 2] e-def by force
also have ...  $\leq (1 - \delta / \gamma) * inverse (c \gamma) - ok-fun-93h \gamma k / (\gamma * c \gamma) / k$ 
  unfolding f47-def  $\delta\gamma$  using BIGH
  by (smt (verit, best) divide-right-mono of-nat-0-le-iff)
finally
have  $2/3 \leq (1 - \delta / \gamma) * inverse (c \gamma) - ok-fun-93h \gamma k / (\gamma * c \gamma) / k$  .
from mult-left-mono [OF this, of k] cgt0 [of  $\gamma$ ]  $\gamma 01$  kn0
have  $real (2 * k) / 3 \leq (1 - \delta / \gamma) * inverse (c \gamma) * k - ok-fun-93h \gamma k /$ 
 $(\gamma * c \gamma)$ 
  by (simp add: divide-simps mult-ac)
with * show ?thesis
  by linarith
qed
qed

```

8.5 Lemma 9.5

context *P0-min*

begin

Again stolen from Bhavik: cannot allow a dependence on γ

definition *ok-fun-95a* $\equiv \lambda k. ok-fun-61 k - (2 + 4 * k \text{ powr } (19/20))$

definition *ok-fun-95b* $\equiv \lambda k. \ln 2 * ok-fun-95a k - 1$

lemma *ok-fun-95a*: *ok-fun-95a* $\in o(\text{real})$

proof –

have $(\lambda k. 2 + 4 * k \text{ powr } (19/20)) \in o(\text{real})$

by *real-asymp*

then show ?thesis

unfolding *ok-fun-95a-def* using *ok-fun-61 sum-in-smallo* by blast

qed

lemma *ok-fun-95b*: *ok-fun-95b* $\in o(\text{real})$

using *ok-fun-95a* by (auto simp: *ok-fun-95b-def sum-in-smallo const-smallo-real*)

definition *Big-Far-9-5* $\equiv \lambda \mu l. \text{Big-Red-5-3 } \mu l \wedge \text{Big-Y-6-1 } \mu l \wedge \text{Big-ZZ-8-5 } \mu l$

lemma *Big-Far-9-5*:

assumes $0 < \mu 0 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow \text{Big-Far-9-5 } \mu l$
using *assms* *Big-Red-5-3 Big-Y-6-1 Big-ZZ-8-5*
unfolding *Big-Far-9-5-def eps-def*
by (*simp add: eventually-conj-iff all-imp-conj-distrib*)

end

Y0 is an additional assumption found in Bhavik's version. (He had a couple of others). The first $o(k)$ function adjusts for the error in $n/2$

lemma (in *Book'*) *Far-9-5*:

fixes $\delta \eta :: \text{real}$
defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$
defines $t \equiv \text{card } \mathcal{R}$
assumes $nV: \text{real } nV \geq \exp(-\delta * k) * (k+l \text{ choose } l)$ **and** $Y0: \text{card } Y0 \geq nV$
div 2
assumes $p0: 1/2 \leq 1-\gamma-\eta$ $1-\gamma-\eta \leq p0$ **and** $0 \leq \eta$
assumes *big: Big-Far-9-5* γl
shows $\text{card } (Y\text{seq halted-point}) \geq$
 $\exp(-\delta * k + \text{ok-fun-95b } k) * (1-\gamma-\eta) \text{ powr } (\gamma * t / (1-\gamma)) * ((1-\gamma-\eta)/(1-\gamma)) \wedge t$
 $* \exp(\gamma * (\text{real } t)^2 / (2 * k)) * (k-t+l \text{ choose } l)$ (**is** $- \geq$ *?rhs*)

proof –

define \mathcal{S} **where** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$
define s **where** $s \equiv \text{card } \mathcal{S}$
have $\gamma 01: 0 < \gamma$ $\gamma < 1$
using *ln0 l-le-k* **by** (*auto simp:* $\gamma\text{-def}$)
have *big85: Big-ZZ-8-5* γl **and** *big61: Big-Y-6-1* γl **and** *big53: Big-Red-5-3* γl
using *big* **by** (*auto simp: Big-Far-9-5-def*)
have $\text{bigbeta} \leq \gamma$
using *bigbeta-def* $\gamma 01$ *big53 bigbeta-le* **by** *blast*
have $85: s \leq (\text{bigbeta} / (1-\text{bigbeta})) * t + (2 / (1-\gamma)) * k \text{ powr } (19/20)$
unfolding *s-def t-def* $\mathcal{R}\text{-def}$ $\mathcal{S}\text{-def}$ **using** *ZZ-8-5* $\gamma 01$ *big85* **by** *blast*
also have $\dots \leq (\gamma / (1-\gamma)) * t + (2 / (1-\gamma)) * k \text{ powr } (19/20)$
using $\gamma 01$ $\langle \text{bigbeta} \leq \gamma \rangle$ **by** (*intro add-mono mult-right-mono frac-le*) *auto*
finally have $D85: s \leq \gamma * t / (1-\gamma) + (2 / (1-\gamma)) * k \text{ powr } (19/20)$
by *auto*
have $t < k$
unfolding *t-def* $\mathcal{R}\text{-def}$ **using** $\gamma 01$ *red-step-limit* **by** *blast*
have $st: \text{card } (\text{Step-class } \{\text{red-step}, \text{dboost-step}\}) = t + s$
using $\gamma 01$
by (*simp add: s-def t-def* $\mathcal{R}\text{-def}$ $\mathcal{S}\text{-def}$ *Step-class-insert-NO-MATCH card-Un-disjnt disjnt-Step-class*)
then have $61: 2 \text{ powr } (\text{ok-fun-61 } k) * p0 \wedge (t+s) * \text{card } Y0 \leq \text{card } (Y\text{seq halted-point})$
using *Y-6-1[OF big61] card-XY0* $\gamma 01$ **by** (*simp add: divide-simps*)
have $(1-\gamma-\eta) \text{ powr } (t + \gamma * t / (1-\gamma)) * nV \leq (1-\gamma-\eta) \text{ powr } (t+s - 4 * k \text{ powr } (19/20)) * (4 * \text{card } Y0)$
proof (*intro mult-mono*)

show $(1-\gamma-\eta) \text{ powr } (t + \gamma * t / (1-\gamma)) \leq (1-\gamma-\eta) \text{ powr } (t+s - 4 * k \text{ powr } (19/20))$
proof (*intro powr-mono'*)
have $\gamma \leq 1/2$
using $\langle 0 \leq \eta \rangle p0$ **by** *linarith*
then have $22: 1 / (1 - \gamma) \leq 2$
using *divide-le-eq-1* **by** *fastforce*
show $\text{real } (t + s) - 4 * \text{real } k \text{ powr } (19 / 20) \leq \text{real } t + \gamma * \text{real } t / (1 - \gamma)$
using *mult-left-mono* [*OF 22, of 2 * real k powr (19 / 20)*] *D85*
by (*simp add: algebra-simps*)
next
show $0 \leq 1 - \gamma - \eta$ $1 - \gamma - \eta \leq 1$
using *assms* $\gamma 01$ **by** *linarith+*
qed
have $nV \geq 2$
by (*metis nontriv wellformed two-edges card-mono ex-in-conv finV*)
then have $nV \leq 4 * (nV \text{ div } 2)$ **by** *linarith*
also have $\dots \leq 4 * \text{card } Y0$
using *Y0 mult-le-mono2* **by** *presburger*
finally show $\text{real } nV \leq \text{real } (4 * \text{card } Y0)$
by force
qed (*use Y0 in auto*)
also have $\dots \leq (1-\gamma-\eta) \text{ powr } (t+s) / (1-\gamma-\eta) \text{ powr } (4 * k \text{ powr } (19/20))$
 $* (4 * \text{card } Y0)$
by (*simp add: divide-powr-uminus powr-diff*)
also have $\dots \leq (1-\gamma-\eta) \text{ powr } (t+s) / (1/2) \text{ powr } (4 * k \text{ powr } (19/20)) * (4 * \text{card } Y0)$
proof (*intro mult-mono divide-left-mono*)
show $(1/2) \text{ powr } (4 * k \text{ powr } (19/20)) \leq (1-\gamma-\eta) \text{ powr } (4 * k \text{ powr } (19/20))$
using $\gamma 01 p0 \langle 0 \leq \eta \rangle$ **by** (*intro powr-mono-both'*) *auto*
qed (*use p0 in auto*)
also have $\dots \leq p0 \text{ powr } (t+s) / (1/2) \text{ powr } (4 * k \text{ powr } (19/20)) * (4 * \text{card } Y0)$
using *p0 powr-mono2* **by** (*intro mult-mono divide-right-mono*) *auto*
also have $\dots = (2 \text{ powr } (2 + 4 * k \text{ powr } (19/20))) * p0 ^ (t+s) * \text{card } Y0$
using *p0-01* **by** (*simp add: powr-divide powr-add power-add powr-realpow*)
finally have $2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (t + \gamma * t / (1-\gamma)) * nV$
 $\leq 2 \text{ powr } (ok\text{-fun-61 } k) * p0 ^ (t+s) * \text{card } Y0$
by (*simp add: ok-fun-95a-def powr-diff field-simps*)
with *61* **have** $*: \text{card } (Y\text{seq halted-point}) \geq 2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (t + \gamma * t / (1-\gamma)) * nV$
by *linarith*

have $F: \text{exp } (ok\text{-fun-95b } k) = 2 \text{ powr } ok\text{-fun-95a } k * \text{exp } (-1)$
by (*simp add: ok-fun-95b-def exp-diff exp-minus powr-def field-simps*)
have *?rhs*
 $\leq \text{exp } (-\delta * k) * 2 \text{ powr } (ok\text{-fun-95a } k) * \text{exp } (-1) * (1-\gamma-\eta) \text{ powr } (\gamma * t / (1-\gamma))$

$$* (((1-\gamma-\eta)/(1-\gamma)) \wedge t * \exp (\gamma * (\text{real } t)^2 / \text{real}(2*k)) * (k-t+l \text{ choose } l))$$
unfolding *exp-add F by simp*
also have $\dots \leq \exp (-\delta * k) * 2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (\gamma*t / (1-\gamma))$

$$* (\exp (-1) * (((1-\gamma-\eta)/(1-\gamma)) \wedge t * \exp (\gamma * (\text{real } t)^2 / \text{real}(2*k)) * (k-t+l \text{ choose } l))$$
by (*simp add: mult.assoc*)
also have $\dots \leq 2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (t + \gamma*t / (1-\gamma)) * \exp (-\delta * k)$

$$* (\exp (-1) * (1-\gamma) \text{ powr } (- \text{real } t) * \exp (\gamma * (\text{real } t)^2 / \text{real}(2*k)) * (k-t+l \text{ choose } l))$$
using *p0 γ 01*
unfolding *powr-add powr-minus by (simp add: mult-ac divide-simps flip: powr-realpow)*
also have $\dots \leq 2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (t + \gamma*t / (1-\gamma)) * \exp (-\delta * k) * (k+l \text{ choose } l)$
proof (*cases t=0*)
case *False*
then show *?thesis*
unfolding $\gamma\text{-def}$ **using** $\langle t < k \rangle$ **by** (*intro mult-mono order-refl Far-9-6*) *auto*
qed *auto*
also have $\dots \leq 2 \text{ powr } (ok\text{-fun-95a } k) * (1-\gamma-\eta) \text{ powr } (t + \gamma*t / (1-\gamma)) * nV$
using *nV mult-left-mono by fastforce*
also have $\dots \leq \text{card } (Y\text{seq halted-point})$
by (*rule **)
finally show *?thesis .*
qed

8.6 Lemma 9.2

context *P0-min*
begin

lemma *error-9-2:*

assumes $\mu > 0 \ d > 0$

shows $\forall^\infty k. ok\text{-fun-95b } k + \mu * \text{real } k / d \geq 0$

proof $-$

have $\forall^\infty k. |ok\text{-fun-95b } k| \leq (\mu/d) * k$

using *ok-fun-95b assms* **unfolding** *smallo-def*

by (*auto dest!: spec [where x = μ/d]*)

then show *?thesis*

by *eventually-elim force*

qed

definition *Big-Far-9-2* $\equiv \lambda\mu \ l. \text{Big-Far-9-3 } \mu \ l \wedge \text{Big-Far-9-5 } \mu \ l \wedge (\forall k \geq l. ok\text{-fun-95b } k + \mu*k/60 \geq 0)$

```

lemma Big-Far-9-2:
  assumes  $0 < \mu_0$   $\mu_0 \leq \mu_1$   $\mu_1 < 1$ 
  shows  $\forall^\infty l. \forall \mu. \mu_0 \leq \mu \wedge \mu \leq \mu_1 \longrightarrow \text{Big-Far-9-2 } \mu \ l$ 
proof -
  have  $\forall^\infty l. \forall k \geq l. (\forall \mu. \mu_0 \leq \mu \wedge \mu \leq \mu_1 \longrightarrow 0 \leq \text{ok-fun-95b } k + \mu * k / 60)$ 
  using assms
  apply (intro eventually-all-ge-at-top eventually-all-geI0 error-9-2)
  apply (auto simp: divide-right-mono mult-right-mono elim!: order-trans)
  done
  then show ?thesis
  using assms Big-Far-9-3 Big-Far-9-5
  unfolding Big-Far-9-2-def
  apply (simp add: eventually-conj-iff all-imp-conj-distrib)
  by (smt (verit, ccfv-threshold) eventually-sequentially)
qed

end

```

Used for both 9.2 and 10.2

```

lemma (in Book') Off-diagonal-conclusion:
  defines  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ 
  defines  $t \equiv \text{card } \mathcal{R}$ 
  assumes  $Y: (k-t+l \text{ choose } l) \leq \text{card } (Y\text{seq halted-point})$ 
  shows False
proof -
  have  $t < k$ 
  unfolding t-def  $\mathcal{R}$ -def using red-step-limit by blast
  have  $RN \ (k-t) \ l \leq \text{card } (Y\text{seq halted-point})$ 
  by (metis Y add.commute RN-commute RN-le-choose le-trans)
  then obtain  $K$ 
  where  $K\text{sub}: K \subseteq Y\text{seq halted-point}$ 
  and  $K: \text{card } K = k-t \wedge \text{clique } K \text{ Red} \vee \text{card } K = l \wedge \text{clique } K \text{ Blue}$ 
  by (meson Red-Blue-RN Yseq-subset-V size-clique-def)
  show False
  using  $K$ 
proof
  assume  $K: \text{card } K = k-t \wedge \text{clique } K \text{ Red}$ 
  have  $\text{clique } (K \cup A\text{seq halted-point}) \text{ Red}$ 
  proof (intro clique-Un)
  show  $\text{clique } (A\text{seq halted-point}) \text{ Red}$ 
  by (meson A-Red-clique valid-state-seq)
  have  $\text{all-edges-betw-un } (A\text{seq halted-point}) \ (Y\text{seq halted-point}) \subseteq \text{Red}$ 
  using valid-state-seq Ksub
  by (auto simp: valid-state-def RB-state-def all-edges-betw-un-Un2)
  then show  $\text{all-edges-betw-un } K \ (A\text{seq halted-point}) \subseteq \text{Red}$ 
  using  $K\text{sub}$  all-edges-betw-un-commute all-edges-betw-un-mono2 by blast
  show  $K \subseteq V$ 
  using  $K\text{sub}$  Yseq-subset-V by blast
  qed (use  $K$  Aseq-subset-V in auto)

```

```

moreover have  $\text{card } (K \cup \text{Aseq halted-point}) = k$ 
proof –
  have  $\text{eqt: card } (\text{Aseq halted-point}) = t$ 
    using red-step-eq-Aseq R-def t-def by simp
  have  $\text{card } (K \cup \text{Aseq halted-point}) = \text{card } K + \text{card } (\text{Aseq halted-point})$ 
proof (intro card-Un-disjoint)
  show finite K
    by (meson Ksub Yseq-subset-V finV finite-subset)
  have disjnt (Yseq halted-point) (Aseq halted-point)
    using valid-state-seq by (auto simp: valid-state-def disjoint-state-def)
  with Ksub show  $K \cap \text{Aseq halted-point} = \{\}$ 
    by (auto simp: disjnt-def)
  qed (simp add: finite-Aseq)
  also have  $\dots = k$ 
    using eqt K <t <k> by simp
  finally show ?thesis .
qed
moreover have  $K \cup \text{Aseq halted-point} \subseteq V$ 
  using Aseq-subset-V Ksub Yseq-subset-V by blast
ultimately show False
  using no-Red-clique size-clique-def by blast
next
  assume  $\text{card } K = l \wedge \text{clique } K \text{ Blue}$ 
  then show False
    using Ksub Yseq-subset-V no-Blue-clique size-clique-def by blast
qed
qed

```

A little tricky to express since the Book locale assumes that there are no cliques in the original graph (page 9). So it's a contrapositive

lemma (in *Book*) *Far-9-2-aux*:

```

fixes  $\delta \eta :: \text{real}$ 
defines  $\delta \equiv \gamma / 20$ 
assumes  $0: \text{real } (\text{card } X0) \geq nV / 2 \text{ card } Y0 \geq nV \text{ div } 2 \text{ } p0 \geq 1 - \gamma - \eta$ 
  – These are the assumptions about the red density of the graph
assumes  $\gamma: \gamma \leq 1 / 10$  and  $\eta: 0 \leq \eta \leq \gamma / 15$ 
assumes  $nV: \text{real } nV \geq \exp (-\delta * k) * (k + l \text{ choose } l)$ 
assumes big: Big-Far-9-2  $\gamma \ l$ 
shows False
proof –
  define  $\mathcal{R}$  where  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ 
  define  $t$  where  $t \equiv \text{card } \mathcal{R}$ 
  have  $\gamma 01: 0 < \gamma \ \gamma < 1$ 
    using ln0 l-le-k by (auto simp: \gamma-def)
  have big93: Big-Far-9-3  $\gamma \ l$ 
    using big by (auto simp: Big-Far-9-2-def)
  have  $t23: t \geq 2 * k / 3$ 
    unfolding t-def R-def
  proof (rule Far-9-3)

```

show $\gamma \leq 1/5$
using γ **unfolding** γ -def **by** *linarith*
have $\min (1/200) (\gamma / 20) \geq \delta$
unfolding δ -def **using** γ *ln0* **by** (*simp add: γ -def*)
then show $\exp (-\min (1/200) (\gamma / 20) * k) * (k+l \text{ choose } l) \leq nV$
using δ -def γ -def *nV* **by force**
show $1/4 \leq p0$
using η γ *0* **by** *linarith*
show *Big-Far-9-3* (γ) *l*
using γ -def *big93* **by** *blast*
qed (*use assms in auto*)
have $t < k$
unfolding *t-def* \mathcal{R} -def **using** $\gamma01$ *red-step-limit* **by** *blast*

have *ge-half*: $1/2 \leq 1-\gamma-\eta$
using γ η **by** *linarith*
have $\exp (-1/3 + (1/5)::\text{real}) \leq \exp (10/9 * \ln (134/150))$
by (*approximation 9*)
also have $\dots \leq \exp (1 / (1-\gamma) * \ln (134/150))$
using γ **by** (*auto simp: divide-simps*)
also have $\dots \leq \exp (1 / (1-\gamma) * \ln (1-\gamma-\eta))$
using γ η **by** (*auto simp: divide-simps*)
also have $\dots = (1-\gamma-\eta) \text{ powr } (1 / (1-\gamma))$
using *ge-half* **by** (*simp add: powr-def*)
finally have $A: \exp (-1/3 + 1/5) \leq (1-\gamma-\eta) \text{ powr } (1 / (1-\gamma)) .$

have $3*t / (10*k) \leq (-1/3 + 1/5) + t/(2*k)$
using *t23 kn0* **by** (*simp add: divide-simps*)
from *mult-right-mono* [*OF this, of $\gamma*t$*] $\gamma01$
have $3*\gamma*t^2 / (10*k) \leq \gamma*t*(-1/3 + 1/5) + \gamma*t^2/(2*k)$
by (*simp add: eval-nat-numeral algebra-simps*)
then have $\exp (3*\gamma*t^2 / (10*k)) \leq \exp (-1/3 + 1/5) \text{ powr } (\gamma*t) * \exp (\gamma*t^2/(2*k))$
by (*simp add: mult-exp-exp exp-powr-real*)
also have $\dots \leq (1-\gamma-\eta) \text{ powr } ((\gamma*t) / (1-\gamma)) * \exp (\gamma*t^2/(2*k))$
using $\gamma01$ *powr-powr powr-mono2* [*of $\gamma*t \exp (-1/3 + 1/5)$, OF - - A*]
by (*intro mult-right-mono*) *auto*
finally have $B: \exp (3*\gamma*t^2 / (10*k)) \leq (1-\gamma-\eta) \text{ powr } ((\gamma*t) / (1-\gamma)) * \exp (\gamma*t^2/(2*k)) .$

have $(2*k / 3)^2 \leq t^2$
using *t23* **by** *auto*
from *kn0 $\gamma01$ mult-right-mono* [*OF this, of $\gamma/(80*k)$*]
have $C: \delta*k + \gamma*k/60 \leq 3*\gamma*t^2 / (20*k)$
by (*simp add: field-simps δ -def eval-nat-numeral*)

have $\exp (-3*\gamma*t / (20*k)) \leq \exp (-3 * \eta/2)$
proof -
have $1 \leq 3/2 * t/k$

```

    using t23 kn0 by (auto simp: divide-simps)
    from mult-right-mono [OF this, of  $\gamma/15$ ]  $\gamma 01 \eta$ 
    show ?thesis
      by simp
qed
also have ...  $\leq 1 - \eta / (1-\gamma)$ 
proof -
  have  $\S: 2/3 \leq (1 - \gamma - \eta)$ 
    using  $\gamma \eta$  by linarith
  have  $1 / (1-\eta / (1-\gamma)) = 1 + \eta / (1-\gamma-\eta)$ 
    using ge-half  $\eta$  by (simp add: divide-simps split: if-split-asm)
  also have ...  $\leq 1 + 3 * \eta / 2$ 
    using mult-right-mono [OF  $\S$ , of  $\eta$ ]  $\eta$  ge-half by (simp add: field-simps)
  also have ...  $\leq \exp (3 * \eta / 2)$ 
    using exp-minus-ge [of  $-3*\eta/2$ ] by simp
  finally show ?thesis
    using  $\gamma 01$  ge-half
    by (simp add: exp-minus divide-simps mult.commute split: if-split-asm)
qed
also have ...  $= (1-\gamma-\eta) / (1-\gamma)$ 
  using  $\gamma 01$  by (simp add: divide-simps)
finally have  $\exp (- 3*\gamma*t / (20*k)) \leq (1-\gamma-\eta) / (1-\gamma) .$ 
from powr-mono2 [of  $t$ , OF - - this] ge-half  $\gamma 01$ 
have  $D: \exp (- 3*\gamma*t^2 / (20*k)) \leq ((1-\gamma-\eta) / (1-\gamma)) ^t$ 
  by (simp add: eval-nat-numeral powr-powr exp-powr-real mult-ac flip: powr-realpow)

have  $Y: (k-t+l \text{ choose } l) \leq \text{card } (Y\text{seq halted-point})$ 
proof -
  have  $1 * \text{real}(k-t+l \text{ choose } l)$ 
     $\leq \exp (ok\text{-fun-95b } k + \gamma*k/60) * (k-t+l \text{ choose } l)$ 
    using big l-le-k unfolding Big-Far-9-2-def
    by (intro mult-right-mono mult-ge1-I) auto
  also have ...  $\leq \exp (3*\gamma*t^2 / (20*k) + -\delta * k + ok\text{-fun-95b } k) * (k-t+l$ 
    choose  $l)$ 
    using  $C$  by simp
  also have ...  $= \exp (3*\gamma*t^2 / (10*k)) * \exp (-\delta * k + ok\text{-fun-95b } k) * \exp$ 
     $(- 3*\gamma*t^2 / (20*k))$ 
     $* (k-t+l \text{ choose } l)$ 
    by (simp flip: exp-add)
  also have ...  $\leq \exp (3*\gamma*t^2 / (10*k)) * \exp (-\delta * k + ok\text{-fun-95b } k) *$ 
     $((1-\gamma-\eta)/(1-\gamma)) ^t$ 
     $* (k-t+l \text{ choose } l)$ 
    using  $\gamma 01$  ge-half  $D$  by (intro mult-right-mono) auto
  also have ...  $\leq (1-\gamma-\eta) \text{ powr } (\gamma*t / (1-\gamma)) * \exp (\gamma * t^2 / (2*k)) * \exp$ 
     $(-\delta * k + ok\text{-fun-95b } k)$ 
     $* ((1-\gamma-\eta)/(1-\gamma)) ^t * (k-t+l \text{ choose } l)$ 
    using  $\gamma 01$  ge-half by (intro mult-right-mono  $B$ ) auto
  also have ...  $= \exp (-\delta * k + ok\text{-fun-95b } k) * (1-\gamma-\eta) \text{ powr } (\gamma*t / (1-\gamma))$ 
     $* ((1-\gamma-\eta)/(1-\gamma)) ^t$ 

```

```

      * exp ( $\gamma * (\text{real } t)^2 / (2*k)$ ) * ( $k-t+l$  choose  $l$ )
    by (simp add: mult-ac)
  also have 95: ...  $\leq$  real (card (Yseq halted-point))
  unfolding t-def  $\mathcal{R}$ -def
  proof (rule Far-9-5)
    show  $1/2 \leq 1 - \gamma - \eta$ 
      using ge-half  $\gamma$ -def by blast
    show Big-Far-9-5 ( $\gamma$ )  $l$ 
      using Big-Far-9-2-def big unfolding  $\gamma$ -def by presburger
  qed (use assms in auto)
  finally show ?thesis by simp
qed
then show False
  using Off-diagonal-conclusion by (simp flip:  $\mathcal{R}$ -def t-def)
qed

```

Mediation of 9.2 (and 10.2) from locale *Book-Basis* to the book locales with the starting sets of equal size

lemma (in *No-Cliques*) *to-Book*:

```

  assumes gd:  $p0\text{-min} \leq \text{graph-density Red}$ 
  assumes  $\mu01$ :  $0 < \mu \mu < 1$ 
  obtains  $X0 Y0$  where  $l \geq 2$  card  $X0 \geq \text{real } nV / 2$  card  $Y0 = \text{gorder div } 2$ 
    and  $X0 = V \setminus Y0$   $Y0 \subseteq V$ 
    and  $\text{graph-density Red} \leq \text{gen-density Red } X0 Y0$ 
    and Book  $V E p0\text{-min Red Blue } l k \mu X0 Y0$ 
  proof -
    have  $\text{Red} \neq \{\}$ 
      using gd  $p0\text{-min}$  by (auto simp: graph-density-def)
    then have  $\text{gorder} \geq 2$ 
      by (metis Red-E card-mono equalsOI finV subset-empty two-edges wellformed)
    then have  $\text{div}2$ :  $0 < \text{gorder div } 2$   $\text{gorder div } 2 < \text{gorder}$ 
      by auto
    then obtain  $Y0$  where  $Y0$ : card  $Y0 = \text{gorder div } 2$   $Y0 \subseteq V$ 
      graph-density  $\text{Red} \leq \text{gen-density Red } (V \setminus Y0) Y0$ 
      by (metis complete Red-E exists-density-edge-density gen-density-commute)
    define  $X0$  where  $X0 \equiv V \setminus Y0$ 
    interpret Book  $V E p0\text{-min Red Blue } l k \mu X0 Y0$ 
  proof
    show  $X0 \subseteq V$  disjoint  $X0 Y0$ 
      by (auto simp:  $X0$ -def disjoint-iff)
    show  $p0\text{-min} \leq \text{gen-density Red } X0 Y0$ 
      using  $X0$ -def  $Y0$  gd gen-density-commute  $p0\text{-min}$  by auto
  qed (use assms  $\langle Y0 \subseteq V \rangle$  in auto)
  have False if  $l < 2$ 
    using that unfolding less-2-cases-iff
  proof
    assume  $l = \text{Suc } 0$ 
    with  $Y0$   $\text{div}2$  show False
      by (metis RN-1' no-Red-clique no-Blue-clique Red-Blue-RN Suc-leI kn0)
  qed

```

qed (use *ln0* in *auto*)
with *l-le-k* **have** $l \geq 2$
by *force*
have *card-X0*: $\text{card } X0 \geq nV/2$
using $Y0 \langle Y0 \subseteq V \rangle$ **unfolding** *X0-def*
by (*simp add: card-Diff-subset finite-Y0*)
then show *thesis*
using *Book-axioms X0-def Y0* $\langle 2 \leq l \rangle$ **that by** *blast*
qed

Material that needs to be proved **outside** the book locales

As above, for *Book'*

lemma (in *No-Cliques*) *to-Book'*:

assumes *gd*: $p0\text{-min} \leq \text{graph-density Red}$

assumes *l*: $0 < l \leq k$

obtains $X0 Y0$ **where** $l \geq 2$ $\text{card } X0 \geq \text{real } nV / 2$ $\text{card } Y0 = \text{gorder div } 2$ **and**
 $X0 = V \setminus Y0$ $Y0 \subseteq V$

and $\text{graph-density Red} \leq \text{gen-density Red } X0 Y0$

and *Book'* $V E p0\text{-min Red Blue } l k (\text{real } l / (\text{real } k + \text{real } l)) X0 Y0$

proof –

define γ **where** $\gamma \equiv \text{real } l / (\text{real } k + \text{real } l)$

have $0 < \gamma$ $\gamma < 1$

using *l* **by** (*auto simp: γ -def*)

with *assms to-Book* [of γ]

obtain $X0 Y0$ **where** $*$: $l \geq 2$ $\text{card } X0 \geq \text{real } nV / 2$ $\text{card } Y0 = \text{gorder div } 2$ $X0 = V \setminus Y0$ $Y0 \subseteq V$

$\text{graph-density Red} \leq \text{gen-density Red } X0 Y0$ *Book* $V E p0\text{-min Red Blue } l k \gamma X0 Y0$

by *blast*

then interpret *Book* $V E p0\text{-min Red Blue } l k \gamma X0 Y0$

by *blast*

have *Book'* $V E p0\text{-min Red Blue } l k \gamma X0 Y0$

using *Book'* γ -def **by** *auto*

with $*$ *assms* **show** *thesis*

using γ -def *that by* *blast*

qed

lemma (in *No-Cliques*) *Far-9-2*:

fixes $\delta \gamma \eta :: \text{real}$

defines $\gamma \equiv l / (\text{real } k + \text{real } l)$

defines $\delta \equiv \gamma / 20$

assumes *gd*: $\text{graph-density Red} \geq 1 - \gamma - \eta$ **and** *p0-min-OK*: $p0\text{-min} \leq 1 - \gamma - \eta$

assumes $\gamma \leq 1/10$ **and** η : $0 \leq \eta \leq \gamma/15$

assumes *nV*: $\text{real } nV \geq \exp(-\delta * k) * (k+l \text{ choose } l)$

assumes *big*: *Big-Far-9-2* γl

shows *False*

proof –

obtain $X0 Y0$ **where** $l \geq 2$ **and** *card-X0*: $\text{card } X0 \geq \text{real } nV / 2$

and *card-Y0*: $\text{card } Y0 = \text{gorder div } 2$

```

and X0-def:  $X0 = V \setminus Y0$  and  $Y0 \subseteq V$ 
and gd-le: graph-density  $Red \leq$  gen-density  $Red$   $X0$   $Y0$ 
and Book'  $V$   $E$  p0-min  $Red$   $Blue$   $l$   $k$   $\gamma$   $X0$   $Y0$ 
using to-Book' assms p0-min no-Red-clique no-Blue-clique ln0 by auto
then interpret Book'  $V$   $E$  p0-min  $Red$   $Blue$   $l$   $k$   $\gamma$   $X0$   $Y0$ 
by blast
show False
proof (intro Far-9-2-aux [of  $\eta$ ])
show  $1 - \gamma - \eta \leq p0$ 
using X0-def  $\gamma$ -def gd gd-le gen-density-commute p0-def by auto
qed (use assms card-X0 card-Y0 in auto)
qed

```

8.7 Theorem 9.1

An arithmetical lemma proved outside of the locales

lemma *kl-choose*:

```

fixes  $l$   $k::nat$ 
assumes  $m < l$   $k > 0$ 
defines  $PM \equiv \prod i < m. (l - real\ i) / (k + l - real\ i)$ 
shows  $(k + l\ choose\ l) = (k + l - m\ choose\ (l - m)) / PM$ 
proof -
have inj: inj-on  $(\lambda i. i - m)$   $\{m..<l\}$  — relating the power and binomials; maybe
easier using factorials
by (auto simp: inj-on-def)
have  $(\prod i < l. (k + l - i) / (l - i)) / (\prod i < m. (k + l - i) / (l - i))$ 
=  $(\prod i = m..<l. (k + l - i) / (l - i))$ 
using prod-divide-nat-ivl [of  $0$   $m$   $l$   $\lambda i. (k + l - i) / (l - i)$ ]  $\langle m < l \rangle$ 
by (simp add: atLeast0LessThan)
also have  $\dots = (\prod i < l - m. (k + l - m - i) / (l - m - i))$ 
apply (intro prod.reindex-cong [OF inj, symmetric])
by (auto simp: image-minus-const-atLeastLessThan-nat)
finally
have  $(\prod i < l - m. (k + l - m - i) / (l - m - i))$ 
=  $(\prod i < l. (k + l - i) / (l - i)) / (\prod i < m. (k + l - i) / (l - i))$ 
by linarith
also have  $\dots = (k + l\ choose\ l) * inverse (\prod i < m. (k + l - i) / (l - i))$ 
by (simp add: field-simps atLeast0LessThan binomial-altdef-of-nat)
also have  $\dots = (k + l\ choose\ l) * PM$ 
unfolding PM-def using  $\langle m < l \rangle$   $\langle k > 0 \rangle$ 
by (simp add: atLeast0LessThan flip: prod-inversef)
finally have  $(k + l - m\ choose\ (l - m)) = (k + l\ choose\ l) * PM$ 
by (simp add: atLeast0LessThan binomial-altdef-of-nat)
then show  $real(k + l\ choose\ l) = (k + l - m\ choose\ (l - m)) / PM$ 
by auto
qed

```

context *P0-min*

begin

The proof considers a smaller graph, so l needs to be so big that the smaller l' will be big enough.

definition *Big-Far-9-1* :: *real* \Rightarrow *nat* \Rightarrow *bool* **where**

Big-Far-9-1 $\equiv \lambda \mu l. l \geq 3 \wedge (\forall l'. \text{real } l' \geq (10/11) * \mu * \text{real } l \longrightarrow \mu^2 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-2 } \gamma l')$

The proof of theorem 10.1 requires a range of values

lemma *Big-Far-9-1*:

assumes $0 < \mu 0 \ \mu 0 \leq 1/10$

shows $\forall^\infty l. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq 1/10 \longrightarrow \text{Big-Far-9-1 } \mu l$

proof –

have $\mu 0^2 \leq 1/10$

using *assms* **by** (*smt (verit, ccfv-threshold) le-divide-eq-1 mult-left-le power2-eq-square*)

then have $\forall^\infty l. \forall \gamma. \mu 0^2 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-2 } \gamma l$

using *assms* **by** (*intro Big-Far-9-2*) *auto*

then obtain N **where** $N: \forall l \geq N. \forall \gamma. \mu 0^2 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-2 } \gamma$

l

using *eventually-sequentially* **by** *auto*

define M **where** $M \equiv \text{nat}[11*N / (10*\mu 0)]$

have $(10/11) * \mu 0 * l \geq N$ **if** $l \geq M$ **for** l

using *that* **by** (*simp add: M-def <μ0>0 mult-of-nat-commute pos-divide-le-eq*)

with N **have** $\forall l \geq M. \forall l'. \gamma. (10/11) * \mu 0 * l \leq l' \longrightarrow \mu 0^2 \leq \gamma \wedge \gamma \leq 1 / 10 \longrightarrow \text{Big-Far-9-2 } \gamma l'$

by (*smt (verit, ccfv-SIG) of-nat-le-iff*)

then have $\forall^\infty l. \forall l' \gamma. (10/11) * \mu 0 * l \leq l' \longrightarrow \mu 0^2 \leq \gamma \wedge \gamma \leq 1 / 10 \longrightarrow \text{Big-Far-9-2 } \gamma l'$

by (*auto simp: eventually-sequentially*)

moreover have $\forall^\infty l. l \geq 3$

by *simp*

ultimately show *?thesis*

unfolding *Big-Far-9-1-def*

apply *eventually-elim*

by (*smt (verit) <0<μ0> mult-left-mono mult-right-mono of-nat-less-0-iff power-mono zero-less-mult-iff*)

qed

The text claims the result for all k and l , not just those sufficiently large, but the $o(k)$ function allowed in the exponent provides a fudge factor

theorem *Far-9-1*:

fixes $l k :: \text{nat}$

fixes $\delta \gamma :: \text{real}$

defines $\gamma \equiv \text{real } l / (\text{real } k + \text{real } l)$

defines $\delta \equiv \gamma / 20$

assumes $\gamma: \gamma \leq 1/10$

assumes *big: Big-Far-9-1* γl

assumes *p0-min-91: p0-min* $\leq 1 - (1/10) * (1 + 1/15)$

shows $RN k l \leq \text{exp } (-\delta*k + 1) * (k+l \text{ choose } l)$

proof (*rule ccontr*)
assume *non*: $\neg \text{RN } k \ l \leq \text{exp } (-\delta * k + 1) * (k+l \text{ choose } l)$
with *RN-eq-0-iff* **have** $l > 0$ **by** *force*
with γ **have** *l9k*: $9 * l \leq k$
by (*auto simp: γ -def divide-simps*)
have $l \leq k$
using γ -*def* γ *nat-le-real-less* **by** *fastforce*
with $\langle l > 0 \rangle$ **have** $k > 0$ **by** *linarith*
define $\xi :: \text{real}$ **where** $\xi \equiv 1/15$
define *U-lower-bound-ratio* **where** — Bhavik's name
U-lower-bound-ratio $\equiv \lambda m. (1 + \xi)^m * (\prod_{i < m. (l - \text{real } i) / (k+l - \text{real } i)})$

define n **where** $n \equiv \text{RN } k \ l - 1$
have $l \geq 3$
using *big* **by** (*auto simp: Big-Far-9-1-def*)
have $k \geq 27$
using *l9k* $\langle l \geq 3 \rangle$ **by** *linarith*
have $\text{exp } 1 / (\text{exp } 1 - 2) < (27 :: \text{real})$
by (*approximation 5*)
also have *RN27*: $\dots \leq \text{RN } k \ l$
by (*meson RN-3plus' $\langle l \geq 3 \rangle \langle k \geq 27 \rangle$ le-trans numeral-le-real-of-nat-iff*)
finally have $\text{exp } 1 / (\text{exp } 1 - 2) < \text{RN } k \ l$.
moreover have $n < \text{RN } k \ l$
using *RN27* **by** (*simp add: n-def*)
moreover have $2 < \text{exp } (1 :: \text{real})$
by (*approximation 5*)
ultimately have *nRNe*: $n/2 > \text{RN } k \ l / \text{exp } 1$
by (*simp add: n-def field-split-simps*)

have $(k+l \text{ choose } l) / \text{exp } (-1 + \delta * k) < \text{RN } k \ l$
by (*smt (verit) divide-inverse exp-minus mult-minus-left mult-of-nat-commute non*)

then have $(\text{RN } k \ l / \text{exp } 1) * \text{exp } (\delta * k) > (k+l \text{ choose } l)$
unfolding *exp-add exp-minus* **by** (*simp add: field-simps*)
with *nRNe* **have** *n2exp-gt*: $(n/2) * \text{exp } (\delta * k) > (k+l \text{ choose } l)$
by (*smt (verit, best) exp-gt-zero mult-le-cancel-right-pos*)
then have *nexp-gt*: $n * \text{exp } (\delta * k) > (k+l \text{ choose } l)$
by *simp*

define V **where** $V \equiv \{.. < n\}$
define E **where** $E \equiv \text{all-edges } V$
interpret *Book-Basis* $V \ E$
proof qed (*auto simp: V-def E-def comp-sgraph.wellformed comp-sgraph.two-edges*)
have [*simp*]: $nV = n$
by (*simp add: V-def*)
then obtain *Red Blue*
where *Red-E*: $\text{Red} \subseteq E$ **and** *Blue-def*: $\text{Blue} = E - \text{Red}$
and *no-Red-K*: $\neg (\exists K. \text{size-clique } k \ K \ \text{Red})$
and *no-Blue-K*: $\neg (\exists K. \text{size-clique } l \ K \ \text{Blue})$

by (*metis* $\langle n < RN \ k \ l \rangle$ *less-RN-Red-Blue*)
have *Blue-E*: $Blue \subseteq E$ **and** *disjnt-Red-Blue*: *disjnt Red Blue*
and *Blue-eq*: $Blue = all\text{-}edges \ V - Red$
using *complete* **by** (*auto simp*: *Blue-def disjnt-iff E-def*)
define *is-good-clique* **where**
 $is\text{-}good\text{-}clique \equiv \lambda i \ K. \ clique \ K \ Blue \wedge K \subseteq V \wedge$
 $\quad \quad \quad card \ (V \cap (\bigcap_{w \in K}. \ Neighbours \ Blue \ w))$
 $\quad \quad \quad \geq \ real \ i * U\text{-lower-bound-ratio} \ (card \ K) - card \ K$
have *is-good-card*: $card \ K < l$ **if** *is-good-clique* $i \ K$ **for** $i \ K$
using *no-Blue-K* **that** **unfolding** *is-good-clique-def*
by (*metis nat-neq-iff size-clique-def size-clique-smaller*)
define *GC* **where** $GC \equiv \{C. \ is\text{-}good\text{-}clique \ n \ C\}$
have $GC \neq \{\}$
by (*auto simp*: *GC-def is-good-clique-def U-lower-bound-ratio-def E-def V-def*)
have $GC \subseteq Pow \ V$
by (*auto simp*: *is-good-clique-def GC-def*)
then have *finite GC*
by (*simp add*: *finV finite-subset*)
then obtain W **where** $W \in GC$ **and** $MaxW$: $Max \ (card \ 'GC) = card \ W$
using $\langle GC \neq \{\} \rangle$ *obtains-MAX* **by** *blast*
then have *49*: *is-good-clique* $n \ W$
using *GC-def* **by** *blast*
have *max49*: $\neg \ is\text{-}good\text{-}clique \ n \ (insert \ x \ W)$ **if** $x \in V \setminus W$ **for** x
proof
assume x : *is-good-clique* $n \ (insert \ x \ W)$
then have $card \ (insert \ x \ W) = Suc \ (card \ W)$
using *finV is-good-clique-def finite-subset* **that** **by** *fastforce*
with $x \ \langle finite \ GC \rangle$ **have** $Max \ (card \ 'GC) \geq Suc \ (card \ W)$
by (*simp add*: *GC-def rev-image-eqI*)
then show *False*
by (*simp add*: *MaxW*)
qed

have $W \subseteq V$
using *49* **by** (*auto simp*: *is-good-clique-def*)
define m **where** $m \equiv card \ W$
define γ' **where** $\gamma' \equiv (l - real \ m) / (k+l-real \ m)$
define η **where** $\eta \equiv \xi * \gamma'$

have *Red-Blue-RN*: $\exists K \subseteq X. \ size\text{-}clique \ m \ K \ Red \vee size\text{-}clique \ n \ K \ Blue$
if $card \ X \geq RN \ m \ n \ X \subseteq V$ **for** $m \ n$ **and** X
using *partn-lst-imp-is-clique-RN* [*OF is-Ramsey-number-RN* [*of m n*]] *finV* **that**

unfolding *is-clique-RN-def size-clique-def clique-indep-def Blue-eq*
by (*metis clique-iff-indep finite-subset subset-trans*)
define U **where** $U \equiv V \cap (\bigcap_{w \in W}. \ Neighbours \ Blue \ w)$
define EU **where** $EU \equiv E \cap Pow \ U$
define $RedU$ **where** $RedU \equiv Red \cap Pow \ U$
define $BlueU$ **where** $BlueU \equiv Blue \cap Pow \ U$

```

have RN k l > 0
  using <n < RN k l> by auto
have  $\gamma' > 0$ 
  using is-good-card [OF 49] by (simp add:  $\gamma'$ -def m-def)
then have  $\eta > 0$ 
  by (simp add:  $\eta$ -def  $\xi$ -def)
have finite W
  using <W  $\subseteq$  V> finV finite-subset by (auto simp: V-def)
have U  $\subseteq$  V and VUU: V  $\cap$  U = U
  by (force simp: U-def)+
have disjoint U W
  using Blue-E not-own-Neighbour unfolding E-def V-def U-def disjoint-iff by
blast
have m < l
  using 49 is-good-card m-def by blast
then have  $\gamma_{1516}$ :  $\gamma' \leq 15/16$ 
  using  $\gamma$ -def  $\gamma$  by (simp add:  $\gamma'$ -def divide-simps)
then have  $\gamma'$ -le1:  $(1+\xi) * \gamma' \leq 1$ 
  by (simp add:  $\xi$ -def)

have cardU:  $n * U$ -lower-bound-ratio  $m \leq m + \text{card } U$ 
  using 49 VUU unfolding is-good-clique-def U-def m-def by force
obtain [iff]: finite RedU finite BlueU RedU  $\subseteq$  EU
  using BlueU-def EU-def RedU-def E-def V-def Red-E Blue-E fin-edges finite-subset
by blast
have card-RedU-le:  $\text{card } \text{RedU} \leq \text{card } \text{EU}$ 
  by (metis EU-def E-def <RedU  $\subseteq$  EU> card-mono fin-all-edges finite-Int)
interpret UBB: Book-Basis U E  $\cap$  Pow U p0-min
proof
  fix e
  assume e  $\in$  E  $\cap$  Pow U
  with two-edges show e  $\subseteq$  U  $\text{card } e = 2$  by auto
next
show finite U
  using <U  $\subseteq$  V> by (simp add: V-def finite-subset)
have x  $\in$  E if x  $\in$  all-edges U for x
  using <U  $\subseteq$  V> all-edges-mono that complete E-def by blast
then show E  $\cap$  Pow U = all-edges U
  using comp-sgraph.wellformed <U  $\subseteq$  V> by (auto intro: e-in-all-edges-ss)
qed auto

have clique-W: size-clique m W Blue
  using 49 is-good-clique-def size-clique-def V-def m-def by blast

define PM where PM  $\equiv \prod_{i < m}. (l - \text{real } i) / (k+l - \text{real } i)$ 
then have U-lower-m: U-lower-bound-ratio m =  $(1+\xi)^m * PM$ 
  using U-lower-bound-ratio-def by blast
have prod-gt0: PM > 0

```

unfolding *PM-def* **using** $\langle m < l \rangle$ **by** (*intro prod-pos*) *auto*

have *kl-choose*: $\text{real}(k+l \text{ choose } l) = (k+l-m \text{ choose } (l-m)) / \text{PM}$
unfolding *PM-def* **using** *kl-choose* $\langle 0 < k \rangle \langle m < l \rangle$ **by** *blast*

— Now a huge effort just to show that U is nontrivial. Proof probably shows its cardinality exceeds a multiple of l

define *ekl20* **where** $\text{ekl20} \equiv \exp(k / (20*(k+l)))$
have *ekl20-eq*: $\exp(\delta*k) = \text{ekl20}^l$
by (*simp add: δ -def γ -def ekl20-def field-simps flip: exp-of-nat2-mult*)
have $\text{ekl20} \leq \exp(1/20)$
unfolding *ekl20-def* **using** $\langle m < l \rangle$ **by** *fastforce*
also have $\dots \leq (1+\xi)$
unfolding ξ -*def* **by** (*approximation 10*)
finally have *exp120*: $\text{ekl20} \leq 1 + \xi$.
have *ekl20-gt0*: $0 < \text{ekl20}$
by (*simp add: ekl20-def*)

have $3*l + \text{Suc } l - q \leq (k+q \text{ choose } q) / \exp(\delta*k) * (1+\xi) ^ (l - q)$
if $1 \leq q \leq l$ **for** q
using *that*

proof (*induction q rule: nat-induct-at-least*)
case *base*
have $\text{ekl20}^l = \text{ekl20}^{(l-1)} * \text{ekl20}$
by (*metis $\langle 0 < l \rangle$ power-minus-mult*)
also have $\dots \leq (1+\xi) ^ (l-1) * \text{ekl20}$
using *ekl20-def exp120 power-mono* **by** *fastforce*
also have $\dots \leq 2 * (1+\xi) ^ (l-1)$
proof –
have \S : $\text{ekl20} \leq 2$
using ξ -*def exp120* **by** *linarith*
from *mult-right-mono* [*OF this, of* $(1+\xi) ^ (l-1)$]
show *?thesis* **by** (*simp add: mult-ac ξ -def*)
qed
finally have $\text{ekl20}^l \leq 2 * (1+\xi) ^ (l-1)$
by *argo*
then have $1/2 \leq (1+\xi) ^ (l-1) / \text{ekl20}^l$
using *ekl20-def* **by** *auto*
moreover have $4 * \text{real } l / (1 + \text{real } k) \leq 1/2$
using *l9k* **by** (*simp add: divide-simps*)
ultimately have $4 * \text{real } l / (1 + \text{real } k) \leq (1+\xi) ^ (l-1) / \text{ekl20}^l$
by *linarith*
then show *?case*
by (*simp add: field-simps ekl20-eq*)

next
case (*Suc q*)
then have \ddagger : $(1+\xi) ^ (l - q) = (1+\xi) * (1+\xi) ^ (l - \text{Suc } q)$
by (*metis Suc-diff-le diff-Suc-Suc power.simps(2)*)
have $\text{real}(k + q \text{ choose } q) \leq \text{real}(k + q \text{ choose } \text{Suc } q) 0 \leq (1+\xi) ^ (l - \text{Suc } q)$

using $\langle \text{Suc } q \leq l \rangle$ $l9k$ **by** (*auto simp: ξ -def binomial-mono*)
from *mult-right-mono [OF this]*
have $(k + q \text{ choose } q) * (1 + \xi) ^ (l - q) / \text{exp } (\delta * k) - 1$
 $\leq (\text{real } (k + q \text{ choose } q) + (k + q \text{ choose } \text{Suc } q)) * (1 + \xi) ^ (l - \text{Suc } q) /$
 $\text{exp } (\delta * k)$
unfolding \ddagger **by** (*simp add: ξ -def field-simps add-increasing*)
with *Suc show ?case by force*
qed
from $\langle m < l \rangle$ *this [of $l - m$]*
have $1 + 3 * l + \text{real } m \leq (k + l - m \text{ choose } (l - m)) / \text{exp } \delta ^ k * (1 + \xi) ^ m$
by (*simp add: Suc-leI exp-of-nat2-mult*)
also have $\dots \leq (k + l - m \text{ choose } (l - m)) / \text{exp } (\delta * k) * (1 + \xi) ^ m$
by (*simp add: exp-of-nat2-mult*)
also have $\dots < PM * (\text{real } n * (1 + \xi) ^ m)$
proof –
have $\S: (k + l \text{ choose } l) / \text{exp } (\delta * k) < n$
by (*simp add: less-eq-real-def nexp-gt pos-divide-less-eq*)
show *?thesis*
using *mult-strict-left-mono [OF \S , of $PM * (1 + \xi) ^ m$] kl-choose prod-gt0*
by (*auto simp: field-simps ξ -def*)
qed
also have $\dots = \text{real } n * U\text{-lower-bound-ratio } m$
by (*simp add: U-lower-m*)
finally have *U-MINUS-M: $3 * l + 1 < \text{real } n * U\text{-lower-bound-ratio } m - m$*
by *linarith*
then have *cardU-gt: $\text{card } U > 3 * l + 1$*
using *cardU by linarith*
with *UBB.complete have $\text{card } EU > 0$ $\text{card } U > 1$*
by (*simp-all add: EU-def UBB.finV card-all-edges*)
have *BlueU-eq: $\text{BlueU} = EU \setminus \text{RedU}$*
using *Blue-eq complete by (fastforce simp: BlueU-def RedU-def EU-def V-def*
E-def)
have [*simp*]: *UBB.graph-size = card EU*
using *EU-def by blast*
have $\gamma' \leq \gamma$
using $\langle m < l \rangle \langle k > 0 \rangle$ **by** (*simp add: γ -def γ' -def field-simps*)
have *False if UBB.graph-density RedU < 1 - γ' - η*
proof – — *by maximality, etc.*
have $\S: \text{UBB.graph-density } \text{BlueU} \geq \gamma' + \eta$
using *that $\langle \text{card } EU > 0 \rangle$ card-RedU-le*
by (*simp add: BlueU-eq UBB.graph-density-def diff-divide-distrib card-Diff-subset*)
have $Nx: \text{Neighbours } \text{BlueU } x \cap (U \setminus \{x\}) = \text{Neighbours } \text{BlueU } x$ **for** x
using *that by (auto simp: BlueU-eq EU-def Neighbours-def)*
have $\text{BlueU} \subseteq E \cap \text{Pow } U$
using *BlueU-eq EU-def by blast*
with *UBB.exists-density-edge-density [of 1 BlueU]*
obtain x **where** $x \in U$ **and** $x: \text{UBB.graph-density } \text{BlueU} \leq \text{UBB.gen-density}$
 $\text{BlueU } \{x\} (U \setminus \{x\})$
by (*metis UBB.complete $\langle 1 < \text{UBB.gorder} \rangle$ card-1-singletonE insertI1*)

```

zero-less-one subsetD)
  with § have  $\gamma' + \eta \leq \text{UBB.gen-density BlueU } (U \setminus \{x\}) \{x\}$ 
  using UBB.gen-density-commute by auto
  then have *:  $(\gamma' + \eta) * (\text{card } U - 1) \leq \text{card } (\text{Neighbours BlueU } x)$ 
  using  $\langle \text{BlueU} \subseteq E \cap \text{Pow } U \rangle \langle \text{card } U > 1 \rangle \langle x \in U \rangle$ 
  by (simp add: UBB.gen-density-def UBB.edge-card-eq-sum-Neighbours UBB.finV
divide-simps Nx)

  have  $x: x \in V \setminus W$ 
  using  $\langle x \in U \rangle \langle U \subseteq V \rangle \langle \text{disjnt } U \ W \rangle$  by (auto simp: U-def disjnt-iff)
  moreover
  have is-good-clique n (insert x W)
  unfolding is-good-clique-def
  proof (intro conjI)
  show clique (insert x W) Blue
  proof (intro clique-insert)
  show clique W Blue
  using 49 is-good-clique-def by blast
  show all-edges-betw-un {x} W ⊆ Blue
  using  $\langle x \in U \rangle$  by (auto simp: U-def all-edges-betw-un-def insert-commute
in-Neighbours-iff)
  qed (use  $\langle W \subseteq V \rangle \langle x \in V \setminus W \rangle$  in auto)
  next
  show insert x W ⊆ V
  using  $\langle W \subseteq V \rangle \langle x \in V \setminus W \rangle$  by auto
  next
  have NB-Int-U: Neighbours Blue x ∩ U = Neighbours BlueU x
  using  $\langle x \in U \rangle$  by (auto simp: BlueU-def U-def Neighbours-def)
  have ulb-ins: U-lower-bound-ratio (card (insert x W)) = U-lower-bound-ratio
m * (1+ξ) * γ'
  using  $\langle x \in V \setminus W \rangle \langle \text{finite } W \rangle$  by (simp add: U-lower-bound-ratio-def γ'-def
m-def)
  have  $n * \text{U-lower-bound-ratio } (\text{card } (\text{insert } x \ W)) = n * \text{U-lower-bound-ratio}$ 
m * (1+ξ) * γ'
  by (simp add: ulb-ins)
  also have  $\dots \leq \text{real } (m + \text{card } U) * (1+\xi) * \gamma'$ 
  using mult-right-mono [OF cardU, of  $(1+\xi) * \gamma'$ ]  $\langle 0 < \eta \rangle \langle 0 < \gamma' \rangle$  η-def
  by argo
  also have  $\dots \leq m + \text{card } U * (1+\xi) * \gamma'$ 
  using mult-left-mono [OF γ'-le1, of m] by (simp add: algebra-simps)
  also have  $\dots \leq \text{Suc } m + (\gamma' + \eta) * (\text{UBB.gorder} - \text{Suc } 0)$ 
  using  $* \langle x \in V \setminus W \rangle \langle \text{finite } W \rangle \text{cardU-gt } \gamma 1516$ 
  apply (simp add: U-lower-bound-ratio-def ξ-def η-def)
  by (simp add: algebra-simps)
  also have  $\dots \leq \text{Suc } m + \text{card } (V \cap \bigcap (\text{Neighbours Blue } \text{'insert } x \ W))$ 
  using * NB-Int-U finV by (simp add: U-def Int-ac)
  also have  $\dots = \text{real } (\text{card } (\text{insert } x \ W) + \text{card } (V \cap \bigcap (\text{Neighbours Blue } \text{'$ 
insert } x \ W)))
  using  $x \langle \text{finite } W \rangle \text{VUU}$  by (auto simp: U-def m-def)

```

```

finally show  $n * U\text{-lower-bound-ratio} (\text{card}(\text{insert } x \ W)) - \text{card}(\text{insert } x \ W)$ 
   $\leq \text{card} (V \cap \bigcap (\text{Neighbours } \text{Blue } ' \text{insert } x \ W))$ 
  by simp
qed
ultimately show False
  using max49 by blast
qed
then have  $gd\text{-RedU-ge}: UBB.\text{graph-density } RedU \geq 1 - \gamma' - \eta$  by force

— Bhavik's gamma'_le_gamma_iff
have  $\gamma' \gamma 2: \gamma' < \gamma^2 \iff (\text{real } k * \text{real } l) + (\text{real } l * \text{real } l) < (\text{real } k * \text{real } m)$ 
+  $(\text{real } l * (\text{real } m * 2))$ 
  using  $\langle m < l \rangle$ 
apply (simp add:  $\gamma'$ -def  $\gamma$ -def eval-nat-numeral divide-simps; simp add: algebra-simps)
  by (metis  $\langle k > 0 \rangle$  mult-less-cancel-left-pos of-nat-0-less-iff distrib-left)
also have  $\dots \iff (l * (k+l)) / (k + 2 * l) < m$ 
  using  $\langle m < l \rangle$  by (simp add: field-simps)
finally have  $\gamma' \gamma 2\text{-iff}: \gamma' < \gamma^2 \iff (l * (k+l)) / (k + 2 * l) < m .$ 
— in both cases below, we find a blue clique of size  $l - m$ 
have extend-Blue-clique:  $\exists K'. \text{size-clique } l \ K' \ \text{Blue}$ 
if  $K \subseteq U$  size-clique  $(l-m) \ K \ \text{Blue}$  for  $K$ 
proof —
  have  $K: \text{card } K = l - m$  clique  $K \ \text{Blue}$ 
    using that by (auto simp: size-clique-def)
  define  $K'$  where  $K' \equiv K \cup W$ 
  have  $\text{card } K' = l$ 
    unfolding  $K'\text{-def}$ 
proof (subst card-Un-disjnt)
  show finite  $K$  finite  $W$ 
    using  $UBB.\text{finV} \langle K \subseteq U \rangle$  finite-subset  $\langle \text{finite } W \rangle$  by blast+
  show disjnt  $K \ W$ 
    using  $\langle \text{disjnt } U \ W \rangle \langle K \subseteq U \rangle$  disjnt-subset1 by blast
  show  $\text{card } K + \text{card } W = l$ 
    using  $K \langle m < l \rangle$  m-def by auto
qed
moreover have clique  $K' \ \text{Blue}$ 
  using  $\langle \text{clique } K \ \text{Blue} \rangle$  clique-W  $\langle K \subseteq U \rangle$ 
  unfolding  $K'\text{-def}$  size-clique-def  $U\text{-def}$ 
  by (force simp: in-Neighbours-iff insert-commute intro: Ramsey.clique-Un)
ultimately show ?thesis
  unfolding  $K'\text{-def}$  size-clique-def using  $\langle K \subseteq U \rangle \langle U \subseteq V \rangle \langle W \subseteq V \rangle$  by
auto
qed

show False
proof (cases  $\gamma' < \gamma^2$ )
  case True
  with  $\gamma' \gamma 2$  have  $YKK: \gamma * k \leq m$ 
    using  $\langle 0 < k \rangle \langle m < l \rangle$ 

```

```

apply (simp add:  $\gamma$ -def field-simps)
by (smt (verit, best) distrib-left mult-left-mono of-nat-0-le-iff)
have  $\ln 1 \xi$ :  $\ln (1 + \xi) * 20 \geq 1$ 
unfolding  $\xi$ -def by (approximation 10)
with YKK have  $\xi$ :  $m * \ln (1 + \xi) \geq \delta * k$ 
unfolding  $\delta$ -def using zero-le-one mult-mono by fastforce
have powerm:  $(1 + \xi)^m \geq \exp (\delta * k)$ 
using exp-mono [OF  $\xi$ ]
by (smt (verit)  $\eta$ -def  $\langle 0 < \eta \rangle \langle 0 < \gamma' \rangle$  exp-ln-iff exp-of-nat-mult zero-le-mult-iff)
have  $n * (1 + \xi)^m \geq (k + l \text{ choose } l)$ 
by (smt (verit, best) mult-left-mono nexp-gt of-nat-0-le-iff powerm)
then have **:  $n * U\text{-lower-bound-ratio } m \geq (k + l - m \text{ choose } (l - m))$ 
using  $\langle m < l \rangle$  prod-gt0 kl-choose by (auto simp: U-lower-m field-simps)

have m-le-choose:  $m \leq (k + l - m - 1 \text{ choose } (l - m))$ 
proof (cases  $m = 0$ )
  case False
    have  $m \leq (k + l - m - 1 \text{ choose } 1)$ 
      using  $\langle l \leq k \rangle \langle m < l \rangle$  by simp
    also have  $\dots \leq (k + l - m - 1 \text{ choose } (l - m))$ 
      using False  $\langle l \leq k \rangle \langle m < l \rangle$  by (intro binomial-mono) auto
    finally have m-le-choose:  $m \leq (k + l - m - 1 \text{ choose } (l - m))$  .
    then show ?thesis .
  qed auto
have RN k  $(l - m) \leq k + (l - m) - 2 \text{ choose } (k - 1)$ 
by (rule RN-le-choose-strong)
also have  $\dots \leq (k + l - m - 1 \text{ choose } k)$ 
using  $\langle l \leq k \rangle \langle m < l \rangle$  choose-reduce-nat by simp
also have  $\dots = (k + l - m - 1 \text{ choose } (l - m - 1))$ 
using  $\langle m < l \rangle$  by (simp add: binomial-symmetric [of k])
also have  $\dots = (k + l - m \text{ choose } (l - m)) - (k + l - m - 1 \text{ choose } (l - m))$ 
using  $\langle l \leq k \rangle \langle m < l \rangle$  choose-reduce-nat by simp
also have  $\dots \leq (k + l - m \text{ choose } (l - m)) - m$ 
using m-le-choose by linarith
finally have RN k  $(l - m) \leq (k + l - m \text{ choose } (l - m)) - m$  .
then have card U  $\geq$  RN k  $(l - m)$ 
using 49 ** VUU by (force simp: is-good-clique-def U-def m-def)
with Red-Blue-RN no-Red-K  $\langle U \subseteq V \rangle$ 
obtain K where  $K \subseteq U$  size-clique  $(l - m)$  K Blue by meson
then show False
  using no-Blue-K extend-Blue-clique by blast
next
case False
have YMK:  $\gamma - \gamma' \leq m/k$ 
using  $\langle m < l \rangle$ 
apply (simp add:  $\gamma$ -def  $\gamma'$ -def divide-simps)
apply (simp add: algebra-simps)
by (smt (verit) mult-left-mono mult-right-mono nat-less-real-le of-nat-0-le-iff)

```

```

define  $\delta'$  where  $\delta' \equiv \gamma'/20$ 
have no-RedU-K:  $\neg (\exists K. \text{UBB.size-clique } k \ K \ \text{RedU})$ 
  unfolding UBB.size-clique-def RedU-def
by (metis Int-subset-iff VUU all-edges-subset-iff-clique no-Red-K size-clique-def)
  have  $(\exists K. \text{UBB.size-clique } k \ K \ \text{RedU}) \vee (\exists K. \text{UBB.size-clique } (l-m) \ K$ 
BlueU)
  proof (rule ccontr)
    assume neg:  $\neg ((\exists K. \text{UBB.size-clique } k \ K \ \text{RedU}) \vee (\exists K. \text{UBB.size-clique}$ 
(l-m) K BlueU))
    interpret UBB-NC: No-Cliques U E  $\cap$  Pow U p0-min RedU BlueU l-m k
    proof
      show BlueU = E  $\cap$  Pow U  $\setminus$  RedU
      using BlueU-eq EU-def by fastforce
    qed (use neg EU-def  $\langle$ RedU  $\subseteq$  EU $\rangle$  no-RedU-K  $\langle$ l $\leq$ k $\rangle$  in auto)
    show False
    proof (intro UBB-NC.Far-9-2)
      have exp ( $\delta*k$ ) * exp ( $-\delta'*k$ ) = exp ( $\gamma*k/20 - \gamma'*k/20$ )
      unfolding  $\delta$ -def  $\delta'$ -def by (simp add: mult-exp-exp)
      also have  $\dots \leq \text{exp } (m/20)$ 
      using YMK  $\langle$ 0 < k $\rangle$  by (simp add: left-diff-distrib divide-simps)
      also have  $\dots \leq (1+\xi)^m$ 
      proof -
        have  $\ln (16 / 15) * 20 \geq (1::\text{real})$ 
        by (approximation 5)
        from mult-left-mono [OF this]
        show ?thesis
        by (simp add:  $\xi$ -def powr-def mult-ac flip: powr-realpow)
      qed
      finally have expexp: exp ( $\delta*k$ ) * exp ( $-\delta'*k$ )  $\leq$  (1+ $\xi$ ) ^ m .

      have exp ( $-\delta'*k$ ) * (k + (l-m) choose (l-m)) = exp ( $-\delta'*k$ ) * PM * (k+l
choose l)
      using  $\langle m < l \rangle$  kl-choose by force
      also have  $\dots < (n/2) * \text{exp } (\delta*k) * \text{exp } (-\delta'*k) * PM$ 
      using n2exp-gt prod-gt0 by auto
      also have  $\dots \leq (n/2) * (1+\xi)^m * PM$ 
      using expexp less-eq-real-def prod-gt0 by fastforce
      also have  $\dots \leq n * U\text{-lower-bound-ratio } m - m$  — where I was stuck: the
      "minus m"
      using PM-def U-MINUS-M U-lower-bound-ratio-def  $\langle$ m < l $\rangle$  by fastforce
      finally have exp ( $-\delta'*k$ ) * (k + (l-m) choose (l-m))  $\leq$  n * U-lower-bound-ratio
m - m
      by linarith
      also have  $\dots \leq \text{UBB.nV}$ 
      using cardU by linarith
      finally have exp ( $-\delta'*k$ ) * (k + (l-m) choose (l-m))  $\leq$  UBB.nV .
      then show exp ( $-\delta'*k$ ) * ((l-m) / (k + real (l-m)) / 20) * k * (k + (l-m)
choose (l-m))  $\leq$  UBB.nV
      using  $\langle m < l \rangle$  by (simp add:  $\delta'$ -def  $\gamma'$ -def) argo

```

```

next
  show  $1 - \text{real } (l-m) / (\text{real } k + \text{real } (l-m)) - \eta \leq \text{UBB.graph-density}$ 
RedU
  using gd-RedU-ge  $\langle \gamma' \leq \gamma \rangle \langle m < l \rangle$  unfolding  $\gamma\text{-def } \gamma'\text{-def}$ 
  by (smt (verit) less-or-eq-imp-le of-nat-add of-nat-diff)
  have  $p0\text{-min} \leq 1 - \gamma - \eta$ 
  using  $\langle \gamma' \leq \gamma \rangle \gamma$  p0-min-91 by (auto simp: \eta-def \xi-def)
  also have  $\dots \leq 1 - (l-m) / (\text{real } k + \text{real } (l-m)) - \eta$ 
  using  $\langle \gamma' \leq \gamma \rangle \langle m < l \rangle$  by (simp add: \gamma-def \gamma'\text{-def algebra-simps)
  finally show  $p0\text{-min} \leq 1 - (l-m) / (\text{real } k + \text{real } (l-m)) - \eta$  .
next
  have  $m \leq l * (k + \text{real } l) / (k + 2 * \text{real } l)$ 
  using False \gamma'\gamma2-iff by auto
  also have  $\dots \leq l * (1 - (10/11)*\gamma)$ 
  using  $\gamma \langle l > 0 \rangle$  by (simp add: \gamma-def field-split-simps)
  finally have  $m \leq \text{real } l * (1 - (10/11)*\gamma)$ 
  by force
  then have  $\text{real } l - \text{real } m \geq (10/11) * \gamma * l$ 
  by (simp add: algebra-simps)
  then have Big-Far-9-2  $\gamma' (l-m)$ 
  using False big  $\langle \gamma' \leq \gamma \rangle \gamma \langle m < l \rangle$ 
  by (simp add: Big-Far-9-1-def)
  then show Big-Far-9-2  $((l-m) / (\text{real } k + \text{real } (l-m))) (l-m)$ 
  by (simp add: \gamma'\text{-def} \langle m < l \rangle add-diff-eq less-or-eq-imp-le)
  show  $(l-m) / (\text{real } k + \text{real } (l-m)) \leq 1/10$ 
  using  $\gamma \gamma\text{-def} \langle m < l \rangle$  by fastforce
  show  $0 \leq \eta$ 
  using  $\langle 0 < \eta \rangle$  by linarith
  show  $\eta \leq (l-m) / (\text{real } k + \text{real } (l-m)) / 15$ 
  using mult-right-mono [OF  $\langle \gamma' \leq \gamma \rangle$ , of  $\xi$ ]
  by (simp add: \eta-def \gamma'\text{-def} \langle m < l \rangle \xi\text{-def add-diff-eq less-or-eq-imp-le
mult.commute)
  qed
  qed
  with no-RedU-K obtain  $K$  where  $K \subseteq U$  UBB.size-clique  $(l-m)$   $K$  BlueU
  by (meson UBB.size-clique-def)
  then show False
  using no-Blue-K extend-Blue-clique VUU
  unfolding UBB.size-clique-def size-clique-def BlueU-def
  by (metis Int-subset-iff all-edges-subset-iff-clique)
  qed
  qed
end
end

```

9 An exponential improvement closer to the diagonal

theory *Closer-To-Diagonal*
imports *Far-From-Diagonal*

begin

9.1 Lemma 10.2

context *P0-min*
begin

lemma *error-10-2*:

assumes $\mu / \text{real } d > 1/200$

shows $\forall^\infty k. \text{ok-fun-95b } k + \mu * \text{real } k / \text{real } d \geq k/200$

proof –

have $d > 0 \ \mu > 0$

using *assms* **by** (*auto simp: divide-simps split: if-split-asm*)

then have $*: \text{real } k \leq \mu * (\text{real } k * 200) / \text{real } d$ **for** k

using *assms* **by** (*fastforce simp: divide-simps less-eq-real-def*)

have $\forall^\infty k. |\text{ok-fun-95b } k| \leq (\mu/d - 1/200) * k$

using *ok-fun-95b assms* **unfolding** *smallo-def*

by (*auto dest!: spec [where x = μ/d]*)

then show *?thesis*

apply *eventually-elim*

using *assms <d>0* *

by (*simp add: algebra-simps not-less abs-if add-increasing split: if-split-asm*)

qed

The "sufficiently large" assumptions are problematical. The proof's calculation for $(3::'a) / (20::'a) < \gamma$ is sharp. We need a finite gap for the limit to exist. We can get away with $1/300$.

definition *x320::real* **where** $x320 \equiv 3/20 + 1/300$

lemma *error-10-2-True*: $\forall^\infty k. \text{ok-fun-95b } k + x320 * \text{real } k / \text{real } 30 \geq k/200$

unfolding *x320-def*

by (*intro error-10-2*) *auto*

lemma *error-10-2-False*: $\forall^\infty k. \text{ok-fun-95b } k + (1/10) * \text{real } k / \text{real } 15 \geq k/200$

by (*intro error-10-2*) *auto*

definition *Big-Closer-10-2* $\equiv \lambda\mu l. \text{Big-Far-9-3 } \mu l \wedge \text{Big-Far-9-5 } \mu l$

$\wedge (\forall k \geq l. \text{ok-fun-95b } k + (\text{if } \mu > x320 \text{ then } \mu*k/30 \text{ else } \mu*k/15) \geq k/200)$

lemma *Big-Closer-10-2*:

assumes $1/10 \leq \mu 1 \ \mu 1 < 1$

shows $\forall^\infty l. \forall \mu. 1/10 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow \text{Big-Closer-10-2 } \mu l$

```

proof –
  have  $T: \forall^\infty l. \forall k \geq l. (\forall \mu. x320 \leq \mu \wedge \mu \leq \mu1 \longrightarrow k/200 \leq ok\text{-fun-95b } k + \mu*k / \text{real } 30)$ 
    using assms
    apply (intro eventually-all-ge-at-top eventually-all-geI0 error-10-2-True)
    apply (auto simp: mult-right-mono elim!: order-trans)
    done
  have  $F: \forall^\infty l. \forall k \geq l. (\forall \mu. 1/10 \leq \mu \wedge \mu \leq \mu1 \longrightarrow k/200 \leq ok\text{-fun-95b } k + \mu*k / \text{real } 15)$ 
    using assms
    apply (intro eventually-all-ge-at-top eventually-all-geI0 error-10-2-False)
    by (smt (verit, ccfv-SIG) divide-right-mono mult-right-mono of-nat-0-le-iff)
  have  $\forall^\infty l. \forall k \geq l. (\forall \mu. 1/10 \leq \mu \wedge \mu \leq \mu1 \longrightarrow k/200 \leq ok\text{-fun-95b } k + (\text{if } \mu > x320 \text{ then } \mu*k/30 \text{ else } \mu*k/15))$ 
    using assms
    apply (split if-split)
    unfolding eventually-conj-iff all-imp-conj-distrib all-conj-distrib
    by (force intro: eventually-mono [OF T] eventually-mono [OF F])
  then show ?thesis
    using assms Big-Far-9-3[of 1/10] Big-Far-9-5[of 1/10]
    unfolding Big-Closer-10-2-def eventually-conj-iff all-imp-conj-distrib
    by (force simp: elim!: eventually-mono)
qed

end

```

A little tricky to express since the Book locale assumes that there are no cliques in the original graph (page 10). So it's a contrapositive

lemma (in *Book*) *Closer-10-2-aux*:

```

assumes  $0: \text{real } (\text{card } X0) \geq nV/2 \text{ card } Y0 \geq nV \text{ div } 2 \text{ } p0 \geq 1-\gamma$ 
  — These are the assumptions about the red density of the graph
assumes  $\gamma: 1/10 \leq \gamma \ \gamma \leq 1/5$ 
assumes  $nV: \text{real } nV \geq \exp(-k/200) * (k+l \text{ choose } l)$ 
assumes big: Big-Closer-10-2  $\gamma \ l$ 
shows False

```

proof –

```

define  $\mathcal{R}$  where  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ 
define  $t$  where  $t \equiv \text{card } \mathcal{R}$ 
define  $\delta::\text{real}$  where  $\delta \equiv 1/200$ 
have  $\gamma01: 0 < \gamma \ \gamma < 1$ 
  using ln0 l-le-k by (auto simp: \gamma-def)
have  $t < k$ 
  unfolding t-def \mathcal{R}-def using  $\gamma01$  red-step-limit by blast
have big93: Big-Far-9-3  $\gamma \ l$ 
  using big by (auto simp: Big-Closer-10-2-def Big-Far-9-2-def)
have  $t23: t \geq 2*k / 3$ 
  unfolding t-def \mathcal{R}-def
proof (rule Far-9-3)
  have  $\min(1/200) (l / (\text{real } k + \text{real } l) / 20) = 1/200$ 

```

```

    using  $\gamma$  ln0 by (simp add:  $\gamma$ -def)
  then show  $\exp(-\min(1/200)(\gamma/20) * \text{real } k) * \text{real } (k+l \text{ choose } l) \leq nV$ 
    using  $nV$  divide-real-def inverse-eq-divide minus-mult-right mult commute
 $\gamma$ -def
  by (metis of-int-of-nat-eq of-int-minus)
  show  $1/4 \leq p0$ 
    using  $\gamma$  0 by linarith
  show Big-Far-9-3  $\gamma$  l
    using  $\gamma$ -def big93 by blast
  qed (use assms  $\gamma$ -def in auto)

  have card (Yseq halted-point)  $\geq$ 
     $\exp(-\delta * k + \text{ok-fun-95b } k) * (1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) *$ 
 $((1-\gamma)/(1-\gamma))^{t}$ 
    *  $\exp(\gamma * (\text{real } t)^2 / (2*k)) * (k-t+l \text{ choose } l)$ 
  proof (rule order-trans [OF - Far-9-5])
    show  $\exp(-\delta * k) * \text{real } (k+l \text{ choose } l) \leq \text{real } nV$ 
      using  $nV$  by (auto simp:  $\delta$ -def)
    show  $1/2 \leq 1 - \gamma - 0$ 
      using divide-le-eq-1 l-le-k  $\gamma$ -def by fastforce
  next
    show Big-Far-9-5  $\gamma$  l
      using big by (simp add: Big-Closer-10-2-def Big-Far-9-2-def  $\gamma$ -def)
    qed (use 0 kn0 in <auto simp flip: t-def  $\gamma$ -def  $\mathcal{R}$ -def>)
    then have 52: card (Yseq halted-point)  $\geq$ 
       $\exp(-\delta * k + \text{ok-fun-95b } k) * (1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) * \exp(\gamma$ 
*  $(\text{real } t)^2 / (2*k)) * (k-t+l \text{ choose } l)$ 
      using  $\gamma$  by simp

  define gamf where gamf  $\equiv \lambda x::\text{real}. (1-x) \text{ powr } (1/(1-x))$ 
  have deriv-gamf:  $\exists y. \text{DERIV } \text{gamf } x :> y \wedge y \leq 0$  if  $0 < a \leq x \leq b < 1$  for
  a b x
    unfolding gamf-def
    using that ln-less-self[of 1-x]
    by (force intro!: DERIV-powr derivative-eq-intros simp: divide-simps mult-le-0-iff
simp del: ln-less-self)
    have  $(1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) * \exp(\gamma * (\text{real } t)^2 / (2*k)) \geq \exp(\delta * k -$ 
 $\text{ok-fun-95b } k)$ 
    proof (cases  $\gamma > x320$ )
      case True
        then have  $\text{ok-fun-95b } k + \gamma * k / 30 \geq k/200$ 
          using big l-le-k by (auto simp: Big-Closer-10-2-def Big-Far-9-2-def)
        with True kn0 have  $\delta * k - \text{ok-fun-95b } k \leq (\gamma/30) * k$ 
          by (simp add:  $\delta$ -def)
        also have  $\dots \leq 3 * \gamma * (\text{real } t)^2 / (40*k)$ 
          using True mult-right-mono [OF mult-mono [OF t23 t23], of  $3*\gamma / (40*k)$ ]
          <k>0>
          by (simp add: power2-eq-square x320-def)
        finally have †:  $\delta * k - \text{ok-fun-95b } k \leq 3 * \gamma * (\text{real } t)^2 / (40*k)$  .

```

```

have gamf  $\gamma \geq \text{gamf } (1/5)$ 
  by (smt (verit, best) DERIV-nonpos-imp-nonincreasing[of  $\gamma$  1/5 gamf]  $\gamma$ 
 $\gamma 01$  deriv-gamf divide-less-eq-1)
moreover have  $\ln (\text{gamf } (1/5)) \geq -1/3 + 1/20$ 
  unfolding gamf-def by (approximation 10)
moreover have  $\text{gamf } (1/5) > 0$ 
  by (simp add: gamf-def)
ultimately have  $\text{gamf } \gamma \geq \exp (-1/3 + 1/20)$ 
  using ln-ge-iff by auto
from powr-mono2 [OF - - this]
have  $(1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) \geq \exp (-17/60) \text{ powr } (\gamma * t)$ 
  unfolding gamf-def using  $\gamma 01$  powr-powr by fastforce
from mult-left-mono [OF this, of  $\exp (\gamma * (\text{real } t)^2 / (2 * k))$ ]
have  $(1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) * \exp (\gamma * (\text{real } t)^2 / (2 * k)) \geq \exp (-17/60$ 
 $* (\gamma * t) + (\gamma * (\text{real } t)^2 / (2 * k)))$ 
  by (smt (verit) mult.commute exp-add exp-ge-zero exp-powr-real)
moreover have  $(-17/60 * (\gamma * t) + (\gamma * (\text{real } t)^2 / (2 * k))) \geq (3 * \gamma * (\text{real } t)^2$ 
 $/ (40 * k))$ 
  using t23  $\langle k > 0 \rangle$   $\langle \gamma > 0 \rangle$  by (simp add: divide-simps eval-nat-numeral)
ultimately have  $(1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) * \exp (\gamma * (\text{real } t)^2 / (2 * k)) \geq$ 
 $\exp (3 * \gamma * (\text{real } t)^2 / (40 * k))$ 
  by (smt (verit) exp-mono)
with † show ?thesis
  by (smt (verit, best) exp-le-cancel-iff)
next
case False
then have  $ok\text{-fun-95b } k + \gamma * k / 15 \geq k / 200$ 
  using big l-le-k by (auto simp: Big-Closer-10-2-def Big-Far-9-2-def)
with kn0 have  $\delta * k - ok\text{-fun-95b } k \leq (\gamma / 15) * k$ 
  by (simp add:  $\delta$ -def x320-def)
also have  $\dots \leq 3 * \gamma * (\text{real } t)^2 / (20 * k)$ 
  using  $\gamma$  mult-right-mono [OF mult-mono [OF t23 t23], of  $3 * \gamma / (40 * k)$ ] kn0
  by (simp add: power2-eq-square field-simps)
finally have †:  $\delta * k - ok\text{-fun-95b } k \leq 3 * \gamma * (\text{real } t)^2 / (20 * k) .$ 

have gamf  $\gamma \geq \text{gamf } x320$ 
  using False  $\gamma$ 
  by (intro DERIV-nonpos-imp-nonincreasing[of  $\gamma$  x320 gamf] deriv-gamf)
  (auto simp: x320-def)
moreover have  $\ln (\text{gamf } x320) \geq -1/3 + 1/10$ 
  unfolding gamf-def x320-def by (approximation 6)
moreover have  $\text{gamf } x320 > 0$ 
  by (simp add: gamf-def x320-def)
ultimately have  $\text{gamf } \gamma \geq \exp (-1/3 + 1/10)$ 
  using ln-ge-iff by auto
from powr-mono2 [OF - - this]
have  $(1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) \geq \exp (-7/30) \text{ powr } (\gamma * t)$ 
  unfolding gamf-def using  $\gamma 01$  powr-powr by fastforce

```

```

from mult-left-mono [OF this, of  $\exp(\gamma * (\text{real } t)^2 / (2*k))$ ]
have  $(1-\gamma) \text{ powr } (\gamma*t / (1-\gamma)) * \exp(\gamma * (\text{real } t)^2 / (2*k)) \geq \exp(-7/30 * (\gamma*t) + (\gamma * (\text{real } t)^2 / (2*k)))$ 
by (smt (verit) mult.commute exp-add exp-ge-zero exp-powr-real)
moreover have  $(-7/30 * (\gamma*t) + (\gamma * (\text{real } t)^2 / (2*k))) \geq (3*\gamma * (\text{real } t)^2 / (20*k))$ 
using t23 <k>0 <gamma>0 by (simp add: divide-simps eval-nat-numeral)
ultimately have  $(1-\gamma) \text{ powr } (\gamma*t / (1-\gamma)) * \exp(\gamma * (\text{real } t)^2 / (2*k)) \geq \exp(3*\gamma * (\text{real } t)^2 / (20*k))$ 
by (smt (verit) exp-mono)
with † show ?thesis
by (smt (verit, best) exp-le-cancel-iff)
qed
then have  $1 \leq \exp(-\delta*k + \text{ok-fun-95b } k) * (1-\gamma) \text{ powr } (\gamma * t / (1-\gamma)) * \exp(\gamma * (\text{real } t)^2 / (2 * k))$ 
by (simp add: exp-add exp-diff mult-ac pos-divide-le-eq)
then have  $(k-t+l \text{ choose } l) \leq \exp(-\delta * k + \text{ok-fun-95b } k) * (1-\gamma) \text{ powr } (\gamma*t / (1-\gamma)) * \exp(\gamma * (\text{real } t)^2 / (2*k)) * (k-t+l \text{ choose } l)$ 
by auto
with 52 have  $(k-t+l \text{ choose } l) \leq \text{card } (Y \text{seq halted-point})$  by linarith
then show False
using Off-diagonal-conclusion by (simp flip: R-def t-def)
qed

```

Material that needs to be proved **outside** the book locales

lemma (in *No-Cliques*) *Closer-10-2*:

```

fixes  $\gamma::\text{real}$ 
defines  $\gamma \equiv l / (\text{real } k + \text{real } l)$ 
assumes nV:  $\text{real } nV \geq \exp(-\text{real } k/200) * (k+l \text{ choose } l)$ 
assumes gd: graph-density Red  $\geq 1-\gamma$  and p0-min-OK:  $p0\text{-min} \leq 1-\gamma$ 
assumes big: Big-Closer-10-2  $\gamma \ l$  and  $l \leq k$ 
assumes  $\gamma: 1/10 \leq \gamma \ \gamma \leq 1/5$ 
shows False
proof –
obtain X0 Y0 where  $l \geq 2$  and card-X0:  $\text{card } X0 \geq nV/2$ 
and card-Y0:  $\text{card } Y0 = \text{gorder div } 2$ 
and X0-def:  $X0 = V \setminus Y0$  and  $Y0 \subseteq V$ 
and gd-le: graph-density Red  $\leq \text{gen-density Red } X0 \ Y0$ 
and Book' V E p0-min Red Blue l k gamma X0 Y0
using to-Book' assms order.trans ln0 by blast
then interpret Book' V E p0-min Red Blue l k gamma X0 Y0
by blast
show False
proof (intro Closer-10-2-aux)
show  $1 - \gamma \leq p0$ 
using X0-def gamma-def gd gd-le gen-density-commute p0-def by auto
qed (use assms card-X0 card-Y0 in auto)
qed

```

9.2 Theorem 10.1

context *P0-min*

begin

definition *Big101a* $\equiv \lambda k. 2 + \text{real } k / 2 \leq \exp (\text{of-int} \lfloor k/10 \rfloor * 2 - k/200)$

definition *Big101b* $\equiv \lambda k. (\text{real } k)^2 - 10 * \text{real } k > (k/10) * \text{real}(10 + 9*k)$

The proof considers a smaller graph, so l needs to be so big that the smaller l' will be big enough.

definition *Big101c* $\equiv \lambda \gamma 0 l. \forall l' \gamma. l' \geq \text{nat } \lfloor 2/5 * l \rfloor \longrightarrow \gamma 0 \leq \gamma \longrightarrow \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-1 } \gamma l'$

definition *Big101d* $\equiv \lambda l. (\forall l' \gamma. l' \geq \text{nat } \lfloor 2/5 * l \rfloor \longrightarrow 1/10 \leq \gamma \longrightarrow \gamma \leq 1/5 \longrightarrow \text{Big-Closer-10-2 } \gamma l')$

definition *Big-Closer-10-1* $\equiv \lambda \gamma 0 l. l \geq 9 \wedge (\forall k \geq l. \text{Big101c } \gamma 0 k \wedge \text{Big101d } k \wedge \text{Big101a } k \wedge \text{Big101b } k)$

lemma *Big-Closer-10-1-upward*: $\llbracket \text{Big-Closer-10-1 } \gamma 0 l; l \leq k; \gamma 0 \leq \gamma \rrbracket \Longrightarrow \text{Big-Closer-10-1 } \gamma k$

unfolding *Big-Closer-10-1-def* *Big101c-def* **by** (*meson order.trans*)

The need for $\gamma 0$ is unfortunate, but it seems simpler to hide the precise value of this term in the main proof.

lemma *Big-Closer-10-1*:

fixes $\gamma 0 :: \text{real}$

assumes $\gamma 0 > 0$

shows $\forall^\infty l. \text{Big-Closer-10-1 } \gamma 0 l$

proof –

have $a: \forall^\infty k. \text{Big101a } k$

unfolding *Big101a-def* **by** *real-asymp*

have $b: \forall^\infty k. \text{Big101b } k$

unfolding *Big101b-def* **by** *real-asymp*

have $c: \forall^\infty l. \text{Big101c } \gamma 0 l$

proof –

have $\forall^\infty l. \forall \gamma. \gamma 0 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-1 } \gamma l$

using *Big-Far-9-1* $\langle \gamma 0 > 0 \rangle$ *eventually-sequentially order.trans* **by** *blast*

then obtain N **where** $N: \forall l \geq N. \forall \gamma. \gamma 0 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow \text{Big-Far-9-1}$

γl

using *eventually-sequentially* **by** *auto*

define M **where** $M \equiv \text{nat} \lfloor 5 * N / 2 \rfloor$

have $\text{nat} \lfloor (2/5) * l \rfloor \geq N$ **if** $l \geq M$ **for** l

using *that assms* **by** (*simp add: M-def le-nat-floor*)

with N **have** $\forall l \geq M. \forall l' \gamma. \text{nat} \lfloor (2/5) * l \rfloor \leq l' \longrightarrow \gamma 0 \leq \gamma \wedge \gamma \leq 1/10 \longrightarrow$

Big-Far-9-1 $\gamma l'$

by (*meson order.trans*)

then show *?thesis*

by (*auto simp: Big101c-def eventually-sequentially*)

```

qed
have  $d: \forall^\infty l. \text{Big101d } l$ 
proof –
  have  $\forall^\infty l. \forall \gamma. 1/10 \leq \gamma \wedge \gamma \leq 1/5 \longrightarrow \text{Big-Closer-10-2 } \gamma \ l$ 
    using assms Big-Closer-10-2 [of 1/5] by linarith
  then obtain  $N$  where  $N: \forall l \geq N. \forall \gamma. 1/10 \leq \gamma \wedge \gamma \leq 1/5 \longrightarrow \text{Big-Closer-10-2}$ 
 $\gamma \ l$ 
    using eventually-sequentially by auto
  define  $M$  where  $M \equiv \text{nat}[5 * N / 2]$ 
  have  $\text{nat}[(2/5) * l] \geq N$  if  $l \geq M$  for  $l$ 
    using that assms by (simp add: M-def le-nat-floor)
  with  $N$  have  $\forall l \geq M. \forall l'. \gamma. l' \geq \text{nat} [2/5 * l] \longrightarrow 1/10 \leq \gamma \wedge \gamma \leq 1/5 \longrightarrow$ 
 $\text{Big-Closer-10-2 } \gamma \ l'$ 
    by (smt (verit, ccfv-SIG) of-nat-le-iff)
  then show ?thesis
    by (auto simp: eventually-sequentially Big101d-def)
qed
show ?thesis
  using  $a \ b \ c \ d$  eventually-all-ge-at-top eventually-ge-at-top
  unfolding Big-Closer-10-1-def eventually-conj-iff all-imp-conj-distrib
  by blast
qed

```

The strange constant $\gamma 0$ is needed for the case where we consider a subgraph; see near the end of this proof

theorem *Closer-10-1:*

```

fixes  $l \ k::\text{nat}$ 
fixes  $\delta \ \gamma::\text{real}$ 
defines  $\gamma \equiv \text{real } l / (\text{real } k + \text{real } l)$ 
defines  $\delta \equiv \gamma / 40$ 
defines  $\gamma 0 \equiv \min \gamma \ (0.07)$  — Since  $36 \leq k$ , the lower bound  $1 / (10::'a) - 1$ 
/  $(36::'a)$  works
assumes big: Big-Closer-10-1  $\gamma 0 \ l$ 
assumes  $\gamma: \gamma \leq 1/5$ 
assumes p0-min-101: p0-min  $\leq 1 - 1/5$ 
shows  $RN \ k \ l \leq \exp(-\delta * k + 3) * (k+l \text{ choose } l)$ 
proof (rule ccontr)
assume non:  $\neg RN \ k \ l \leq \exp(-\delta * k + 3) * (k+l \text{ choose } l)$ 
have  $l \leq k$ 
  using  $\gamma\text{-def } \gamma \ \text{nat-le-real-less}$  by fastforce
moreover have  $l \geq 9$ 
  using big by (simp add: Big-Closer-10-1-def)
ultimately have  $l > 0 \ k > 0 \ l \geq 3$  by linarith+
then have  $l 4k: 4 * l \leq k$ 
  using  $\gamma$  by (auto simp:  $\gamma\text{-def divide-simps}$ )
have  $k \geq 36$ 
  using  $\langle l \geq 9 \rangle \ l 4k$  by linarith
have exp-gt21: exp  $(x + 2) > \exp(x + 1)$  for  $x::\text{real}$ 
by auto

```

```

have exp2: exp (2::real) = exp 1 * exp 1
  by (simp add: mult-exp-exp)
have Big91-I:  $\bigwedge l' \mu. \llbracket l' \geq \text{nat } \lfloor 2/5 * l \rfloor; \gamma 0 \leq \mu; \mu \leq 1/10 \rrbracket \implies \text{Big-Far-9-1}$ 
 $\mu l'$ 
  using big by (meson Big101c-def Big-Closer-10-1-def order.refl)
show False
proof (cases  $\gamma \leq 1/10$ )
  case True
  have  $\gamma > 0$ 
    using  $\langle 0 < l \rangle$   $\gamma$ -def by auto
  have  $RN\ k\ l \leq \text{exp } (-\delta * k + 1) * (k+l\ \text{choose } l)$ 
  proof (intro order.trans [OF Far-9-1] strip)
    show Big-Far-9-1 (l / (real k + real l)) l
    proof (intro Big91-I)
      show  $l \geq \text{nat } \lfloor 2/5 * l \rfloor$ 
        by linarith
      qed (use True  $\gamma 0$ -def  $\gamma$ -def in auto)
    next
    show  $\text{exp } (- (l / (k + \text{real } l) / 20) * k + 1) * (k+l\ \text{choose } l) \leq \text{exp } (-\delta * k + 1) * (k+l\ \text{choose } l)$ 
      by (smt (verit, best)  $\langle 0 < \gamma \rangle$   $\gamma$ -def  $\delta$ -def exp-mono frac-le mult-right-mono of-nat-0-le-iff)
    qed (use  $\langle l \geq 9 \rangle$  p0-min-101 True  $\gamma$ -def in auto)
  then show False
    using non-exp-gt21 by (smt (verit, ccfv-SIG) mult-right-mono of-nat-0-le-iff)
  next
  case False
  with  $\langle l > 0 \rangle$  have  $\gamma > 0$   $\gamma > 1/10$  and k9l:  $k < 9 * l$ 
    by (auto simp:  $\gamma$ -def)
  — Much overlap with the proof of 9.2, but key differences too
  define U-lower-bound-ratio where
    U-lower-bound-ratio  $\equiv \lambda m. (\prod_{i < m}. (l - \text{real } i) / (k+l - \text{real } i))$ 
  define n where  $n \equiv \text{nat } \lceil RN\ k\ l - 1 \rceil$ 
  have  $k \geq 12$ 
    using  $\llcorner k \langle l \geq 3 \rangle$  by linarith
  have  $\text{exp } 1 / (\text{exp } 1 - 2) < (12::\text{real})$ 
    by (approximation 5)
  also have  $RN12: \dots \leq RN\ k\ l$ 
    by (meson RN-3plus'  $\langle l \geq 3 \rangle$   $\langle k \geq 12 \rangle$  le-trans numeral-le-real-of-nat-iff)
  finally have  $\text{exp } 1 / (\text{exp } 1 - 2) < RN\ k\ l$  .
  moreover have  $n < RN\ k\ l$ 
    using RN12 by (simp add: n-def)
  moreover have  $2 < \text{exp } (1::\text{real})$ 
    by (approximation 5)
  ultimately have  $nRNn: n/2 > RN\ k\ l / \text{exp } 1$ 
    by (simp add: n-def field-split-simps)

have  $(k+l\ \text{choose } l) / \text{exp } (-3 + \delta * k) < RN\ k\ l$ 
  by (smt (verit) divide-inverse exp-minus mult-minus-left mult-of-nat-commute

```

non)
then have $(k+l \text{ choose } l) < (RN \ k \ l / \text{exp } 2) * \text{exp } (\delta*k - 1)$
by (*simp add: divide-simps exp-add exp-diff flip: exp-add*)
also have $\dots \leq (n/2) * \text{exp } (\delta*k - 2)$
using *nRNe* **by** (*simp add: divide-simps exp-diff*)
finally have $n2\text{exp-gt}': (n/2) * \text{exp } (\delta*k) > (k+l \text{ choose } l) * \text{exp } 2$
by (*metis exp-diff exp-gt-zero linorder-not-le pos-divide-le-eq times-divide-eq-right*)
then have $n2\text{exp-gt}: (n/2) * \text{exp } (\delta*k) > (k+l \text{ choose } l)$
by (*smt (verit, best) mult-le-cancel-left1 of-nat-0-le-iff one-le-exp-iff*)
then have $n\text{exp-gt}: n * \text{exp } (\delta*k) > (k+l \text{ choose } l)$
using *less-le-trans linorder-not-le* **by force**

define *V* **where** $V \equiv \{..<n\}$
define *E* **where** $E \equiv \text{all-edges } V$
interpret *Book-Basis* *V E*
proof qed (*auto simp: V-def E-def comp-sgraph.wellformed comp-sgraph.two-edges*)
have [*simp*]: $nV = n$
by (*simp add: V-def*)
then obtain *Red Blue*
where *Red-E*: $Red \subseteq E$ **and** *Blue-def*: $Blue = E - Red$
and *no-Red-K*: $\neg (\exists K. \text{size-clique } k \ K \ Red)$
and *no-Blue-K*: $\neg (\exists K. \text{size-clique } l \ K \ Blue)$
by (*metis <n < RN k l> less-RN-Red-Blue*)
have *Blue-E*: $Blue \subseteq E$ **and** *disjnt-Red-Blue*: *disjnt* *Red Blue* **and** *Blue-eq*:
Blue = all-edges V - Red
using *complete* **by** (*auto simp: Blue-def disjnt-iff E-def*)
define *is-good-clique* **where**
is-good-clique $\equiv \lambda i \ K. \text{clique } K \ Blue \wedge K \subseteq V$
 $\wedge \text{card } (V \cap (\bigcap_{w \in K} \text{Neighbours } Blue \ w))$
 $\geq i * U\text{-lower-bound-ratio } (\text{card } K) - \text{card } K$
have *is-good-card*: $\text{card } K < l$ **if** *is-good-clique* *i K* **for** *i K*
using *no-Blue-K* **that** **unfolding** *is-good-clique-def*
by (*metis nat-neq-iff size-clique-def size-clique-smaller*)
define *max-m* **where** $\text{max-m} \equiv \text{Suc } (\text{nat } \lfloor l - k/9 \rfloor)$
define *GC* **where** $GC \equiv \{C. \text{is-good-clique } n \ C \wedge \text{card } C \leq \text{max-m}\}$
have *maxm-bounds*: $l - k/9 \leq \text{max-m} \ \text{max-m} \leq l+1 - k/9 \ \text{max-m} > 0$
using *k9l* **unfolding** *max-m-def* **by** *linarith+*
then have $GC \neq \{\}$
by (*auto simp: GC-def is-good-clique-def U-lower-bound-ratio-def E-def V-def*)
intro: exI [where x={}])
have $GC \subseteq \text{Pow } V$
by (*auto simp: is-good-clique-def GC-def*)
then have *finite GC*
by (*simp add: finV finite-subset*)
then obtain *W* **where** $W \in GC$ **and** *MaxW*: $\text{Max } (\text{card } 'GC) = \text{card } W$
using $\langle GC \neq \{\} \rangle$ *obtains-MAX* **by** *blast*
then have *53*: *is-good-clique* *n W*
using *GC-def* **by** *blast*
then have $W \subseteq V$

by (*auto simp: is-good-clique-def*)

define m **where** $m \equiv \text{card } W$
define γ' **where** $\gamma' \equiv (l - \text{real } m) / (k+l - \text{real } m)$

have max53 : $\neg (\text{is-good-clique } n (\text{insert } x W) \wedge \text{card } (\text{insert } x W) \leq \text{max-m})$
if $x \in V \setminus W$ **for** x

proof — Setting up the case analysis for γ'
assume x : $\text{is-good-clique } n (\text{insert } x W) \wedge \text{card } (\text{insert } x W) \leq \text{max-m}$
then have $\text{card } (\text{insert } x W) = \text{Suc } (\text{card } W)$
using $\text{finV is-good-clique-def finite-subset that by fastforce}$
with $x \langle \text{finite } GC \rangle$ **have** $\text{Max } (\text{card } \langle GC \rangle) \geq \text{Suc } (\text{card } W)$
by (*metis (no-types, lifting) GC-def Max-ge finite-imageI image-iff mem-Collect-eq*)
then show *False*
by (*simp add: MaxW*)

qed
then have clique-cases : $m < \text{max-m} \wedge (\forall x \in V \setminus W. \neg \text{is-good-clique } n (\text{insert } x W)) \vee m = \text{max-m}$
using $GC\text{-def } \langle W \in GC \rangle \langle W \subseteq V \rangle \text{finV finite-subset m-def by fastforce}$

have Red-Blue-RN : $\exists K \subseteq X. \text{size-clique } m K \text{ Red} \vee \text{size-clique } n K \text{ Blue}$
if $\text{card } X \geq \text{RN } m n X \subseteq V$ **for** $m n$ **and** X
using $\text{partn-lst-imp-is-clique-RN [OF is-Ramsey-number-RN [of m n]] finV}$

that
unfolding $\text{is-clique-RN-def size-clique-def clique-indep-def Blue-eq}$
by (*metis clique-iff-indep finite-subset subset-trans*)
define U **where** $U \equiv V \cap (\bigcap_{w \in W. \text{Neighbours Blue } w})$
have $\text{RN } k l > 0$
by (*metis RN-eq-0-iff gr0I <k>0 <l>0*)
with $\langle n < \text{RN } k l \rangle$ **have** $n\text{-less}$: $n < (k+l \text{ choose } l)$
by (*metis add.commute RN-commute RN-le-choose le-trans linorder-not-less*)

have $\gamma' > 0$
using $\text{is-good-card [OF 53]}$ **by** (*simp add: \(\gamma'\)-def m-def*)
have $\text{finite } W$
using $\langle W \subseteq V \rangle \text{finV finite-subset by (auto simp: V-def)}$
have $U \subseteq V$
by (*force simp: U-def*)
then have VUU : $V \cap U = U$
by *blast*
have $\text{disjnt } U W$
using $\text{Blue-E not-own-Neighbour unfolding E-def V-def U-def disjnt-iff by blast}$

have $m < l$
using $53 \text{ is-good-card m-def by blast}$
have $\gamma' \leq 1$
using $\langle m < l \rangle$ **by** (*simp add: \(\gamma'\)-def divide-simps*)

have $\text{card } U$: $n * U\text{-lower-bound-ratio } m \leq m + \text{card } U$

```

    using 53 VUU unfolding is-good-clique-def m-def U-def by force
  have clique-W: size-clique m W Blue
    using 53 is-good-clique-def m-def size-clique-def V-def by blast
  have prod-gt0: U-lower-bound-ratio m > 0
    unfolding U-lower-bound-ratio-def using <m<l> by (intro prod-pos) auto
  have kl-choose: real(k+l choose l) = (k+l-m choose (l-m)) / U-lower-bound-ratio
  m
    unfolding U-lower-bound-ratio-def using kl-choose <0 <k> <m <l> by blast

— in both cases below, we find a blue clique of size l - m
  have extend-Blue-clique:  $\exists K'. \text{size-clique } l \ K' \ \text{Blue}$ 
    if  $K \subseteq U$  size-clique (l-m) K Blue for K
  proof -
    have K: card K = l-m clique K Blue
      using that by (auto simp: size-clique-def)
    define K' where  $K' \equiv K \cup W$ 
    have card K' = l
      unfolding K'-def
    proof (subst card-Un-disjnt)
      show finite K finite W
        using finV <K  $\subseteq$  U> <U  $\subseteq$  V> finite-subset <finite W> that by meson+
      show disjnt K W
        using <disjnt U W> <K  $\subseteq$  U> disjnt-subset1 by blast
      show card K + card W = l
        using K <m <l> m-def by auto
    qed
    moreover have clique K' Blue
      using <clique K Blue> clique-W <K  $\subseteq$  U>
      unfolding K'-def size-clique-def U-def
      by (force simp: in-Neighbours-iff insert-commute intro: Ramsey.clique-Un)
    ultimately show ?thesis
      unfolding K'-def size-clique-def using <K  $\subseteq$  U> <U  $\subseteq$  V> <W  $\subseteq$  V> by
  auto
  qed

  have  $\gamma' \leq \gamma$ 
    using <m<l> by (simp add:  $\gamma$ -def  $\gamma'$ -def field-simps)

  consider m < max-m | m = max-m
    using clique-cases by blast
  then consider m < max-m  $\gamma' \geq 1/10$  |  $1/10 - 1/k \leq \gamma' \wedge \gamma' \leq 1/10$ 
  proof cases
    case 1
      then have  $\gamma' \geq 1/10$ 
        using < $\gamma > 1/10\gamma$ -def  $\gamma'$ -def)
      with 1 that show thesis by blast
    next
      case 2
        then have  $\gamma'$ -le110:  $\gamma' \leq 1/10$ 

```

```

    using ⟨γ>1/10⟩ ⟨k>0⟩ maxm-bounds by (auto simp: γ-def γ'-def)
  have 1/10 - 1/k ≤ γ'
  proof -
    have §: l-m ≥ k/9 - 1
      using ⟨γ>1/10⟩ ⟨k>0⟩ 2 by (simp add: max-m-def γ-def) linarith
    have 1/10 - 1/k ≤ 1 - k / (10*k/9 - 1)
      using γ'-le110 ⟨m<l⟩ ⟨k>0⟩ by (simp add: γ'-def field-simps)
    also have ... ≤ 1 - k / (k + l - m)
      using ⟨l≤k⟩ ⟨m<l⟩ § by (simp add: divide-left-mono)
    also have ... = γ'
      using ⟨l>0⟩ ⟨l≤k⟩ ⟨m<l⟩ ⟨k>0⟩ by (simp add: γ'-def divide-simps)
    finally show 1/10 - 1 / real k ≤ γ' .
  qed
  with γ'-le110 that show thesis
  by linarith
qed
note γ'-cases = this
have 110: 1/10 - 1/k ≤ γ'
  using γ'-cases by (smt (verit, best) divide-nonneg-nonneg of-nat-0-le-iff)
have (real k)2 - 10 * real k ≤ (l-m) * (10 + 9*k)
  using 110 ⟨m<l⟩ ⟨k>0⟩
  by (simp add: γ'-def field-split-simps power2-eq-square)
with big ⟨k≥l⟩ have k/10 ≤ l-m
  unfolding Big101b-def Big-Closer-10-1-def by (smt (verit, best) mult-right-mono
of-nat-0-le-iff of-nat-mult)
then have k10-lm: nat ⌊k/10⌋ ≤ l - m
  by linarith
have lm-ge-25: nat ⌊2/5 * l⌋ ≤ l - m
  using False l4k k10-lm by linarith

— As with 9: a huge effort just to show that  $U$  is nontrivial. Proof actually
shows its cardinality exceeds a small multiple of  $l$  (7/5).
have l + Suc l - q ≤ (k+q choose q) / exp(δ*k)
  if nat ⌊k/10⌋ ≤ q q ≤ l for q
  using that
proof (induction q rule: nat-induct-at-least)
  case base
  have †: 0 < 10 + 10 * real-of-int ⌊k/10⌋ / k
  using ⟨k>0⟩ by (smt (verit) divide-nonneg-nonneg of-nat-0-le-iff of-nat-int-floor)
  have ln9: ln (10::real) ≥ 2
    by (approximation 5)
  have l + real (Suc l - nat ⌊k/10⌋) ≤ 2 + k/2
    using l4k by linarith
  also have ... ≤ exp(of-int ⌊k/10⌋ * 2 - k/200)
    using big by (simp add: Big101a-def Big-Closer-10-1-def ⟨l ≤ k⟩)
  also have ... ≤ exp(⌊k/10⌋ * ln(10) - k/200)
    by (intro exp-mono diff-mono mult-left-mono ln9) auto
  also have ... ≤ exp(⌊k/10⌋ * ln(10)) * exp (-real k/200)
    by (simp add: mult-exp-exp)

```

```

also have ... ≤ exp( $\lfloor k/10 \rfloor * \ln(10 + (10 * \text{nat}\lfloor k/10 \rfloor) / k)$ ) * exp (−real
k/200)
  using † by (intro mult-mono exp-mono) auto
  also have ... ≤  $(10 + (10 * \text{nat}\lfloor k/10 \rfloor) / k) ^ \text{nat}\lfloor k/10 \rfloor * \text{exp} (−\text{real}$ 
k/200)
  using † by (auto simp: powr-def simp flip: powr-realpow)
  also have ... ≤  $((k + \text{nat}\lfloor k/10 \rfloor) / (k/10)) ^ \text{nat}\lfloor k/10 \rfloor * \text{exp} (−\text{real}$ 
k/200)
  using <k>0 by (simp add: mult.commute add-divide-distrib)
  also have ... ≤  $((k + \text{nat}\lfloor k/10 \rfloor) / \text{nat}\lfloor k/10 \rfloor) ^ \text{nat}\lfloor k/10 \rfloor * \text{exp} (−\text{real}$ 
k/200)
  proof (intro mult-mono power-mono divide-left-mono)
    show  $\text{nat}\lfloor k/10 \rfloor \leq k/10$ 
    by linarith
  qed (use <k≥36> in auto)
  also have ... ≤  $(k + \text{nat}\lfloor k/10 \rfloor \text{gchoose } \text{nat}\lfloor k/10 \rfloor) * \text{exp} (−\text{real } k/200)$ 
  by (meson exp-gt-zero gbinomial-ge-n-over-k-pow-k le-add2 mult-le-cancel-right-pos
of-nat-mono)
  also have ... ≤  $(k + \text{nat}\lfloor k/10 \rfloor \text{choose } \text{nat}\lfloor k/10 \rfloor) * \text{exp} (−\text{real } k/200)$ 
  by (simp add: binomial-gbinomial)
  also have ... ≤  $(k + \text{nat}\lfloor k/10 \rfloor \text{choose } \text{nat}\lfloor k/10 \rfloor) / \text{exp} (\delta * k)$ 
  using  $\gamma < 0 < k$  by (simp add: algebra-simps  $\delta$ -def exp-minus' frac-le)
  finally show ?case by linarith
next
  case (Suc q)
  then show ?case
    apply simp
    by (smt (verit) divide-right-mono exp-ge-zero of-nat-0-le-iff)
  qed
from <m<l> this [of l−m]
have  $1 + l + \text{real } m \leq (k+l−m \text{choose } (l−m)) / \text{exp } \delta ^ k$ 
  by (simp add: exp-of-nat2-mult k10-lm)
also have ... ≤  $(k+l−m \text{choose } (l−m)) / \text{exp} (\delta * k)$ 
  by (simp add: exp-of-nat2-mult)
also have ... < U-lower-bound-ratio m * (real n)
proof −
  have  $\S: (k+l \text{choose } l) / \text{exp} (\delta * k) < n$ 
  by (simp add: less-eq-real-def nexp-gt pos-divide-less-eq)
  show ?thesis
    using mult-strict-left-mono [OF  $\S$ , of U-lower-bound-ratio m] kl-choose
prod-gt0
    by (auto simp: field-simps)
  qed
finally have U-MINUS-M:  $1+l < \text{real } n * \text{U-lower-bound-ratio } m - m$ 
  by argo
then have cardU-gt:  $\text{card } U > l + 1 \text{ card } U > 1$ 
  using cardU by linarith+

show False

```

```

using  $\gamma'$ -cases
proof cases
  case 1
  — Restricting attention to U
  define EU where  $EU \equiv E \cap Pow\ U$ 
  define RedU where  $RedU \equiv Red \cap Pow\ U$ 
  define BlueU where  $BlueU \equiv Blue \cap Pow\ U$ 
  have RedU-eq:  $RedU = EU \setminus BlueU$ 
    using BlueU-def Blue-def EU-def RedU-def Red-E by fastforce
  obtain [iff]: finite RedU finite BlueU  $RedU \subseteq EU$ 
    using BlueU-def EU-def RedU-def E-def V-def Red-E Blue-E fin-edges
  finite-subset by blast
  then have card-EU:  $card\ EU = card\ RedU + card\ BlueU$ 
  by (simp add: BlueU-def Blue-def Diff-Int-distrib2 EU-def RedU-def card-Diff-subset
  card-mono)
  then have card-RedU-le:  $card\ RedU \leq card\ EU$ 
    by linarith
  interpret UBB: Book-Basis U  $E \cap Pow\ U$  p0-min
  proof
    fix e assume  $e \in E \cap Pow\ U$ 
    with two-edges show  $e \subseteq U$   $card\ e = 2$  by auto
  next
    show finite U
    using  $\langle U \subseteq V \rangle$  by (simp add: V-def finite-subset)
    have  $x \in E$  if  $x \in all\ edges\ U$  for x
    using  $\langle U \subseteq V \rangle$  all-edges-mono that complete E-def by blast
    then show  $E \cap Pow\ U = all\ edges\ U$ 
    using comp-sgraph.wellformed  $\langle U \subseteq V \rangle$  by (auto intro: e-in-all-edges-ss)
  qed auto

  have BlueU-eq:  $BlueU = EU \setminus RedU$ 
  using Blue-eq complete by (fastforce simp: BlueU-def RedU-def EU-def V-def
  E-def)
  have [simp]:  $UBB.graph\ size = card\ EU$ 
    using EU-def by blast
  have  $card\ EU > 0$ 
    using  $\langle card\ U > 1 \rangle$  UBB.complete by (simp add: EU-def UBB.finV
  card-all-edges)

  have False if  $UBB.graph\ density\ BlueU > \gamma'$ 
  proof — — by maximality, etc.; only possible in case 1
    have Nx:  $Neighbours\ BlueU\ x \cap (U \setminus \{x\}) = Neighbours\ BlueU\ x$  for x
    using that by (auto simp: BlueU-eq EU-def Neighbours-def)
    have  $BlueU \subseteq E \cap Pow\ U$ 
    using BlueU-eq EU-def by blast
    with UBB.exists-density-edge-density [of 1 BlueU]
    obtain x where  $x \in U$  and x:  $UBB.graph\ density\ BlueU \leq UBB.gen\ density$ 
     $BlueU\ \{x\}\ (U \setminus \{x\})$ 
    by (metis UBB.complete  $\langle 1 < UBB.gorder \rangle$  card-1-singletonE insertI1

```

```

zero-less-one subsetD)
  with that have  $\gamma' \leq \text{UBB.gen-density BlueU } (U \setminus \{x\}) \{x\}$ 
    using UBB.gen-density-commute by auto
  then have *:  $\gamma' * (\text{card } U - 1) \leq \text{card } (\text{Neighbours BlueU } x)$ 
    using  $\langle \text{BlueU} \subseteq E \cap \text{Pow } U \rangle \langle \text{card } U > 1 \rangle \langle x \in U \rangle$ 
    by (simp add: UBB.gen-density-def UBB.edge-card-eq-sum-Neighbours
UBB.finV divide-simps Nx)

  have  $x: x \in V \setminus W$ 
    using  $\langle x \in U \rangle \langle U \subseteq V \rangle \langle \text{disjnt } U \ W \rangle$  by (auto simp: U-def disjnt-iff)
  moreover
  have is-good-clique  $n$  (insert x W)
    unfolding is-good-clique-def
  proof (intro conjI)
    show clique (insert x W) Blue
    proof (intro clique-insert)
      show clique  $W$  Blue
        using 53 is-good-clique-def by blast
      show all-edges-betw-un  $\{x\} \ W \subseteq \text{Blue}$ 
        using  $\langle x \in U \rangle$  by (auto simp: U-def all-edges-betw-un-def insert-commute
in-Neighbours-iff)
    qed (use  $\langle W \subseteq V \rangle \langle x \in V \setminus W \rangle$  in auto)
  next
  show insert x W  $\subseteq V$ 
    using  $\langle W \subseteq V \rangle \langle x \in V \setminus W \rangle$  by auto
  next
  have NB-Int-U: Neighbours Blue  $x \cap U = \text{Neighbours BlueU } x$ 
    using  $\langle x \in U \rangle$  by (auto simp: BlueU-def U-def Neighbours-def)
  have ulb-ins: U-lower-bound-ratio (card (insert x W)) = U-lower-bound-ratio
 $m * \gamma'$ 
    using  $\langle x \in V \setminus W \rangle \langle \text{finite } W \rangle$  by (simp add: m-def U-lower-bound-ratio-def
 $\gamma'$ -def)
  have  $n * \text{U-lower-bound-ratio } (\text{card } (\text{insert } x \ W)) = n * \text{U-lower-bound-ratio}$ 
 $m * \gamma'$ 
    by (simp add: ulb-ins)
  also have  $\dots \leq \text{real } (m + \text{card } U) * \gamma'$ 
    using mult-right-mono [OF cardU, of  $\gamma'$ ]  $\langle 0 < \gamma' \rangle$  by argo
  also have  $\dots \leq m + \text{card } U * \gamma'$ 
    using mult-left-mono [OF  $\langle \gamma' \leq 1 \rangle$ , of  $m$ ] by (simp add: algebra-simps)
  also have  $\dots \leq \text{Suc } m + \gamma' * (\text{UBB.gorder} - \text{Suc } 0)$ 
    using  $\langle x \in V \setminus W \rangle \langle \text{finite } W \rangle \langle 1 < \text{UBB.gorder} \rangle \langle \gamma' \leq 1 \rangle$ 
    by (simp add: U-lower-bound-ratio-def algebra-simps)
  also have  $\dots \leq \text{Suc } m + \text{card } (V \cap \bigcap (\text{Neighbours Blue } \langle \text{insert } x \ W \rangle))$ 
    using  $*$  NB-Int-U finV by (simp add: U-def Int-ac)
  also have  $\dots = \text{real } (\text{card } (\text{insert } x \ W) + \text{card } (V \cap \bigcap (\text{Neighbours Blue}$ 
 $\langle \text{insert } x \ W \rangle)))$ 
    using  $x$   $\langle \text{finite } W \rangle$  VUU by (auto simp: m-def U-def)
  finally show  $n * \text{U-lower-bound-ratio } (\text{card}(\text{insert } x \ W)) - \text{card}(\text{insert } x$ 
 $W)$ 

```

```

      ≤ card ( V ∩ ∩ ( Neighbours Blue ' insert x W ) )
    by simp
  qed
  ultimately show False
    using 1 clique-cases by blast
  qed
  then have *: UBB.graph-density BlueU ≤ γ' by force
  have no-RedU-K: ¬ (∃ K. UBB.size-clique k K RedU)
    unfolding UBB.size-clique-def RedU-def
  by (metis Int-subset-iff VUU all-edges-subset-iff-clique no-Red-K size-clique-def)
  have (∃ K. UBB.size-clique k K RedU) ∨ (∃ K. UBB.size-clique (l-m) K
BlueU)
  proof (rule ccontr)
    assume neg: ¬ ((∃ K. UBB.size-clique k K RedU) ∨ (∃ K. UBB.size-clique
(l-m) K BlueU))
    interpret UBB-NC: No-Cliques U E ∩ Pow U p0-min RedU BlueU l-m k
  proof
    show BlueU = E ∩ Pow U \ RedU
      using BlueU-eq EU-def by fastforce
    qed (use neg EU-def ⟨RedU ⊆ EU⟩ no-RedU-K ⟨l≤k⟩ in auto)
    show False
  proof (intro UBB-NC.Closer-10-2)
    have δ ≤ 1/200
      using γ by (simp add: δ-def field-simps)
    then have exp (δ * real k) ≤ exp (real k/200)
      using ⟨0 < k⟩ by auto
    then have expexp: exp (δ*k) * exp (- real k/200) ≤ 1
    by (metis divide-minus-left exp-ge-zero exp-minus-inverse mult-right-mono)
    have exp (- real k/200) * (k + (l-m) choose (l-m)) = exp (- real
k/200) * U-lower-bound-ratio m * (k+l choose l)
      using ⟨m < l⟩ kl-choose by force
    also have ... < (n/2) * exp (δ*k) * exp (- real k/200) * U-lower-bound-ratio
m
      using n2exp-gt prod-gt0 by auto
    also have ... ≤ (n/2) * U-lower-bound-ratio m
      using mult-left-mono [OF expexp, of (n/2) * U-lower-bound-ratio m]
prod-gt0 by (simp add: mult-ac)
    also have ... ≤ n * U-lower-bound-ratio m - m — formerly stuck here,
due to the "minus m"
      using U-MINUS-M ⟨m < l⟩ by auto
    finally have exp (- real k/200) * (k + (l-m) choose (l-m)) ≤ UBB.nV
      using cardU by linarith
    then show exp (- real k / 200) * (k + (l-m) choose (l-m)) ≤ UBB.nV
      using ⟨m < l⟩ by (simp add: γ'-def)
  next
    have 1 - γ' ≤ UBB.graph-density RedU
      using * card-EU ⟨card EU > 0⟩
      by (simp add: UBB.graph-density-def BlueU-eq field-split-simps split:
if-split-asm)

```

```

    then show  $1 - \text{real } (l-m) / (\text{real } k + \text{real } (l-m)) \leq \text{UBB.graph-density}$ 
  RedU
    unfolding  $\gamma'$ -def using  $\langle m < l \rangle$  by (smt (verit, ccfv-threshold) less-imp-le-nat
of-nat-add of-nat-diff)
    next
      show  $p0\text{-min} \leq 1 - \text{real } (l-m) / (\text{real } k + \text{real } (l-m))$ 
        using  $p0\text{-min-101 } \langle \gamma' \leq \gamma \rangle \langle m < l \rangle \gamma$ 
        by (smt (verit, del-insts) of-nat-add  $\gamma'$ -def less-imp-le-nat of-nat-diff)
      next
        have Big-10-2I:  $\bigwedge l' \mu. \llbracket \text{nat } \lfloor 2/5 * l \rfloor \leq l'; 1/10 \leq \mu; \mu \leq 1 / 5 \rrbracket \implies$ 
Big-Closer-10-2  $\mu l'$ 
          using big by (meson Big101d-def Big-Closer-10-1-def order.refl)
          have  $m \leq \text{real } l * (1 - (10/11)*\gamma)$ 
            using  $\langle m < l \rangle \langle \gamma > 1/10 \rangle \langle \gamma' \geq 1/10 \rangle \gamma$ 
            apply (simp add:  $\gamma$ -def  $\gamma'$ -def field-simps)
            by (smt (verit, ccfv-SIG) mult.commute mult-left-mono distrib-left)
          then have  $\text{real } l - \text{real } m \geq (10/11) * \gamma * l$ 
            by (simp add: algebra-simps)
          moreover
            have  $1/10 \leq \gamma' \wedge \gamma' \leq 1/5$ 
              using mult-mono [OF  $\gamma \gamma$ ]  $\langle \gamma' \geq 1/10 \rangle \langle \gamma' \leq \gamma \rangle \gamma$  by (auto simp:
power2-eq-square)
          ultimately
            have Big-Closer-10-2  $\gamma' (l-m)$ 
              using  $lm\text{-ge-25}$  by (intro Big-10-2I) auto
            then show Big-Closer-10-2  $((l-m) / (\text{real } k + \text{real } (l-m))) (l-m)$ 
              by (simp add:  $\gamma'$ -def  $\langle m < l \rangle$  add-diff-eq less-or-eq-imp-le)
          next
            show  $l-m \leq k$ 
              using  $\langle l \leq k \rangle$  by auto
            show  $(l-m) / (\text{real } k + \text{real } (l-m)) \leq 1/5$ 
              using  $\gamma \gamma$ -def  $\langle m < l \rangle$  by fastforce
            show  $1/10 \leq (l-m) / (\text{real } k + \text{real } (l-m))$ 
              using  $\gamma'$ -def  $\langle 1/10 \leq \gamma' \rangle \langle m < l \rangle$  by auto
          qed
        qed
      with no-RedU-K UBB.size-clique-def obtain K where  $K \subseteq U$  UBB.size-clique
(l-m) K BlueU
        by meson
      then show False
        using no-Blue-K extend-Blue-clique VUU
        unfolding UBB.size-clique-def size-clique-def BlueU-def
        by (metis Int-subset-iff all-edges-subset-iff-clique)
    next
      case 2
      have RN  $k (l-m) \leq \exp(-((l-m) / (k + \text{real } (l-m))) / 20) * k + 1) * (k$ 
+  $(l-m)$  choose  $(l-m)$ )
        proof (intro Far-9-1 strip)
          show  $\text{real } (l-m) / (\text{real } k + \text{real } (l-m)) \leq 1/10$ 

```

```

    using  $\gamma'$ -def 2  $\langle m < l \rangle$  by auto
next — here is where we need the specified definition of  $\gamma_0$ 
show Big-Far-9-1 (real (l-m) / (k + real (l-m))) (l-m)
proof (intro Big91-I [OF lm-ge-25])
  have  $0.07 \leq (1::real)/10 - 1/36$ 
  by (approximation 5)
  also have ...  $\leq 1/10 - 1/k$ 
  using  $\langle k \geq 36 \rangle$  by (intro diff-mono divide-right-mono) auto
  finally have  $\gamma: \gamma' \geq 0.07$  using 110 by linarith
  with  $\langle m < l \rangle$  show  $\gamma_0 \leq \text{real } (l-m) / (\text{real } k + \text{real } (l-m))$ 
  by (simp add:  $\gamma_0$ -def min-le-iff-disj  $\gamma'$ -def algebra-simps)
next
  show  $\text{real } (l-m) / (\text{real } k + \text{real } (l-m)) \leq 1/10$ 
  using 2  $\langle m < l \rangle$  by (simp add:  $\gamma'$ -def)
qed
next
  show  $p_0\text{-min} \leq 1 - 1/10 * (1 + 1 / 15)$ 
  using  $p_0\text{-min-101}$  by auto
qed
also have ...  $\leq \text{real } n * U\text{-lower-bound-ratio } m - m$ 
proof —
  have  $\gamma * \text{real } k \leq k/5$ 
  using  $\gamma \langle 0 < k \rangle$  by auto
  also have ...  $\leq \gamma' * (\text{real } k * 2) + 2$ 
  using mult-left-mono [OF 110, of k*2]  $\langle k > 0 \rangle$  by (simp add: algebra-simps)
  finally have  $\gamma * \text{real } k \leq \gamma' * (\text{real } k * 2) + 2$  .
  then have expexp:  $\exp (\delta * \text{real } k) * \exp (-\gamma' * k / 20 - 1) \leq 1$ 
  by (simp add:  $\delta$ -def flip: exp-add)
  have  $\exp (-\gamma' * k / 20 + 1) * (k + (l-m) \text{ choose } (l-m)) = \exp (-\gamma' * k / 20 + 1)$ 
* U-lower-bound-ratio m * (k+l choose l)
  using  $\langle m < l \rangle$  kl-choose by force
  also have ...  $< (n/2) * \exp (\delta * k) * \exp (-\gamma' * k / 20 - 1) * U\text{-lower-bound-ratio}$ 
m
  using  $n2\text{exp-gt}' \text{ prod-gt0}$  by (simp add: exp2 exp-diff exp-minus' mult-ac
pos-less-divide-eq)
  also have ...  $\leq (n/2) * U\text{-lower-bound-ratio } m$ 
  using expexp order-le-less prod-gt0 by fastforce
  also have ...  $\leq n * U\text{-lower-bound-ratio } m - m$ 
  using U-MINUS-M  $\langle m < l \rangle$  by fastforce
  finally show ?thesis
  using  $\langle m < l \rangle$  by (simp add:  $\gamma'$ -def) argo
qed
also have ...  $\leq \text{card } U$ 
  using cardU by auto
  finally have RN k (l-m)  $\leq \text{card } U$  by linarith
  then show False
  using Red-Blue-RN  $\langle U \subseteq V \rangle$  extend-Blue-clique no-Blue-K no-Red-K by
blast
qed

```

qed
qed

definition *ok-fun-10-1* $\equiv \lambda \gamma k$. if *Big-Closer-10-1* ($\min \gamma 0.07$) ($\text{nat}[(\gamma / (1-\gamma)) * k]$) then 3 else ($\gamma/40 * k$)

lemma *ok-fun-10-1*:

assumes $0 < \gamma$ $\gamma < 1$

shows *ok-fun-10-1* $\gamma \in o(\text{real})$

proof –

define γ_0 **where** $\gamma_0 \equiv \min \gamma 0.07$

have $\gamma_0 > 0$

using *assms* **by** (*simp add: γ_0 -def*)

then have $\forall^\infty l$. *Big-Closer-10-1* $\gamma_0 l$

by (*simp add: Big-Closer-10-1*)

then obtain l **where** $\bigwedge l'. l' \geq l \implies \text{Big-Closer-10-1 } \gamma_0 l'$

using *eventually-sequentially* **by** *auto*

moreover

have $\text{nat}[(\gamma / (1-\gamma)) * k] \geq l$ **if** *real* $k \geq l/\gamma - l$ **for** k

using *that assms*

by (*auto simp: field-simps intro!: le-natceiling-iff*)

ultimately have $\forall^\infty k$. *Big-Closer-10-1* ($\min \gamma 0.07$) ($\text{nat}[(\gamma / (1-\gamma)) * k]$)

by (*smt (verit) γ_0 -def eventually-sequentially nat-ceiling-le-eq*)

then have $\forall^\infty k$. *ok-fun-10-1* $\gamma k = 3$

by (*simp add: ok-fun-10-1-def eventually-mono*)

then show *?thesis*

by (*simp add: const-smallo-real landau-o.small.in-cong*)

qed

theorem *Closer-10-1-unconditional*:

fixes $l k :: \text{nat}$

fixes $\delta \gamma :: \text{real}$

defines $\gamma \equiv \text{real } l / (\text{real } k + \text{real } l)$

defines $\delta \equiv \gamma/40$

assumes $\gamma: 0 < \gamma$ $\gamma \leq 1/5$

assumes *p0-min-101*: $p0\text{-min} \leq 1 - 1/5$

shows *RN* $k l \leq \text{exp } (-\delta * k + \text{ok-fun-10-1 } \gamma k) * (k+l \text{ choose } l)$

proof –

define γ_0 **where** $\gamma_0 \equiv \min \gamma 0.07$

show *?thesis*

proof (*cases Big-Closer-10-1* $\gamma_0 l$)

case *True*

show *?thesis*

using *Closer-10-1* [*OF True* [*unfolded γ_0 -def γ -def*]] *assms*

by (*simp add: ok-fun-10-1-def γ -def δ -def RN-le-choose'*)

next

case *False*

have ($\text{nat } [\gamma * k / (1-\gamma)] \leq l$

by (*simp add: γ -def divide-simps*)

```

with False Big-Closer-10-1-upward
have  $\neg$  Big-Closer-10-1  $\gamma 0$  (nat  $\lceil \gamma * k / (1 - \gamma) \rceil$ )
  by blast
then show ?thesis
  by (simp add: ok-fun-10-1-def  $\delta$ -def  $\gamma 0$ -def RN-le-choose')
qed
qed

end

end

```

10 From diagonal to off-diagonal

```

theory From-Diagonal
  imports Closer-To-Diagonal

```

```

begin

```

10.1 Lemma 11.2

```

definition ok-fun-11-2a  $\equiv \lambda k. \lceil \text{real } k \text{ powr } (3/4) \rceil * \log 2 k$ 

```

```

definition ok-fun-11-2b  $\equiv \lambda \mu k. k \text{ powr } (39/40) * (\log 2 \mu + 3 * \log 2 k)$ 

```

```

definition ok-fun-11-2c  $\equiv \lambda \mu k. - k * \log 2 (1 - (2 / (1 - \mu))) * k \text{ powr } (-1/40)$ 

```

```

definition ok-fun-11-2  $\equiv \lambda \mu k. 2 - \text{ok-fun-71 } \mu k + \text{ok-fun-11-2a } k$ 
   $+ \max (\text{ok-fun-11-2b } \mu k) (\text{ok-fun-11-2c } \mu k)$ 

```

```

lemma ok-fun-11-2a: ok-fun-11-2a  $\in o(\text{real})$ 

```

```

  unfolding ok-fun-11-2a-def

```

```

  by real-asymp

```

possibly, the functions that depend upon μ need a more refined analysis to cover a closed interval of possible values. But possibly not, as the text implies $\mu = (2::'a) / (5::'a)$.

```

lemma ok-fun-11-2b: ok-fun-11-2b  $\mu \in o(\text{real})$ 

```

```

  unfolding ok-fun-11-2b-def by real-asymp

```

```

lemma ok-fun-11-2c: ok-fun-11-2c  $\mu \in o(\text{real})$ 

```

```

unfolding ok-fun-11-2c-def

```

```

  by real-asymp

```

```

lemma ok-fun-11-2:

```

```

  assumes  $0 < \mu$   $\mu < 1$ 

```

```

  shows ok-fun-11-2  $\mu \in o(\text{real})$ 

```

```

  unfolding ok-fun-11-2-def

```

by (*simp add: assms const-smallo-real maxmin-in-smallo ok-fun-11-2a ok-fun-11-2b ok-fun-11-2c ok-fun-71 sum-in-smallo*)

definition *Big-From-11-2* \equiv

$\lambda \mu k. \text{Big-ZZ-8-6 } \mu k \wedge \text{Big-X-7-1 } \mu k \wedge \text{Big-Y-6-2 } \mu k \wedge \text{Big-Red-5-3 } \mu k \wedge$
 $\text{Big-Blue-4-1 } \mu k$
 $\wedge 1 \leq \mu^2 * \text{real } k \wedge 2 / (1-\mu) * \text{real } k \text{ powr } (-1/40) < 1 \wedge 1/k < 1/2$
 $- 3 * \text{eps } k$

lemma *Big-From-11-2*:

assumes $0 < \mu 0 \ \mu 0 \leq \mu 1 \ \mu 1 < 1$

shows $\forall^\infty k. \forall \mu. \mu \in \{\mu 0.. \mu 1\} \longrightarrow \text{Big-From-11-2 } \mu k$

proof –

have $A: \forall^\infty k. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 1 \leq \mu^2 * k$

proof (*intro eventually-all-geI0*)

show $*$: $\forall^\infty x. 1 \leq \mu 0^2 * \text{real } x$

using $\langle 0 < \mu 0 \rangle$ **by** *real-asymp*

next

fix $k \ \mu$

assume $1 \leq \mu 0^2 * \text{real } k$ **and** $\mu 0 \leq \mu \ \mu \leq \mu 1$

with $\langle 0 < \mu 0 \rangle$ **show** $1 \leq \mu^2 * k$

by (*smt (verit, ccfv-SIG) mult-le-cancel-right of-nat-less-0-iff power-mono*)

qed

have $B: \forall^\infty k. \forall \mu. \mu 0 \leq \mu \wedge \mu \leq \mu 1 \longrightarrow 2 / (1-\mu) * k \text{ powr } (-1/40) < 1$

proof (*intro eventually-all-geI1*)

show $\forall^\infty k. 2 / (1-\mu 1) * k \text{ powr } (-1/40) < 1$

by *real-asymp*

qed (*use assms in auto*)

have $C: \forall^\infty k. 1/k < 1/2 - 3 * \text{eps } k$

unfolding *eps-def* **by** *real-asymp*

show *?thesis*

unfolding *Big-From-11-2-def*

using *assms Big-ZZ-8-6 Big-X-7-1 Big-Y-6-2 Big-Red-5-3 Big-Blue-4-1 A B C*

by (*simp add: eventually-conj-iff all-imp-conj-distrib*)

qed

Simply to prevent issues about the positioning of the function *real*

abbreviation *ratio* $\equiv \lambda \mu s t. \mu * (\text{real } s + \text{real } t) / \text{real } s$

the text refers to the actual Ramsey number but I don't see how that could work. Theorem 11.1 will define n to be one less than the Ramsey number, hence we add that one back here.

lemma (*in Book*) *From-11-2*:

assumes $l=k$

assumes *big*: *Big-From-11-2* μk

defines $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ **and** $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$

defines $t \equiv \text{card } \mathcal{R}$ **and** $s \equiv \text{card } \mathcal{S}$

defines $nV' \equiv \text{Suc } nV$

assumes 0 : $\text{card } X0 \geq nV \text{ div } 2$ **and** $p0 \geq 1/2$
shows $\log 2 nV' \leq k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + s * \log 2 (\text{ratio } \mu s t) + \text{ok-fun-11-2 } \mu k$
proof –
have big71 : $\text{Big-X-7-1 } \mu k$ **and** big62 : $\text{Big-Y-6-2 } \mu k$ **and** big86 : $\text{Big-ZZ-8-6 } \mu k$ **and** big53 : $\text{Big-Red-5-3 } \mu k$
and big41 : $\text{Big-Blue-4-1 } \mu k$ **and** $\text{big}\mu$: $1 \leq \mu^2 * \text{real } k$
and big-le1 : $2 / (1-\mu) * \text{real } k \text{ powr } (-1/40) < 1$
using big by ($\text{auto simp: Big-From-11-2-def}$)
have $\text{big}\mu1$: $1 \leq \mu * \text{real } k$
using $\text{big}\mu \mu01$
by ($\text{smt (verit, best) mult-less-cancel-right2 mult-right-mono of-nat-less-0-iff power2-eq-square}$)
then have $\log2\mu k$: $\log 2 \mu + \log 2 k \geq 0$
using $\text{kn0 } \mu01 \text{ add-log-eq-powr by auto}$
have $\text{big}\mu2$: $1 \leq \mu * (\text{real } k)^2$
unfolding $\text{power2-eq-square by (smt (verit, ccfv-SIG) big}\mu1 \mu01 \text{ mult-less-cancel-left1 mult-mono')}$
define g **where** $g \equiv \lambda k. \lceil \text{real } k \text{ powr } (3/4) \rceil * \log 2 k$
have g : $g \in o(\text{real})$
unfolding $g\text{-def by real-asymp}$
have bb-gt0 : $\text{bigbeta} > 0$
using $\text{big53 bigbeta-gt0 } \langle l=k \rangle$ **by blast**
have $t < k$
by ($\text{simp add: } \mathcal{R}\text{-def } t\text{-def red-step-limit}$)
have $s < k$
unfolding $\mathcal{S}\text{-def } s\text{-def}$
using $\text{bbblue-dboost-step-limit big41 } \langle l=k \rangle$ **by fastforce**

have $k34$: $k \text{ powr } (3/4) \leq k \text{ powr } 1$
using $\text{kn0 by (intro powr-mono) auto}$

define $g712$ **where** $g712 \equiv \lambda k. 2 - \text{ok-fun-71 } \mu k + g k$
have $nV' \geq 2$
using $\text{gorder-ge2 } nV'\text{-def by linarith}$
have $nV' \leq 4 * \text{card } X0$
using $0 \text{ card-XY0 by (auto simp: } nV'\text{-def odd-iff-mod-2-eq-one)}$
with $\mu01$ **have** $2 \text{ powr } (\text{ok-fun-71 } \mu k - 2) * \mu^k * (1-\mu)^t * (\text{bigbeta} / \mu)^s * nV'$
 $\leq 2 \text{ powr } \text{ok-fun-71 } \mu k * \mu^k * (1-\mu)^t * (\text{bigbeta} / \mu)^s * \text{card } X0$
using $\mu01$ **by** ($\text{simp add: powr-diff mult.assoc bigbeta-ge0 mult-left-mono}$)
also have $\dots \leq \text{card } (X\text{seq halted-point})$
using $X\text{-7-1 assms big71 by blast}$
also have $\dots \leq 2 \text{ powr } (g k)$
proof –
have $1/k < p0 - 3 * \varepsilon$
using $\text{big } \langle p0 \geq 1/2 \rangle$ **by** ($\text{auto simp: Big-From-11-2-def}$)
also have $\dots \leq \text{pseq halted-point}$
using $Y\text{-6-2-halted big62 assms by blast}$

finally have $pseq\ halted_point > 1/k$.
moreover have $termination_condition (Xseq\ halted_point) (Yseq\ halted_point)$
using $halted_point_halted\ step_terminating_iff$ **by** $blast$
ultimately have $card (Xseq\ halted_point) \leq RN\ k\ (nat\ \lceil\ real\ k\ powr\ (3/4)\rceil)$
using $\langle l=k \rangle\ pseq_def\ termination_condition_def$ **by** $auto$
then show $?thesis$
unfolding g_def **by** $(smt\ (verit)\ RN34_le_2powr_ok\ kn0\ of_nat_le_iff)$
qed
finally have $58: 2\ powr\ (g\ k) \geq 2\ powr\ (ok_fun_71\ \mu\ k - 2) * \mu^k * (1-\mu)^t * (bigbeta / \mu)^s * nV'$.
then have $59: nV' \leq 2\ powr\ (g712\ k) * (1/\mu)^k * (1 / (1-\mu))^t * (\mu / bigbeta)^s$
using $\mu01\ bb_gt0$ **by** $(simp\ add: g712_def\ powr_diff\ powr_add\ mult.commute\ divide_simps)$ $argo$

define a **where** $a \equiv 2 / (1-\mu)$
have $ok_less1: a * real\ k\ powr\ (-1/40) < 1$
unfolding a_def **using** big_le1 **by** $blast$
consider $s < k\ powr\ (39/40) \mid s \geq k\ powr\ (39/40)\ bigbeta \geq (1 - a * k\ powr\ (-1/40)) * (s / (s + t))$
using $ZZ-8-6\ big86\ a_def\ \langle l=k \rangle$ **by** $(force\ simp: s_def\ t_def\ \mathcal{S}_def\ \mathcal{R}_def)$
then show $?thesis$
proof $cases$
case 1
define h **where** $h \equiv \lambda c\ k. real\ k\ powr\ (39/40) * (\log\ 2\ \mu + real\ c * \log\ 2\ (real\ k))$
have $h: h\ c \in o(real)$ **for** c
unfolding h_def **by** $real_asympt$

have $le_h: |s * \log\ 2\ (ratio\ \mu\ s\ t)| \leq h\ 1\ k$
proof $(cases\ s > 0)$
case $True$
with $\langle s > 0 \rangle$ **have** $\mu eq: ratio\ \mu\ s\ t = \mu * (1 + t/s)$
by $(auto\ simp: distrib_left\ add_divide_distrib)$
show $?thesis$
proof $(cases\ \log\ 2\ (ratio\ \mu\ s\ t) \leq 0)$
case $True$
have $s * (-\log\ 2\ (\mu * (1 + t/s))) \leq real\ k\ powr\ (39/40) * (\log\ 2\ \mu + \log\ 2\ (real\ k))$
proof $(intro\ mult_mono)$
show $s \leq k\ powr\ (39 / 40)$
using 1 **by** $linarith$
next
have $inverse\ (\mu * (1 + t/s)) \leq inverse\ \mu$
using $\mu01\ inverse_le_1_iff$ **by** $fastforce$
also have $\dots \leq \mu * k$
using $big\mu\ \mu01$ **by** $(metis\ neq_iff\ mult.assoc\ mult_le_cancel_left_pos\ power2_eq_square\ right_inverse)$
finally have $inverse\ (\mu * (1 + t/s)) \leq \mu * k$.

```

moreover have  $0 < \mu * (1 + \text{real } t / \text{real } s)$ 
  using  $\mu 01 <0 < s>$  by (simp add: zero-less-mult-iff add-num-frac)
ultimately have  $-\log 2 (\mu * (1 + \text{real } t / \text{real } s)) \leq \log 2 (\mu * k)$ 
  using  $\mu 01 kn0$  by (simp add: zero-less-mult-iff flip: log-inverse log-mult)
then show  $-\log 2 (\mu * (1 + \text{real } t / \text{real } s)) \leq \log 2 \mu + \log 2 (\text{real } k)$ 
  using  $\langle \mu > 0 \rangle kn0$  log-mult by fastforce
qed (use True  $\mu eq$  in auto)
with  $\langle s > 0 \rangle big\mu 1$  True show ?thesis
  by (simp add:  $\mu eq$  h-def mult-le-0-iff)
next
case False
have lek:  $1 + t/s \leq k$ 
proof -
  have  $\text{real } t \leq \text{real } t * \text{real } s$ 
    using True mult-le-cancel-left1 by fastforce
  then have  $1 + t/s \leq 1 + t$ 
    by (simp add: True pos-divide-le-eq)
  also have  $\dots \leq k$ 
    using  $\langle t < k \rangle$  by linarith
  finally show ?thesis .
qed
have  $|s * \log 2 (\text{ratio } \mu s t)| \leq k \text{ powr } (39/40) * \log 2 (\text{ratio } \mu s t)$ 
  using False 1 by auto
also have  $\dots = k \text{ powr } (39/40) * (\log 2 (\mu * (1 + t/s)))$ 
  by (simp add:  $\mu eq$ )
also have  $\dots = k \text{ powr } (39/40) * (\log 2 \mu + \log 2 (1 + t/s))$ 
using  $\mu 01$  by (smt (verit, best) divide-nonneg-nonneg log-mult of-nat-0-le-iff)

also have  $\dots \leq k \text{ powr } (39/40) * (\log 2 \mu + \log 2 k)$ 
  by (smt (verit, best) 1 Transcendental.log-mono divide-nonneg-nonneg lek
    mult-le-cancel-left-pos of-nat-0-le-iff)
also have  $\dots \leq h 1 k$ 
  unfolding h-def using kn0 by force
finally show ?thesis .
qed
qed (use log2 $\mu k$  h-def in auto)

have  $\beta$ : bigbeta  $\geq 1 / (\text{real } k)^2$ 
  using big53 bigbeta-ge-square <l=k> by blast
then have  $(\mu / \text{bigbeta}) ^ s \leq (\mu * (\text{real } k)^2) ^ s$ 
  using bb-gt0 kn0  $\mu 01$  by (intro power-mono) (auto simp: divide-simps
mult commute)
also have  $\dots \leq (\mu * (\text{real } k)^2) \text{ powr } (k \text{ powr } (39/40))$ 
using  $\mu 01 big\mu 2 1$  by (smt (verit) powr-less-mono powr-one-eq-one powr-realpow)
also have  $\dots = 2 \text{ powr } (\log 2 ((\mu * (\text{real } k)^2) \text{ powr } (k \text{ powr } (39/40))))$ 
  by (smt (verit, best) big $\mu 2$  powr-gt-zero powr-log-cancel)
also have  $\dots = 2 \text{ powr } h 2 k$ 
  using  $\mu 01 big\mu 2 kn0$  by (simp add: log-powr log-nat-power log-mult h-def)
finally have  $\dagger$ :  $(\mu / \text{bigbeta}) ^ s \leq 2 \text{ powr } h 2 k$  .

```

```

have ‡:  $nV' \leq 2 \text{ powr } (g712 k) * (1/\mu) ^ k * (1 / (1-\mu)) ^ t * 2 \text{ powr } h 2 k$ 
  using 59 mult-left-mono [OF ‡, of 2 powr (g712 k) * (1/\mu) ^ k * (1 / (1-\mu))
^ t]
  by (smt (verit)  $\mu 01$  pos-prod-le powr-nonneg-iff zero-less-divide-iff zero-less-power)
have *:  $\log 2 nV' \leq k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + (g712 k + h$ 
2 k)
  using  $\mu 01$   $\langle nV' \geq 2 \rangle$  by (simp add: log-mult log-nat-power order.trans [OF
Transcendental.log-mono [OF - - ‡]])

show ?thesis
proof -
  have le-ok-fun:  $g712 k + h 3 k \leq \text{ok-fun-11-2 } \mu k$ 
  by (simp add: g712-def h-def ok-fun-11-2-def g-def ok-fun-11-2a-def ok-fun-11-2b-def)
  have h3:  $h 3 k = h 1 k + h 2 k - \text{real } k \text{ powr } (39/40) * \log 2 \mu$ 
  by (simp add: h-def algebra-simps)
  have 0 ≤ h 1 k + s * log 2 (( $\mu * \text{real } s + \mu * \text{real } t$ ) / s)
  by (smt (verit, del-insts) of-nat-add distrib-left le-h)
  moreover have log 2  $\mu < 0$ 
  using  $\mu 01$  by simp
  ultimately have  $g712 k + h 2 k \leq s * \log 2 (\text{ratio } \mu s t) + \text{ok-fun-11-2 } \mu k$ 
  by (smt (verit, best) kn0 distrib-left h3 le-ok-fun nat-neq-iff of-nat-eq-0-iff
pos-prod-lt powr-gt-zero)
  then show  $\log 2 nV' \leq k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + s * \log 2$ 
(ratio  $\mu s t$ ) + ok-fun-11-2  $\mu k$ 
  using * by linarith
qed
next
case 2
then have s > 0
  using kn0 powr-gt-zero by fastforce
define h where  $h \equiv \lambda k. \text{real } k * \log 2 (1 - a * k \text{ powr } (-1/40))$ 
have s * log 2 ( $\mu / \text{bigbeta}$ ) = s * log 2  $\mu - s * \log 2 (\text{bigbeta})$ 
  using  $\mu 01$  bb-gt0 2 by (simp add: log-divide algebra-simps)
also have ... ≤ s * log 2  $\mu - s * \log 2 ((1 - a * k \text{ powr } (-1/40)) * (s / (s$ 
+ t)))
  using 2  $\langle s > 0 \rangle$  ok-less1 by (intro diff-mono order-refl mult-left-mono Tran-
scendental.log-mono) auto
  also have ... = s * log 2  $\mu - s * (\log 2 (1 - a * k \text{ powr } (-1/40)) + \log 2$ 
(s / (s + t)))
  using  $\langle 0 < s \rangle$  a-def add-log-eq-powr big-le1 by auto
  also have ... = s * log 2 (ratio  $\mu s t$ ) - s * log 2 (1 - a * k powr (-1/40))
  using  $\langle 0 < \mu \rangle \langle 0 < s \rangle$  minus-log-eq-powr by (auto simp flip: right-diff-distrib')
  also have ... < s * log 2 (ratio  $\mu s t$ ) - h k
proof -
  have log 2 (1 - a * real k powr (-1/40)) < 0
  using  $\mu 01$  kn0 a-def ok-less1 by auto
  with  $\langle s < k \rangle$  show ?thesis
  by (simp add: h-def)
qed

```

```

finally have †:  $s * \log 2 (\mu / \text{bigbeta}) < s * \log 2 (\text{ratio } \mu s t) - h k .$ 
show ?thesis
proof –
  have le-ok-fun:  $g712 k - h k \leq \text{ok-fun-11-2 } \mu k$ 
  by (simp add: g712-def h-def ok-fun-11-2-def g-def ok-fun-11-2a-def a-def ok-fun-11-2c-def)
  have  $\log 2 nV' \leq s * \log 2 (\mu / \text{bigbeta}) + k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + (g712 k)$ 
  proof (intro order.trans [OF Transcendental.log-mono [OF - - 59]])
  show  $\log 2 (2 \text{ powr } g712 k * (1/\mu) ^ k * (1 / (1-\mu)) ^ t * (\mu / \text{bigbeta}) ^ s)$ 
     $\leq s * \log 2 (\mu / \text{bigbeta}) + k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + g712 k$ 
  using bb-gt0  $\mu 01$  by (simp add: log-mult log-nat-power)
  qed (use  $\langle nV' \geq 2 \rangle$  in auto)
  with † le-ok-fun show  $\log 2 nV' \leq k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) + s * \log 2 (\text{ratio } \mu s t) + \text{ok-fun-11-2 } \mu k$ 
  by simp
qed
qed
qed

```

10.2 Lemma 11.3

same remark as in Lemma 11.2 about the use of the Ramsey number in the conclusion

lemma (*in Book*) *From-11-3*:

```

assumes  $l=k$ 
assumes big: Big-Y-6-1  $\mu k$ 
defines  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$  and  $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$ 
defines  $t \equiv \text{card } \mathcal{R}$  and  $s \equiv \text{card } \mathcal{S}$ 
defines  $nV' \equiv \text{Suc } nV$ 
assumes  $0$ :  $\text{card } Y0 \geq nV \text{ div } 2$  and  $p0 \geq 1/2$ 
shows  $\log 2 nV' \leq \log 2 (\text{RN } k (k-t)) + s + t + 2 - \text{ok-fun-61 } k$ 
proof –
  define RS where  $RS \equiv \text{Step-class } \{\text{red-step}, \text{dboost-step}\}$ 
  have  $RS = \mathcal{R} \cup \mathcal{S}$ 
  using Step-class-insert R-def S-def RS-def by blast
  moreover obtain finite  $\mathcal{R}$  finite  $\mathcal{S}$ 
  by (simp add: R-def S-def)
  moreover have disjnt  $\mathcal{R} \mathcal{S}$ 
  using R-def S-def disjnt-Step-class by auto
  ultimately have card-RS:  $\text{card } RS = t+s$ 
  by (simp add: t-def s-def card-Un-disjnt)
  have  $4$ :  $nV'/4 \leq \text{card } Y0$ 
  using  $0$  card-XY0 by (auto simp: nV'-def odd-iff-mod-2-eq-one)
  have ge0:  $0 \leq 2 \text{ powr } \text{ok-fun-61 } k * p0 ^ \text{card } RS$ 
  using p0-01 by fastforce
  have  $nV' \geq 2$ 

```

using *gorder-ge2 nV'-def* **by** *linarith*
have $2 \text{ powr } (- \text{ real } s - \text{ real } t + \text{ ok-fun-61 } k - 2) * nV' = 2 \text{ powr } (\text{ ok-fun-61 } k - 2) * (1/2) ^ \text{ card } RS * nV'$
by (*simp add: powr-add powr-diff powr-minus power-add powr-realpow divide-simps card-RS*)
also have $\dots \leq 2 \text{ powr } (\text{ ok-fun-61 } k - 2) * p0 ^ \text{ card } RS * nV'$
using *power-mono [OF <p0 ≥ 1/2>] <nV' ≥ 2>* **by** *auto*
also have $\dots \leq 2 \text{ powr } (\text{ ok-fun-61 } k) * p0 ^ \text{ card } RS * (nV'/4)$
by (*simp add: divide-simps powr-diff split: if-split-asm*)
also have $\dots \leq 2 \text{ powr } (\text{ ok-fun-61 } k) * p0 ^ \text{ card } RS * \text{ card } Y0$
using *mult-left-mono [OF 4 ge0]* **by** *simp*
also have $\dots \leq \text{ card } (Yseq \text{ halted-point})$
using *Y-6-1 big <l=k>* **by** (*auto simp: RS-def divide-simps split: if-split-asm*)
finally have $2 \text{ powr } (- \text{ real } s - \text{ real } t + \text{ ok-fun-61 } k - 2) * nV' \leq \text{ card } (Yseq \text{ halted-point})$.
moreover
{ assume $\text{ card } (Yseq \text{ halted-point}) \geq RN \ k \ (k-t)$
then obtain K **where** $K: K \subseteq Yseq \text{ halted-point}$ **and** *size-clique (k-t) K Red*
 \vee *size-clique k K Blue*
by (*metis RN-commute Red-Blue-RN Yseq-subset-V*)
then have $KRed: \text{ size-clique } (k-t) \ K \ Red$
using $\langle l=k \rangle$ *no-Blue-clique* **by** *blast*
have $\text{ card } (K \cup Aseq \text{ halted-point}) = k$
proof (*subst card-Un-disjnt*)
show *finite K finite (Aseq halted-point)*
using K *finite-Aseq finite-Yseq infinite-super* **by** *blast+*
show *disjnt K (Aseq halted-point)*
using *valid-state-seq[of halted-point] K disjnt-subset1*
by (*auto simp: valid-state-def disjoint-state-def*)
have $\text{ card } (Aseq \text{ halted-point}) = t$
using *red-step-eq-Aseq R-def t-def* **by** *presburger*
then show $\text{ card } K + \text{ card } (Aseq \text{ halted-point}) = k$
using *Aseq-less-k[OF] nat-less-le KRed size-clique-def* **by** *force*
qed
moreover have *clique (K ∪ Aseq halted-point) Red*
proof –
obtain $K \subseteq V \ Aseq \text{ halted-point} \subseteq V$
by (*meson Aseq-subset-V KRed size-clique-def*)
moreover have *clique K Red*
using $KRed$ *size-clique-def* **by** *blast*
moreover have *clique (Aseq halted-point) Red*
by (*meson A-Red-clique valid-state-seq*)
moreover have *all-edges-betw-un (Aseq halted-point) (Yseq halted-point) ⊆ Red*
Red
using *valid-state-seq[of halted-point] K*
by (*auto simp: valid-state-def RB-state-def all-edges-betw-un-Un2*)
then have *all-edges-betw-un K (Aseq halted-point) ⊆ Red*
using K *all-edges-betw-un-mono2 all-edges-betw-un-commute* **by** *blast*
ultimately show *?thesis*

```

    by (simp add: local.clique-Un)
  qed
  ultimately have size-clique k (K ∪ Aseq halted-point) Red
    using KRed Aseq-subset-V by (auto simp: size-clique-def)
  then have False
    using no-Red-clique by blast
}
ultimately have *: 2 powr (- real s - real t + ok-fun-61 k - 2) * nV' < RN
k (k-t)
  by fastforce
  have - real s - real t + ok-fun-61 k - 2 + log 2 nV' = log 2 (2 powr (- real
s - real t + ok-fun-61 k - 2) * nV')
    using add-log-eq-powr ⟨nV' ≥ 2⟩ by auto
  also have ... ≤ log 2 (RN k (k-t))
    using * Transcendental.log-mono ⟨nV' ≥ 2⟩ less-eq-real-def by auto
  finally show log 2 nV' ≤ log 2 (RN k (k - t)) + real s + real t + 2 - ok-fun-61
k
    by linarith
  qed

```

10.3 Theorem 11.1

definition *FF* :: *nat* ⇒ *real* ⇒ *real* ⇒ *real* **where**

FF ≡ λk x y. log 2 (RN k (nat[real k - x * real k])) / real k + x + y

definition *GG* :: *real* ⇒ *real* ⇒ *real* ⇒ *real* **where**

GG ≡ λμ x y. log 2 (1/μ) + x * log 2 (1/(1-μ)) + y * log 2 (μ * (x+y) / y)

definition *FF-bound* :: *nat* ⇒ *real* ⇒ *real* **where**

FF-bound ≡ λk u. *FF* k 0 u + 1

lemma *log2-RN-ge0*: 0 ≤ log 2 (RN k k) / k

proof (cases k=0)

case *False*

then have RN k k ≥ 1

by (simp add: RN-eq-0-iff leI)

then show ?thesis

by simp

qed *auto*

lemma *le-FF-bound*:

assumes *x*: *x* ∈ {0..1} **and** *y* ∈ {0..u}

shows *FF* k x y ≤ *FF-bound* k u

proof (cases [k - x*k] = 0)

case *True* — to handle the singularity

with *assms* *log2-RN-ge0*[of k] **show** ?thesis

by (simp add: True *FF-def* *FF-bound-def* *log-def*)

next

```

case False
with gr0I have  $k > 0$  by fastforce
with False assms have  $0 < \lfloor k - x*k \rfloor$ 
  using linorder-neqE-linordered-idom by fastforce
have le-k:  $k - x*k \leq k$ 
  using x by auto
then have le-k:  $\text{nat } \lfloor k - x*k \rfloor \leq k$ 
  by linarith
have  $\log 2 (RN\ k\ (\text{nat } \lfloor k - x*k \rfloor)) / k \leq \log 2 (RN\ k\ k) / k$ 
proof (intro divide-right-mono Transcendental.log-mono)
  show  $0 < \text{real } (RN\ k\ (\text{nat } \lfloor k - x*k \rfloor))$ 
    by (metis RN-eq-0-iff <k>0> gr-zeroI * of-nat-0-less-iff zero-less-nat-eq)
qed (auto simp: RN-mono le-k)
then show ?thesis
  using assms False le-SucE by (fastforce simp: FF-def FF-bound-def)
qed

```

```

lemma FF2:  $y' \leq y \implies FF\ k\ x\ y' \leq FF\ k\ x\ y$ 
by (simp add: FF-def)

```

```

lemma FF-GG-bound:
  assumes  $\mu: 0 < \mu < 1$  and  $x: x \in \{0..1\}$  and  $y: y \in \{0.. \mu * x / (1-\mu) + \eta\}$ 
  shows  $\min (FF\ k\ x\ y) (GG\ \mu\ x\ y) + \eta \leq FF\text{-bound}\ k\ (\mu / (1-\mu) + \eta) + \eta$ 
proof -
  have FF-ub:  $FF\ k\ x\ y \leq FF\text{-bound}\ k\ (\mu / (1-\mu) + \eta)$ 
  proof (rule order.trans)
    show  $FF\ k\ x\ y \leq FF\text{-bound}\ k\ y$ 
    using x y by (simp add: le-FF-bound)
  next
    have  $y \leq \mu / (1-\mu) + \eta$ 
    using x y  $\mu$  by simp (smt (verit, best) frac-le mult-left-le)
    then show  $FF\text{-bound}\ k\ y \leq FF\text{-bound}\ k\ (\mu / (1-\mu) + \eta)$ 
    by (simp add: FF-bound-def FF-def)
qed
show ?thesis
  using FF-ub by auto
qed

```

```

context P0-min
begin

```

```

definition ok-fun-11-1  $\equiv \lambda \mu\ k. \max (ok\text{-fun-11-2}\ \mu\ k) (2 - ok\text{-fun-61}\ k)$ 

```

```

lemma ok-fun-11-1:
  assumes  $0 < \mu < 1$ 
  shows  $ok\text{-fun-11-1}\ \mu \in o(\text{real})$ 
  unfolding ok-fun-11-1-def
  by (simp add: assms const-smallo-real maxmin-in-smallo ok-fun-11-2 ok-fun-61)

```

sum-in-smallo)

lemma *eventually-ok111-le- η* :

assumes $\eta > 0$ **and** $\mu: 0 < \mu < 1$

shows $\forall^\infty k. \text{ok-fun-11-1 } \mu \ k / k \leq \eta$

proof –

have $(\lambda k. \text{ok-fun-11-1 } \mu \ k / k) \in o(\lambda k. 1)$

using *eventually-mono ok-fun-11-1 [OF μ]* **by** (*fastforce simp: smallo-def divide-simps*)

with *assms* **have** $\forall^\infty k. |\text{ok-fun-11-1 } \mu \ k| / k \leq \eta$

by (*auto simp: smallo-def*)

then show *?thesis*

by (*metis (mono-tags, lifting) eventually-mono abs-divide abs-le-D1 abs-of-nat*)

qed

lemma *eventually-powr-le- η* :

assumes $\eta > 0$

shows $\forall^\infty k. (2 / (1 - \mu)) * k \text{ powr } (-1/20) \leq \eta$

using *assms* **by** *real-asymp*

definition *Big-From-11-1* \equiv

$\lambda \eta \mu k. \text{Big-From-11-2 } \mu \ k \wedge \text{Big-ZZ-8-5 } \mu \ k \wedge \text{Big-Y-6-1 } \mu \ k \wedge \text{ok-fun-11-1 } \mu \ k / k \leq \eta/2$

$\wedge (2 / (1 - \mu)) * k \text{ powr } (-1/20) \leq \eta/2$

$\wedge \text{Big-Closer-10-1 } (1/101) (\text{nat}\lceil k/100\rceil) \wedge 3 / (k * \ln 2) \leq \eta/2 \wedge k \geq 3$

In sections 9 and 10 (and by implication all proceeding sections), we needed to consider a closed interval of possible values of μ . Let's hope, maybe not here. The fact below can only be proved with the strict inequality $0 < \eta$, which is why it is also strict in the theorems depending on this property.

lemma *Big-From-11-1*:

assumes $\eta > 0$ $0 < \mu < 1$

shows $\forall^\infty k. \text{Big-From-11-1 } \eta \ \mu \ k$

proof –

have $\forall^\infty l. \text{Big-Closer-10-1 } (1/101) \ l$

by (*rule Big-Closer-10-1*) *auto*

then have $a: \forall^\infty k. \text{Big-Closer-10-1 } (1/101) (\text{nat}\lceil k/100\rceil)$

unfolding *eventually-sequentially*

by (*meson le-divide-eq-numeral1(1) le-natceiling-iff nat-ceiling-le-eq*)

have $b: \forall^\infty k. 3 / (k * \ln 2) \leq \eta/2$

using $\langle \eta > 0 \rangle$ **by** *real-asymp*

show *?thesis*

unfolding *Big-From-11-1-def*

using *assms a b Big-From-11-2[of $\mu \ \mu$] Big-ZZ-8-5[of $\mu \ \mu$] Big-Y-6-1[of $\mu \ \mu$]*

using *eventually-ok111-le- η [of $\eta/2$] eventually-powr-le- η [of $\eta/2$]*

by (*auto simp: eventually-conj-iff all-imp-conj-distrib eventually-sequentially*)

qed

The actual proof of theorem 11.1 is now combined with the development

of section 12, since the concepts seem to be inescapably mixed up.

end

end

11 The Proof of Theorem 1.1

theory *The-Proof*

imports *From-Diagonal*

begin

11.1 The bounding functions

definition $H \equiv \lambda p. -p * \log 2 p - (1-p) * \log 2 (1-p)$

definition dH **where** $dH \equiv \lambda x::real. -\ln(x)/\ln(2) + \ln(1 - x)/\ln(2)$

lemma dH [*derivative-intros*]:

assumes $0 < x < 1$

shows (H has-real-derivative dH x) (*at* x)

unfolding H -def dH -def \log -def

by (*rule derivative-eq-intros | use assms in force*)+

lemma $H0$ [*simp*]: $H 0 = 0$ **and** $H1$ [*simp*]: $H 1 = 0$

by (*auto simp: H-def*)

lemma H -reflect: $H (1-p) = H p$

by (*simp add: H-def*)

lemma H -ge0:

assumes $0 \leq p \leq 1$

shows $0 \leq H p$

unfolding H -def

by (*smt (verit, best) assms mult-minus-left mult-le-0-iff zero-less-log-cancel-iff*)

Going up, from 0 to 1/2

lemma H -half-mono:

assumes $0 \leq p' \leq p \leq 1/2$

shows $H p' \leq H p$

proof (*cases p'=0*)

case *True*

then have $H p' = 0$

by (*auto simp: H-def*)

then show *?thesis*

by (*smt (verit) H-ge0 True assms(2) assms(3) divide-le-eq-1-pos*)

next

case *False*

with *assms* **have** $p' > 0$ **by** *simp*
have $dH(1/2) = 0$
by (*simp add: dH-def*)
moreover
have $dH x \geq 0$ **if** $0 < x \leq 1/2$ **for** x
using *that* **by** (*simp add: dH-def divide-right-mono*)
ultimately show *?thesis*
by (*smt (verit) dH DERIV-nonneg-imp-nondecreasing <p'>0> assms le-divide-eq-1-pos*)
qed

Going down, from 1/2 to 1

lemma *H-half-mono'*:
assumes $1/2 \leq p' \leq p \leq 1$
shows $H p' \geq H p$
using *H-half-mono [of 1-p 1-p'] H-reflect assms* **by** *auto*

lemma *H-half*: $H(1/2) = 1$
by (*simp add: H-def log-divide*)

lemma *H-le1*:
assumes $0 \leq p \leq 1$
shows $H p \leq 1$
by (*smt (verit, best) H0 H1 H-ge0 H-half-mono H-half-mono' H-half assms*)

Many thanks to Fedor Petrov on mathoverflow

lemma *H-12-1*:
fixes $a b :: \text{nat}$
assumes $a \geq b$
shows $\log 2 (a \text{ choose } b) \leq a * H(b/a)$
proof (*cases a=b ∨ b=0*)
case *True*
with *assms* **show** *?thesis*
by (*auto simp: H-def*)
next
let $?p = b/a$
case *False*
then have $p01: 0 < ?p \leq 1$
using *assms* **by** *auto*
then have $(a \text{ choose } b) * ?p ^ b * (1 - ?p) ^ (a - b) \leq (?p + (1 - ?p)) ^ a$
by (*subst binomial-ring*) (*force intro!: member-le-sum assms*)
also have $\dots = 1$
by *simp*
finally have $\S: (a \text{ choose } b) * ?p ^ b * (1 - ?p) ^ (a - b) \leq 1$
have $\log 2 (a \text{ choose } b) + b * \log 2 ?p + (a - b) * \log 2 (1 - ?p) \leq 0$
using *Transcendental.log-mono [OF - -] False assms*
by (*force simp add: p01 log-mult log-nat-power*)
then show *?thesis*
using $p01$ *False assms* **unfolding** *H-def* **by** (*simp add: divide-simps*)
qed

definition $gg \equiv GG (2/5)$

lemma $gg\text{-eq}$: $gg\ x\ y = \log 2 (5/2) + x * \log 2 (5/3) + y * \log 2 ((2 * (x+y)) / (5*y))$

by (*simp add: gg-def GG-def*)

definition $f1 \equiv \lambda x\ y. x + y + (2-x) * H(1/(2-x))$

definition $f2 \equiv \lambda x\ y. f1\ x\ y - (1 / (40 * \ln 2)) * ((1-x) / (2-x))$

definition $ff \equiv \lambda x\ y. \text{if } x < 3/4 \text{ then } f1\ x\ y \text{ else } f2\ x\ y$

Incorporating Bhavik's idea, which gives us a lower bound for γ of 1/101

definition $ffGG :: real \Rightarrow real \Rightarrow real \Rightarrow real$ **where**

$ffGG \equiv \lambda \mu\ x\ y. \max 1.9 (\min (ff\ x\ y) (GG\ \mu\ x\ y))$

The proofs involving *Sup* are needlessly difficult because ultimately the sets involved are finite, eliminating the need to demonstrate boundedness. Simpler might be to use the extended reals.

lemma $f1\text{-le}$:

assumes $x \leq 1$

shows $f1\ x\ y \leq y + 2$

unfolding $f1\text{-def}$

using $H\text{-le1}$ [*of* $1/(2-x)$] *assms*

by (*smt (verit) divide-le-eq-1-pos divide-nonneg-nonneg mult-left-le*)

lemma $ff\text{-le4}$:

assumes $x \leq 1\ y \leq 1$

shows $ff\ x\ y \leq 4$

proof –

have $ff\ x\ y \leq f1\ x\ y$

using *assms* **by** (*simp add: ff-def f2-def*)

also have $\dots \leq 4$

using *assms* **by** (*smt (verit) f1-le*)

finally show *?thesis* .

qed

lemma $ff\text{-GG-bound}$:

assumes $x \leq 1\ y \leq 1$

shows $ffGG\ \mu\ x\ y \leq 4$

using $ff\text{-le4}$ [*OF* *assms*] **by** (*auto simp: ffGG-def*)

lemma $bdd\text{-above-ff-GG}$:

assumes $x \leq 1\ u \leq 1$

shows $bdd\text{-above} ((\lambda y. ffGG\ \mu\ x\ y + \eta) \text{ ‘ } \{0..u\})$

using $ff\text{-GG-bound}$ *assms*

by (*intro bdd-above.I2 [where* $M = 4 + \eta]$ *force*)

lemma *bdd-above-SUP-ff-GG*:
assumes $0 \leq u \leq 1$
shows *bdd-above* $((\lambda x. \lfloor \lfloor y \in \{0..u\} \rfloor \cdot \text{ffGG } \mu \ x \ y + \eta) \text{ ' } \{0..1\})$
using *bdd-above-ff-GG assms*
by (*intro bdd-aboveI [where M = 4 + η]*) (*auto simp: cSup-le-iff ff-GG-bound Pi-iff*)

Claim (62). A singularity if $x = 1$. Okay if we put $\ln(0) = 0$

lemma *FF-le-f1*:
fixes $k::\text{nat}$ **and** $x \ y::\text{real}$
assumes $x: 0 \leq x \leq 1$ **and** $y: 0 \leq y \leq 1$
shows *FF* $k \ x \ y \leq \text{f1} \ x \ y$
proof (*cases nat[k - x*k] = 0*)
case *True*
with x **show** *?thesis*
by (*simp add: FF-def f1-def H-ge0 log-def*)
next
case *False*
let $?kl = k + k - \text{nat} \lfloor x*k \rfloor$
have *kk-less-1*: $k / ?kl < 1$
using x *False* **by** (*simp add: field-split-simps, linarith*)
have $le: \text{nat} \lfloor k - x*k \rfloor \leq k - \text{nat} \lfloor x*k \rfloor$
using *floor-ceiling-diff-le x*
by (*meson mult-left-le-one-le mult-nonneg-nonneg of-nat-0-le-iff*)
have $k > 0$
using *False zero-less-iff-neq-zero* **by** *fastforce*
have *RN-gt0*: $RN \ k \ (\text{nat} \lfloor k - x*k \rfloor) > 0$
by (*metis False RN-eq-0-iff <k>0> grOI*)
then have $\S: RN \ k \ (\text{nat} \lfloor k - x*k \rfloor) \leq k + \text{nat} \lfloor k - x*k \rfloor$ *choose k*
using *RN-le-choose* **by** *force*
also have $\dots \leq k + k - \text{nat} \lfloor x*k \rfloor$ *choose k*
using *False Nat.le-diff-conv2 binomial-right-mono le* **by** *fastforce*
finally have $RN \ k \ (\text{nat} \lfloor \text{real } k - x*k \rfloor) \leq ?kl$ *choose k* .
with *RN-gt0* **have** *FF* $k \ x \ y \leq \log 2 \ (?kl \ \text{choose } k) / k + x + y$
by (*simp add: FF-def divide-right-mono nat-less-real-le*)
also have $\dots \leq (?kl * H(k/?kl)) / k + x + y$
proof -
have $k \leq k + k - \text{nat} \lfloor x*k \rfloor$
using *False* **by** *linarith*
then show *?thesis*
by (*simp add: H-12-1 divide-right-mono*)
qed
also have $\dots \leq \text{f1} \ x \ y$
proof -
have $1: ?kl / k \leq 2 - x$
using x **by** (*simp add: field-split-simps*)
have $2: H \ (k / ?kl) \leq H \ (1 / (2 - x))$
proof (*intro H-half-mono'*)
show $1 / (2 - x) \leq k / ?kl$

```

    using x False by (simp add: field-split-simps, linarith)
  qed (use x kk-less-1 in auto)
  have ?kl / k * H (k / ?kl) ≤ (2-x) * H (1 / (2-x))
    using x mult-mono [OF 1 2 - H-ge0] kk-less-1 by fastforce
  then show ?thesis
    by (simp add: f1-def)
  qed
  finally show ?thesis .
qed

```

Bhavik's *eleven-one-large-end*

```

lemma f1-le-19:
  fixes k::nat and x y::real
  assumes x: 0.99 ≤ x x ≤ 1 and y: 0 ≤ y y ≤ 3/4
  shows f1 x y ≤ 1.9
proof -
  have A: 2-x ≤ 1.01
    using x by simp
  have H (1 / (2-x)) ≤ H (1 / (2-0.99))
    using x by (intro H-half-mono') (auto simp: divide-simps)
  also have ... ≤ 0.081
    unfolding H-def by (approximation 15)
  finally have B: H (1 / (2-x)) ≤ 0.081 .
  have (2-x) * H (1 / (2-x)) ≤ 1.01 * 0.081
    using mult-mono [OF A B] x
    by (smt (verit) A H-ge0 divide-le-eq-1-pos divide-nonneg-nonneg)
  with assms show ?thesis by (auto simp: f1-def)
qed

```

Claim (63) in weakened form; we get rid of the extra bit later

```

lemma (in P0-min) FF-le-f2:
  fixes k::nat and x y::real
  assumes x: 3/4 ≤ x x ≤ 1 and y: 0 ≤ y y ≤ 1
  and l: real l = k - x*k
  assumes p0-min-101: p0-min ≤ 1 - 1/5
  defines γ ≡ real l / (real k + real l)
  defines γ0 ≡ min γ (0.07)
  assumes γ > 0
  shows FF k x y ≤ f2 x y + ok-fun-10-1 γ k / (k * ln 2)
proof -
  have l > 0
    using ⟨γ > 0⟩ γ-def less-irrefl by fastforce
  have x > 0
    using x by linarith
  with l have k ≥ l
    by (smt (verit, del-insts) of-nat-0-le-iff of-nat-le-iff pos-prod-lt)
  with ⟨0 < l⟩ have k > 0 by force
  have RN-gt0: RN k l > 0
    by (metis RN-eq-0-iff ⟨0 < k⟩ ⟨0 < l⟩ gr0I)

```

```

define  $\delta$  where  $\delta \equiv \gamma/40$ 
have  $A: l / \text{real}(k+l) = (1-x)/(2-x)$ 
  using  $x \langle k > 0 \rangle$  by (simp add: l field-simps)
have  $B: \text{real}(k+l) / k = 2-x$ 
  using  $\langle 0 < k \rangle l$  by (auto simp: divide-simps left-diff-distrib)
have  $\gamma: \gamma \leq 1/5$ 
  using  $x A$  by (simp add:  $\gamma$ -def)
have  $1 - 1 / (2-x) = (1-x) / (2-x)$ 
  using  $x$  by (simp add: divide-simps)
then have  $\text{Heq}: H (1 / (2-x)) = H ((1-x) / (2-x))$ 
  by (metis H-reflect)
have  $RN k l \leq \exp (-\delta*k + \text{ok-fun-10-1 } \gamma k) * (k+l \text{ choose } l)$ 
  unfolding  $\delta$ -def  $\gamma$ -def
proof (rule Closer-10-1-unconditional)
  show  $0 < l / (\text{real } k + \text{real } l) l / (\text{real } k + \text{real } l) \leq 1/5$ 
    using  $\gamma \langle \gamma > 0 \rangle$  by (auto simp:  $\gamma$ -def)
    have  $\min (l / (k + \text{real } l)) 0.07 > 0$ 
      using  $\langle l > 0 \rangle$  by force
    qed (use p0-min-101 in auto)
  with  $RN$ -gt0 have  $FF k x y \leq \log 2 (\exp (-\delta*k + \text{ok-fun-10-1 } \gamma k) * (k+l \text{ choose } l)) / k + x + y$ 
    unfolding  $FF$ -def
    by (intro add-mono divide-right-mono Transcendental.log-mono; simp flip: l)
    also have  $\dots = (\log 2 (\exp (-\delta*k + \text{ok-fun-10-1 } \gamma k)) + \log 2 (k+l \text{ choose } l)) / k + x + y$ 
      by (simp add: log-mult)
    also have  $\dots \leq ((-\delta*k + \text{ok-fun-10-1 } \gamma k) / \ln 2 + (k+l) * H(l/(k+l))) / k + x + y$ 
      using  $H$ -12-1
    by (smt (verit, ccfv-SIG) log-exp divide-right-mono le-add2 of-nat-0-le-iff)
    also have  $\dots = (-\delta*k + \text{ok-fun-10-1 } \gamma k) / k / \ln 2 + (k+l) / k * H(l/(k+l)) + x + y$ 
      by argo
    also have  $\dots = -\delta / \ln 2 + \text{ok-fun-10-1 } \gamma k / (k * \ln 2) + (2-x) * H((1-x)/(2-x)) + x + y$ 
      proof -
        have  $(-\delta*k + \text{ok-fun-10-1 } \gamma k) / k / \ln 2 = -\delta / \ln 2 + \text{ok-fun-10-1 } \gamma k / (k * \ln 2)$ 
          using  $\langle 0 < k \rangle$  by (simp add: divide-simps)
          with  $A B$  show ?thesis
          by presburger
        qed
    also have  $\dots = -(\log 2 (\exp 1) / 40) * (1-x) / (2-x) + \text{ok-fun-10-1 } \gamma k / (k * \ln 2) + (2-x) * H((1-x)/(2-x)) + x + y$ 
      using  $A$  by (force simp:  $\delta$ -def  $\gamma$ -def field-simps)
    also have  $\dots \leq f2 x y + \text{ok-fun-10-1 } \gamma k / (\text{real } k * \ln 2)$ 
      by (simp add: Heq f1-def f2-def mult-ac)
    finally show ?thesis .
qed

```

The body of the proof has been extracted to allow the symmetry argument. And $1/12$ is $3/4-2/3$, the latter number corresponding to $\mu = (2::'a) / (5::'a)$

lemma (in *Book-Basis*) *From-11-1-Body*:

```

fixes V :: 'a set
assumes  $\mu$ :  $0 < \mu$   $\mu \leq 2/5$  and  $\eta$ :  $0 < \eta$   $\eta \leq 1/12$ 
and ge-RN:  $Suc\ nV \geq RN\ k\ k$ 
and Red: graph-density Red  $\geq 1/2$ 
and p0-min12: p0-min  $\leq 1/2$ 
and Red-E:  $Red \subseteq E$  and Blue-def:  $Blue = E \setminus Red$ 
and no-Red-K:  $\neg (\exists K. \text{size-clique } k\ K\ Red)$ 
and no-Blue-K:  $\neg (\exists K. \text{size-clique } k\ K\ Blue)$ 
and big: Big-From-11-1  $\eta\ \mu\ k$ 
shows  $\log\ 2\ (RN\ k\ k) / k \leq (SUP\ x \in \{0..1\}. SUP\ y \in \{0..3/4\}. \text{ffGG } \mu\ x\ y + \eta)$ 
proof -
have 12:  $3/4 - 2/3 = (1/12::real)$ 
by simp
define  $\eta'$  where  $\eta' \equiv \eta/2$ 
have  $\eta'$ :  $0 < \eta'$   $\eta' \leq 1/12$ 
using  $\eta$  by (auto simp:  $\eta'$ -def)
have  $k > 0$  and big101: Big-Closer-10-1 (1/101) (nat[ $k/100$ ]) and ok-fun-10-1-le:
 $3 / (k * \ln\ 2) \leq \eta'$ 
using big by (auto simp: Big-From-11-1-def  $\eta'$ -def)
interpret No-Cliques where  $l=k$ 
using assms unfolding No-Cliques-def No-Cliques-axioms-def
using Book-Basis-axioms P0-min-axioms by blast
obtain X0 Y0 where card-X0:  $card\ X0 \geq nV/2$  and card-Y0:  $card\ Y0 =$ 
gorder div 2
and  $X0 = V \setminus Y0\ Y0 \subseteq V$ 
and p0-half:  $1/2 \leq \text{gen-density } Red\ X0\ Y0$ 
and Book  $V\ E\ p0\ \text{min}\ Red\ Blue\ k\ k\ \mu\ X0\ Y0$ 
proof (rule to-Book)
show p0-min  $\leq \text{graph-density } Red$ 
using p0-min12 Red by linarith
show  $0 < \mu$   $\mu < 1$ 
using  $\mu$  by auto
qed (use infinite-UNIV p0-min Blue-def Red  $\mu$  in auto)
then interpret Book:  $V\ E\ p0\ \text{min}\ Red\ Blue\ k\ k\ \mu\ X0\ Y0$ 
by meson
define  $\mathcal{R}$  where  $\mathcal{R} \equiv \text{Step-class } \{\text{red-step}\}$ 
define  $\mathcal{S}$  where  $\mathcal{S} \equiv \text{Step-class } \{\text{dboost-step}\}$ 
define  $t$  where  $t \equiv card\ \mathcal{R}$ 
define  $s$  where  $s \equiv card\ \mathcal{S}$ 
define  $x$  where  $x \equiv t/k$ 
define  $y$  where  $y \equiv s/k$ 
have sts:  $(s + \text{real } t) / s = (x+y) / y$ 
using  $\langle k > 0 \rangle$  by (simp add: x-def y-def divide-simps)
have  $t < k$ 

```

```

  by (simp add:  $\mathcal{R}$ -def  $\mu$  t-def red-step-limit)
then obtain x01:  $0 \leq x < 1$ 
  by (auto simp: x-def)

have big41: Big-Blue-4-1  $\mu$  k and big61: Big-Y-6-1  $\mu$  k
  and big85: Big-ZZ-8-5  $\mu$  k and big11-2: Big-From-11-2  $\mu$  k
  and ok111-le: ok-fun-11-1  $\mu$  k /  $k \leq \eta'$ 
  and powr-le:  $(2 / (1 - \mu)) * k \text{ powr } (-1/20) \leq \eta'$  and  $k > 0$ 
using big by (auto simp: Big-From-11-1-def Big-Y-6-1-def Big-Y-6-2-def  $\eta'$ -def)
then have big53: Big-Red-5-3  $\mu$  k
  by (meson Big-From-11-2-def)
have  $\mu < 1$ 
  using  $\mu$  by auto

have s < k
  unfolding s-def  $\mathcal{S}$ -def
  by (meson  $\mu$  le-less-trans bblue-dboost-step-limit big41 le-add2)
then obtain y01:  $0 \leq y < 1$ 
  by (auto simp: y-def)

Now that  $x$  and  $y$  are fixed, here's the body of the outer supremum
define w where  $w \equiv (\bigsqcup y \in \{0..3/4\}. \text{ffGG } \mu x y + \eta)$ 
show ?thesis
proof (intro cSup-upper2 imageI)
  show  $w \in (\lambda x. \bigsqcup y \in \{0..3/4\}. \text{ffGG } \mu x y + \eta) \text{ ' } \{0..1\}$ 
    using x01 by (force simp: w-def intro!: image-eqI [where x=x])
next
  have  $\mu 23$ :  $\mu / (1 - \mu) \leq 2/3$ 
    using  $\mu$  by (simp add: divide-simps)
  have beta-le: bigbeta  $\leq \mu$ 
    using  $\langle \mu < 1 \rangle$   $\mu$  big53 bigbeta-le by blast
  have  $s \leq (\text{bigbeta} / (1 - \text{bigbeta})) * t + (2 / (1 - \mu)) * k \text{ powr } (19/20)$ 
    using ZZ-8-5 [OF big85]  $\mu$  by (auto simp:  $\mathcal{R}$ -def  $\mathcal{S}$ -def s-def t-def)
  also have  $\dots \leq (\mu / (1 - \mu)) * t + (2 / (1 - \mu)) * k \text{ powr } (19/20)$ 
  by (smt (verit, ccfv-SIG)  $\langle \mu < 1 \rangle$   $\mu$  beta-le frac-le mult-right-mono of-nat-0-le-iff)
  also have  $\dots \leq (\mu / (1 - \mu)) * t + (2 / (1 - \mu)) * (k \text{ powr } (-1/20)) * k \text{ powr } (-1/20)$ 
1)
    unfolding powr-add [symmetric] by simp
  also have  $\dots \leq (2/3) * t + (2 / (1 - \mu)) * (k \text{ powr } (-1/20)) * k$ 
    using mult-right-mono [OF  $\mu 23$ , of t] by (simp add: mult-ac)
  also have  $\dots \leq (3/4 - \eta') * k + (2 / (1 - \mu)) * (k \text{ powr } (-1/20)) * k$ 
proof -
  have  $(2/3) * t \leq (2/3) * k$ 
    using  $\langle t < k \rangle$  by simp
  then show ?thesis
    using 12  $\eta'$  by (smt (verit) mult-right-mono of-nat-0-le-iff)
qed
finally have  $s \leq (3/4 - \eta') * k + (2 / (1 - \mu)) * k \text{ powr } (-1/20) * k$ 
  by simp

```

with *mult-right-mono* [*OF pour-le, of k*]
have $\dagger: s \leq 3/4 * k$
by (*simp add: mult.commute right-diff-distrib*)
then have $y \leq 3/4$
by (*metis † <0 < k> of-nat-0-less-iff pos-divide-le-eq y-def*)

have *k-minus-t*: $\text{nat } \lfloor \text{real } k - \text{real } t \rfloor = k - t$
by *linarith*
have $nV \text{ div } 2 \leq \text{card } Y0$
by (*simp add: card-Y0*)
then have $\S: \log 2 (\text{Suc } nV) \leq \log 2 (\text{RN } k (k-t)) + s + t + 2 - \text{ok-fun-61}$
 k
using *From-11-3 [OF - big61] p0-half μ* **by** (*auto simp: \mathcal{R} -def \mathcal{S} -def p0-def s-def t-def*)

define *l* **where** $l \equiv k - t$
define γ **where** $\gamma \equiv \text{real } l / (\text{real } k + \text{real } l)$
have $\gamma < 1$
using $\langle t < k \rangle$ **by** (*simp add: γ -def*)
have $nV \text{ div } 2 \leq \text{card } X0$
using *card-X0* **by** *linarith*
then have *112*: $\log 2 (\text{Suc } nV) \leq k * \log 2 (1/\mu) + t * \log 2 (1 / (1-\mu)) +$
 $s * \log 2 (\text{ratio } \mu s t)$
 $+ \text{ok-fun-11-2 } \mu k$
using *From-11-2 [OF - big11-2] p0-half μ*
unfolding *s-def t-def p0-def \mathcal{R} -def \mathcal{S} -def* **by** *force*
have $\log 2 (\text{Suc } nV) / k \leq \log 2 (1/\mu) + x * \log 2 (1 / (1-\mu)) + y * \log 2$
(*ratio $\mu s t$*)
 $+ \text{ok-fun-11-2 } \mu k / k$
using $\langle k > 0 \rangle$ *divide-right-mono [OF 112, of k]*
by (*simp add: add-divide-distrib x-def y-def*)
also have $\dots = GG \mu x y + \text{ok-fun-11-2 } \mu k / k$
by (*metis GG-def sts times-divide-eq-right*)
also have $\dots \leq GG \mu x y + \text{ok-fun-11-1 } \mu k / k$
by (*simp add: ok-fun-11-1-def divide-right-mono*)
finally have *le-GG*: $\log 2 (\text{Suc } nV) / k \leq GG \mu x y + \text{ok-fun-11-1 } \mu k / k .$

have $\log 2 (\text{Suc } nV) / k \leq \log 2 (\text{RN } k (k-t)) / k + x + y + (2 - \text{ok-fun-61}$
 $k) / k$
using $\langle k > 0 \rangle$ *divide-right-mono [OF \S , of k]* *add-divide-distrib x-def y-def*
by (*smt (verit) add-uminus-conv-diff of-nat-0-le-iff*)
also have $\dots = FF k x y + (2 - \text{ok-fun-61 } k) / k$
by (*simp add: FF-def x-def k-minus-t*)
finally have *DD*: $\log 2 (\text{Suc } nV) / k \leq FF k x y + (2 - \text{ok-fun-61 } k) / k .$

have $\text{RN } k k > 0$
by (*metis RN-eq-0-iff <k>0> gr0I*)
moreover have $\log 2 (\text{Suc } nV) / k \leq \text{ffGG } \mu x y + \eta$
proof (*cases $x < 0.99$*) — a further case split that gives a lower bound for

```

gamma
  case True
  have ‡: Big-Closer-10-1 (min  $\gamma$  0.07) (nat  $\lceil \gamma * \text{real } k / (1 - \gamma) \rceil$ )
  proof (intro Big-Closer-10-1-upward [OF big101])
    show  $1/101 \leq \min \gamma$  0.07
      using <k>0> <t<k> True by (simp add:  $\gamma$ -def l-def x-def divide-simps)
    with < $\gamma < 1$ > less-eq-real-def have  $k/100 \leq \gamma * k / (1 - \gamma)$ 
      by (fastforce simp: field-simps)
    then show nat  $\lceil k/100 \rceil \leq \text{nat } \lceil \gamma * k / (1 - \gamma) \rceil$ 
      using ceiling-mono nat-mono by blast
  qed
  have 122: FF k x y  $\leq$  ff x y +  $\eta'$ 
  proof -
    have FF k x y  $\leq$  f1 x y
      using x01 y01
      by (intro FF-le-f1) auto
    moreover
    have FF k x y  $\leq$  f2 x y + ok-fun-10-1  $\gamma$  k / (k * ln 2) if  $x \geq 3/4$ 
      unfolding  $\gamma$ -def
    proof (intro FF-le-f2 that)
      have  $\gamma = (1-x) / (2-x)$ 
        using <0 <k> <t <k> by (simp add: l-def  $\gamma$ -def x-def divide-simps)
      then have  $\gamma \leq 1/5$ 
        using that <x<1> by simp
      show real l = real k - x * real k
        using <t <k> by (simp add: l-def x-def)
      show 0 < l / (k + real l)
        using <t <k> l-def by auto
    qed (use x01 y01 p0-min12 in auto)
    moreover have ok-fun-10-1  $\gamma$  k / (k * ln 2)  $\leq$   $\eta'$ 
      using ‡ ok-fun-10-1-le by (simp add: ok-fun-10-1-def)
    ultimately show ?thesis
      using  $\eta'$  by (auto simp: ff-def)
  qed
  have log 2 (Suc nV) / k  $\leq$  ff x y +  $\eta'$  + (2 - ok-fun-61 k) / k
    using 122 DD by linarith
  also have ...  $\leq$  ff x y +  $\eta'$  + ok-fun-11-1  $\mu$  k / k
    by (simp add: ok-fun-11-1-def divide-right-mono)
  finally have le-ff: log 2 (Suc nV) / k  $\leq$  ff x y +  $\eta'$  + ok-fun-11-1  $\mu$  k / k .
  then show ?thesis
    using  $\eta$  ok111-le le-ff le-GG unfolding  $\eta'$ -def ffGG-def by linarith
next
  case False — in this case, we can use the existing bound involving f1
  have log 2 (Suc nV) / k  $\leq$  FF k x y + (2 - ok-fun-61 k) / k
    by (metis DD)
  also have ...  $\leq$  f1 x y + (2 - ok-fun-61 k) / k
    using x01 y01 FF-le-f1 [of x y] by simp
  also have ...  $\leq$  1.9 + (2 - ok-fun-61 k) / k
    using x01 y01 by (smt (verit) False <y  $\leq$  3/4> f1-le-19)

```

```

    also have ... ≤ ffGG μ x y + η
    by (smt (verit) P0-min.intro P0-min.ok-fun-11-1-def η'(1) η'-def divide-right-mono
ffGG-def field-sum-of-halves of-nat-0-le-iff ok111-le p0-min(1) p0-min(2))
    finally show ?thesis .
qed
ultimately have log 2 (RN k k) / k ≤ ffGG μ x y + η
    using ge-RN <k>0>
    by (smt (verit, best) Transcendental.log-mono divide-right-mono of-nat-0-less-iff
of-nat-mono)
    also have ... ≤ w
    unfolding w-def
    proof (intro cSup-upper2)
    have y ∈ {0..3/4}
    using divide-right-mono [OF †, of k] <k>0> by (simp add: x-def y-def)
    then show ffGG μ x y + η ∈ (λy. ffGG μ x y + η) ' {0..3/4}
    by blast
    next
    show bdd-above ((λy. ffGG μ x y + η) ' {0..3/4})
    by (simp add: bdd-above-ff-GG less-imp-le x01)
    qed auto
    finally show log 2 (real (RN k k)) / k ≤ w .
    next
    show bdd-above ((λx. ⌊ y ∈ {0..3/4}. ffGG μ x y + η) ' {0..1})
    by (auto intro: bdd-above-SUP-ff-GG)
    qed
qed

```

theorem (in P0-min) From-11-1:
assumes $\mu: 0 < \mu \leq 2/5$ **and** $0 < \eta \leq 1/12$
and $p0\text{-min}12: p0\text{-min} \leq 1/2$ **and** $big: Big\text{-From-11-1} \ \eta \ \mu \ k$
shows $\log 2 (RN \ k \ k) / k \leq (SUP \ x \in \{0..1\}. SUP \ y \in \{0..3/4\}. ffGG \ \mu \ x \ y + \eta)$
proof –
 have $k \geq 3$
 using big by (auto simp: Big-From-11-1-def)
 define n where $n \equiv RN \ k \ k - 1$
 define V where $V \equiv \{..<n\}$
 define E where $E \equiv all\text{-edges} \ V$
 interpret Book-Basis $V \ E$
proof qed (auto simp: V-def E-def comp-sgraph.wellformed comp-sgraph.two-edges)

 have $RN \ k \ k \geq 3$
 using $\langle k \geq 3 \rangle$ RN-3plus le-trans by blast
 then have $n < RN \ k \ k$
 by (simp add: n-def)
 moreover have $[simp]: nV = n$
 by (simp add: V-def)
 ultimately obtain Red Blue
 where Red-E: $Red \subseteq E$ **and** Blue-def: $Blue = E \setminus Red$

```

    and no-Red-K:  $\neg (\exists K. \text{size-clique } k \ K \ Red)$ 
    and no-Blue-K:  $\neg (\exists K. \text{size-clique } k \ K \ Blue)$ 
  by (metis  $\langle n < RN \ k \ k \rangle$  less-RN-Red-Blue)
  have Blue-E:  $Blue \subseteq E$  and disjnt-Red-Blue:  $\text{disjnt } Red \ Blue$  and Blue-eq:  $Blue = \text{all-edges } V \setminus Red$ 
  using complete by (auto simp: Blue-def disjnt-iff E-def)
  have  $nV > 1$ 
  using  $\langle RN \ k \ k \geq 3 \rangle \langle nV = n \rangle$  n-def by linarith
  with graph-size have graph-size  $> 0$ 
  by simp
  then have graph-density  $E = 1$ 
  by (simp add: graph-density-def)
  then have graph-density  $Red + \text{graph-density } Blue = 1$ 
  using graph-density-Un [OF disjnt-Red-Blue] by (simp add: Blue-def Red-E Un-absorb1)
  then consider (Red) graph-density  $Red \geq 1/2$  | (Blue) graph-density  $Blue \geq 1/2$ 
  by force
  then show ?thesis
  proof cases
    case Red
    show ?thesis
    proof (intro From-11-1-Body)
    next
    show  $RN \ k \ k \leq \text{Suc } nV$ 
    by (simp add: n-def)
    show  $\nexists K. \text{size-clique } k \ K \ Red$ 
    using no-Red-K by blast
    show  $\nexists K. \text{size-clique } k \ K \ Blue$ 
    using no-Blue-K by blast
    qed (use Red Red-E Blue-def assms in auto)
  next
  case Blue
  show ?thesis
  proof (intro From-11-1-Body)
  show  $RN \ k \ k \leq \text{Suc } nV$ 
  by (simp add: n-def)
  show  $Blue \subseteq E$ 
  by (simp add: Blue-E)
  show  $Red = E \setminus Blue$ 
  by (simp add: Blue-def Red-E double-diff)
  show  $\nexists K. \text{size-clique } k \ K \ Red$ 
  using no-Red-K by blast
  show  $\nexists K. \text{size-clique } k \ K \ Blue$ 
  using no-Blue-K by blast
  qed (use Blue Red-E Blue-def assms in auto)
  qed
  qed

```

11.2 The monster calculation from appendix A

11.2.1 Observation A.1

lemma *gg-increasing*:
 assumes $x \leq x' \ 0 \leq x \ 0 \leq y$
 shows $gg \ x \ y \leq gg \ x' \ y$
proof (*cases y=0*)
 case *False*
 with *assms* **show** *?thesis*
 unfolding *gg-eq* **by** (*intro add-mono mult-left-mono divide-right-mono Transcendental.log-mono*) *auto*
qed (*auto simp: gg-eq assms*)

Thanks to Manuel Eberl

lemma *continuous-on-x-ln*: *continuous-on* $\{0..\}$ $(\lambda x::real. x * \ln x)$
proof –
 have *continuous* (*at x within* $\{0..\}$) $(\lambda x. x * \ln x)$
 if $x \geq 0$ **for** $x :: real$
 proof (*cases x = 0*)
 case *True*
 have *continuous* (*at-right* 0) $(\lambda x::real. x * \ln x)$
 unfolding *continuous-within* **by** *real-asymp*
 thus *?thesis*
 using *True* **by** (*simp add: at-within-Ici-at-right*)
 qed (*auto intro!: continuous-intros*)
 thus *?thesis*
 by (*simp add: continuous-on-eq-continuous-within*)
qed

lemma *continuous-on-f1*: *continuous-on* $\{..1\}$ $(\lambda x. f1 \ x \ y)$
proof –
 have $\S: (\lambda x::real. (1 - 1/(2-x)) * \ln (1 - 1/(2-x))) = (\lambda x. x * \ln x) \circ (\lambda x. 1 - 1/(2-x))$
 by (*simp add: o-def*)
 have *cont-xln*: *continuous-on* $\{..1\}$ $(\lambda x::real. (1 - 1/(2-x)) * \ln (1 - 1/(2-x)))$
 unfolding \S
 proof (*rule continuous-intros*)
 show *continuous-on* $\{..1::real\}$ $(\lambda x. 1 - 1/(2-x))$
 by (*intro continuous-intros*) *auto*
 next
 show *continuous-on* $((\lambda x::real. 1 - 1/(2-x)) \ ‘ \{..1\})$ $(\lambda x. x * \ln x)$
 by (*rule continuous-on-subset [OF continuous-on-x-ln]*) *auto*
 qed
 show *?thesis*
 apply (*simp add: f1-def H-def log-def*)
 by (*intro continuous-on-subset [OF cont-xln] continuous-intros*) *auto*
qed

definition *df1* **where** $df1 \equiv \lambda x. \log 2 (2 * ((1-x) / (2-x)))$

```

lemma Df1 [derivative-intros]:
  assumes  $x < 1$ 
  shows  $((\lambda x. f1\ x\ y)$  has-real-derivative  $df1\ x$ ) (at  $x$ )
proof -
  have  $(2 - x * 2) = 2 * (1 - x)$ 
  by simp
  then have [simp]:  $\log\ 2\ (2 - x * 2) = \log\ 2\ (1 - x) + 1$ 
  using log-mult [of 2 1-x 2] assms by (smt (verit, best) log-eq-one)
  show ?thesis
  using assms
  unfolding f1-def H-def df1-def
  apply -
  apply (rule derivative-eq-intros | simp)+
  apply (simp add: log-divide divide-simps)
  apply (simp add: algebra-simps)
  done
qed

```

definition delta where $\text{delta} \equiv \lambda u::\text{real}. 1 / (\ln\ 2 * 40 * (2 - u)^2)$

```

lemma Df2:
  assumes  $1/2 \leq x < 1$ 
  shows  $((\lambda x. f2\ x\ y)$  has-real-derivative  $df1\ x + \text{delta}\ x$ ) (at  $x$ )
  using assms unfolding f2-def delta-def
  apply -
  apply (rule derivative-eq-intros Df1 | simp)+
  apply (simp add: divide-simps power2-eq-square)
  done

```

```

lemma antimonononff:
  assumes  $0 \leq y < 1$ 
  shows antimononon  $\{1/2..1\}$   $(\lambda x. ff\ x\ y)$ 
proof -
  have  $\S: 1 - 1 / (2-x) = (1-x) / (2-x)$  if  $x < 2$  for  $x::\text{real}$ 
  using that by (simp add: divide-simps)
  have f1:  $f1\ x'\ y \leq f1\ x\ y$ 
  if  $x \in \{1/2..1\}$   $x' \in \{1/2..1\}$   $x \leq x'$   $x' \leq 1$  for  $x\ x'::\text{real}$ 
  proof (rule DERIV-nonpos-imp-decreasing-open [OF  $\langle x \leq x' \rangle$ , where  $f = \lambda x.$ 
 $f1\ x\ y$ ])
    fix  $u :: \text{real}$ 
    assume  $x < u < x'$ 
    with that show  $\exists D. ((\lambda x. f1\ x\ y)$  has-real-derivative  $D$ ) (at  $u$ )  $\wedge D \leq 0$ 
    by - (rule exI conjI Df1 [unfolded df1-def] | simp)+
  next
  show continuous-on  $\{x..x'\}$   $(\lambda x. f1\ x\ y)$ 
  using that by (intro continuous-on-subset [OF continuous-on-f1]) auto
  qed
  have f1f2:  $f2\ x'\ y \leq f1\ x\ y$ 

```

```

if  $x \in \{1/2..1\}$   $x' \in \{1/2..1\}$   $x \leq x'$   $x < 3/4 \wedge x' < 3/4$  for  $x x'::real$ 
using that
apply (simp add: f2-def)
by (smt (verit, best) divide-nonneg-nonneg f1 ln-le-zero-iff pos-prod-lt that)

have  $f2: f2\ x'\ y \leq f2\ x\ y$ 
if  $A: x \in \{1/2..1\}$   $x' \in \{1/2..1\}$   $x \leq x'$  and  $B: \neg x < 3/4$  for  $x x'::real$ 
proof (rule DERIV-nonpos-imp-decreasing-open [OF <x ≤ x'> , where f = λx.
f2 x y])
  fix  $u :: real$ 
  assume  $u: x < u < x'$ 
  have ( $(\lambda x. f2\ x\ y)$  has-real-derivative  $df1\ u + delta\ u$ ) (at u)
    using  $u$  that by (intro Df2) auto
  moreover have  $df1\ u + delta\ u \leq 0$ 
  proof -
    have  $df1\ (1/2) \leq -1/2$ 
      unfolding df1-def by (approximation 20)
    moreover have  $df1\ u \leq df1\ (1/2)$ 
      using  $u$  that unfolding df1-def
      by (intro Transcendental.log-mono) (auto simp: divide-simps)
    moreover have  $delta\ 1 \leq 0.04$ 
      unfolding delta-def by (approximation 4)
    moreover have  $delta\ u \leq delta\ 1$ 
      using  $u$  that by (auto simp: delta-def divide-simps)
    ultimately show ?thesis
      by auto
  qed
  ultimately show  $\exists D. ((\lambda x. f2\ x\ y)$  has-real-derivative  $D$ ) (at u)  $\wedge D \leq 0$ 
    by blast
next
  show continuous-on  $\{x..x'\}$   $(\lambda x. f2\ x\ y)$ 
    unfolding f2-def
    using that by (intro continuous-on-subset [OF continuous-on-f1] continuous-intros)
auto
  qed
  show ?thesis
    using  $f1\ f1f2\ f2$  by (simp add: monotone-on-def ff-def)
qed

```

11.2.2 Claims A.2–A.4

Called simply x in the paper, but are you kidding me?

definition $x\text{-of} \equiv \lambda y::real. 3*y/5 + 0.5454$

lemma $x\text{-of}: x\text{-of} \in \{0..3/4\} \rightarrow \{1/2..1\}$
by (*simp add: x-of-def*)

definition $y\text{-of} \equiv \lambda x::real. 5 * x/3 - 0.909$

lemma *y-of-x-of* [*simp*]: $y\text{-of } (x\text{-of } y) = y$
by (*simp add: x-of-def y-of-def add-divide-distrib*)

lemma *x-of-y-of* [*simp*]: $x\text{-of } (y\text{-of } x) = x$
by (*simp add: x-of-def y-of-def divide-simps*)

lemma *Df1-y* [*derivative-intros*]:
assumes $x < 1$
shows $((\lambda x. f1\ x\ (y\text{-of } x))\ \text{has-real-derivative } 5/3 + df1\ x)\ (at\ x)$
proof –
have $(2 - x * 2) = 2 * (1 - x)$
by *simp*
then have [*simp*]: $\log\ 2\ (2 - x * 2) = \log\ 2\ (1 - x) + 1$
using *log-mult [of 2 1-x 2] assms* **by** (*smt (verit, best) log-eq-one*)
show *?thesis*
using *assms*
unfolding *f1-def y-of-def H-def df1-def*
apply –
apply (*rule derivative-eq-intros refl | simp*)
apply (*simp add: log-divide divide-simps*)
apply (*simp add: algebra-simps*)
done

qed

lemma *Df2-y* [*derivative-intros*]:
assumes $1/2 \leq x < 1$
shows $((\lambda x. f2\ x\ (y\text{-of } x))\ \text{has-real-derivative } 5/3 + df1\ x + delta\ x)\ (at\ x)$
using *assms* **unfolding** *f2-def delta-def*
apply –
apply (*rule derivative-eq-intros Df1 | simp*)
apply (*simp add: divide-simps power2-eq-square*)
done

definition $Dg\text{-}x \equiv \lambda y. 3 * \log\ 2\ (5/3) / 5 + \log\ 2\ ((2727 + y * 8000) / (y * 12500))$
 $- 2727 / (\ln\ 2 * (2727 + y * 8000))$

lemma *Dg-x* [*derivative-intros*]:
assumes $y \in \{0 < .. < 3/4\}$
shows $((\lambda y. gg\ (x\text{-of } y)\ y)\ \text{has-real-derivative } Dg\text{-}x\ y)\ (at\ y)$
using *assms*
unfolding *x-of-def gg-def GG-def Dg-x-def*
apply –
apply (*rule derivative-eq-intros refl | simp*)
apply (*simp add: field-simps*)
done

Claim A2 is difficult because it comes *real close*: max value = 1.999281, when $y = 0.4339$. There is no simple closed form for the maximum point (where the derivative goes to 0).

Due to the singularity at zero, we need to cover the zero case analytically, but at least interval arithmetic covers the maximum point

lemma A2:

assumes $y \in \{0..3/4\}$
shows $gg (x\text{-of } y) y \leq 2 - 1/2^{11}$

proof –

have *?thesis* **if** $y \in \{0..1/10\}$

proof –

have $gg (x\text{-of } y) y \leq gg (x\text{-of } (1/10)) (1/10)$

proof (*rule DERIV-nonneg-imp-increasing-open [of y 1/10]*)

fix $y' :: \text{real}$

assume $y': y < y' y' < 1/10$

then have $y' > 0$

using *that* **by** *auto*

show $\exists D. ((\lambda u. gg (x\text{-of } u) u) \text{ has-real-derivative } D) (at\ y') \wedge 0 \leq D$

proof (*intro exI conjI*)

show $((\lambda u. gg (x\text{-of } u) u) \text{ has-real-derivative } Dg\text{-x } y') (at\ y')$

using y' **that** **by** (*intro derivative-eq-intros*) *auto*

next

define *Num* **where** $Num \equiv 3 * \log 2 (5/3) / 5 * (\ln 2 * (2727 + y' * 8000)) + \log 2 ((2727 + y' * 8000) / (y' * 12500)) * (\ln 2 * (2727 + y' * 8000)) - 2727$

have $A: 835.81 \leq 3 * \log 2 (5/3) / 5 * \ln 2 * 2727$

by (*approximation 25*)

have $B: 2451.9 \leq 3 * \log 2 (5/3) / 5 * \ln 2 * 8000$

by (*approximation 25*)

have $C: Dg\text{-x } y' = Num / (\ln 2 * (2727 + y' * 8000))$

using $\langle y' > 0 \rangle$ **by** (*simp add: Dg-x-def Num-def add-divide-distrib diff-divide-distrib*)

have $0 \leq -1891.19 + \log 2 (2727 / 1250) * (\ln 2 * (2727))$

by (*approximation 6*)

also have $\dots \leq -1891.19 + 2451.9 * y' + \log 2 ((2727 + y' * 8000) / (y' * 12500)) * (\ln 2 * (2727 + y' * 8000))$

using $y' < 0 < y'$

by (*intro add-mono mult-mono Transcendental.log-mono frac-le order.refl*)

auto

also have $\dots = 835.81 + 2451.9 * y' + \log 2 ((2727 + y' * 8000) / (y' * 12500)) * (\ln 2 * (2727 + y' * 8000)) - 2727$

by *simp*

also have $\dots \leq Num$

using *A mult-right-mono [OF B, of y'] <y'>0*

unfolding *Num-def ring-distrib*

by (*intro add-mono diff-mono order.refl*) (*auto simp: mult-ac*)

finally have $Num \geq 0$.

with *C* **show** $0 \leq Dg\text{-x } y'$

using $\langle 0 < y' \rangle$ **by** *auto*

qed

next

let $?f = \lambda x. x * \log 2 ((16*x/5 + 2727/2500) / (5*x))$

```

have †: continuous-on {0..} ?f
proof -
  have continuous (at x within {0..}) ?f
    if  $x \geq 0$  for  $x :: real$ 
  proof (cases  $x = 0$ )
    case True
      have continuous (at-right 0) ?f
        unfolding continuous-within by real-asymp
      thus ?thesis
        using True by (simp add: at-within-Ici-at-right)
    qed (use that in <auto intro!: continuous-intros>)
  thus ?thesis
    by (simp add: continuous-on-eq-continuous-within)
qed
show continuous-on {y..1/10} ( $\lambda y. gg (x-of y) y$ )
  unfolding gg-eq x-of-def using that
  by (force intro: continuous-on-subset [OF †] continuous-intros)
qed (use that in auto)
also have ...  $\leq 2 - 1/2^{11}$ 
  unfolding gg-eq x-of-def by (approximation 10)
finally show ?thesis .
qed
moreover
have ?thesis if  $y \in \{1/10 .. 3/4\}$ 
  using that unfolding gg-eq x-of-def
  by (approximation 24 splitting:  $y = 12$ ) — many thanks to Fabian Immler
ultimately show ?thesis
  by (meson assms atLeastAtMost-iff linear)
qed

lemma A3:
  assumes  $y \in \{0..0.341\}$ 
  shows  $f1 (x-of y) y \leq 2 - 1/2^{11}$ 
proof -
  define D where  $D \equiv \lambda x. 5/3 + df1 x$ 
  define I where  $I \equiv \{0.5454 .. 3/4::real\}$ 
  define x where  $x \equiv x-of y$ 
  then have yeq:  $y = y-of x$ 
    by (metis y-of-x-of)
  have  $x \in \{x-of 0 .. x-of 0.341\}$ 
    using assms by (simp add: x-def x-of-def)
  then have  $x \in I$ 
    by (simp add: x-of-def I-def)
  have D: ( $\lambda x. f1 x (y-of x)$ ) has-real-derivative D x (at x) if  $x \in I$  for x
    using that Df1-y by (force simp: D-def I-def)
  have Dgt0:  $D x \geq 0$  if  $x \in I$  for x
    using that unfolding D-def df1-def I-def by (approximation 10)
  have  $f1 x y = f1 x (y-of x)$ 
    by (simp add: yeq)

```

```

also have ...  $\leq f1 (3/4) (y\text{-of } (3/4))$ 
  using x Dgt0
  by (force simp: I-def intro!: D DERIV-nonneg-imp-nondecreasing [where f =
 $\lambda x. f1 x (y\text{-of } x)$ ])
  also have ...  $< 1.994$ 
  by (simp add: f1-def H-def y-of-def (approximation 50))
  also have ...  $< 2 - 1/2^{11}$ 
  by (approximation 50)
  finally show ?thesis
  using x-def by auto
qed

```

This one also comes close: max value = 1.999271, when $y = 0.4526$. The specified upper bound is 1.99951

```

lemma A4:
  assumes  $y \in \{0.341..3/4\}$ 
  shows  $f2 (x\text{-of } y) y \leq 2 - 1/2^{11}$ 
  unfolding f2-def f1-def x-of-def H-def
  using assms by (approximation 18 splitting: y = 13)

```

```

context P0-min
begin

```

The truly horrible Lemma 12.3

```

lemma 123:
  assumes  $\delta \leq 1 / 2^{11}$ 
  shows (SUP  $x \in \{0..1\}$ ). SUP  $y \in \{0..3/4\}$ . ffGG  $(2/5) x y \leq 2 - \delta$ 
proof -
  have min (ff  $x y$ ) (gg  $x y$ )  $\leq 2 - 1/2^{11}$  if  $x \in \{0..1\}$   $y \in \{0..3/4\}$  for  $x y$ 
  proof (cases  $x \leq x\text{-of } y$ )
    case True
    with that have  $gg x y \leq gg (x\text{-of } y) y$ 
    by (intro gg-increasing) auto
    with A2 that show ?thesis
    by fastforce
  next
  case False
  with that have  $ff x y \leq ff (x\text{-of } y) y$ 
  by (intro monotone-onD [OF antimono-on-ff]) (auto simp: x-of-def)
  also have ...  $\leq 2 - 1/2^{11}$ 
  proof (cases  $x\text{-of } y < 3/4$ )
    case True
    with that have  $f1 (x\text{-of } y) y \leq 2 - 1/2^{11}$ 
    by (intro A3) (auto simp: x-of-def)
    then show ?thesis
    using True ff-def by presburger
  next
  case False

```

```

with that have f2 (x-of y) y ≤ 2 - 1/2^11
  by (intro A4) (auto simp: x-of-def)
then show ?thesis
  using False ff-def by presburger
qed
finally show ?thesis
  by linarith
qed
moreover have 2 - 1/2^11 ≤ 2 - δ
  using assms by auto
ultimately show ?thesis
  by (fastforce simp: fFGG-def gg-def intro!: cSUP-least)
qed
end

```

11.3 Concluding the proof

we subtract a tiny bit, as we seem to need this gap

definition $\delta::real$ **where** $\delta \equiv 1 / 2^{11} - 1 / 2^{18}$

lemma *Aux-1-1*:

```

assumes p0-min12: p0-min ≤ 1/2
shows ∀∞k. log 2 (RN k k) / k ≤ 2 - delta'
proof -
  define p0-min::real where p0-min ≡ 1/2
  interpret P0-min p0-min
  proof qed (auto simp: p0-min-def)
  define δ::real where δ ≡ 1 / 2^11
  define η::real where η ≡ 1 / 2^18
  have η: 0 < η η ≤ 1/12
    by (auto simp: η-def)
  define μ::real where μ ≡ 2/5
  have ∀∞k. Big-From-11-1 η μ k
    unfolding μ-def using η by (intro Big-From-11-1) auto
  moreover have log 2 (real (RN k k)) / k ≤ 2 - δ + η if Big-From-11-1 η μ k
for k
  proof -
    have *: (⊔ y∈{0..3/4}. fFGG μ x y + η) = (⊔ y∈{0..3/4}. fFGG μ x y) + η
      if x ≤ 1 for x
      using bdd-above-ff-GG [OF that, of 3/4 μ 0]
      by (simp add: add.commute [of - η] Sup-add-eq)
    have log 2 (RN k k) / k ≤ (SUP x ∈ {0..1}. SUP y ∈ {0..3/4}. fFGG μ x y
  + η)
      using that p0-min12 η μ-def
      by (intro From-11-1) (auto simp: p0-min-def)
    also have ... ≤ (SUP x ∈ {0..1}. (SUP y ∈ {0..3/4}. fFGG μ x y) + η)
  proof (intro cSUP-subset-mono bdd-above.I2 [where M = 4+η])
    fix x :: real

```

```

assume  $x: x \in \{0..1\}$ 
have  $(\bigsqcup y \in \{0..3/4\}. \text{ffGG } \mu x y + \eta) \leq 4 + \eta$ 
  using bdd-above-ff-GG ff-GG-bound x by (simp add: cSup-le-iff)
with *  $x$  show  $(\bigsqcup y \in \{0..3/4\}. \text{ffGG } \mu x y) + \eta \leq 4 + \eta$ 
  by simp
qed (use * in auto)
also have  $\dots = (\text{SUP } x \in \{0..1\}. \text{SUP } y \in \{0..3/4\}. \text{ffGG } \mu x y) + \eta$ 
  using bdd-above-SUP-ff-GG [of 3/4 μ 0]
  by (simp add: add.commute [of - η] Sup-add-eq)
also have  $\dots \leq 2 - \delta + \eta$ 
  using 123 [of 1 / 2^11]
  unfolding  $\delta\text{-def ffGG-def}$  by (auto simp: δ-def ffGG-def μ-def)
finally show ?thesis .
qed
ultimately have  $\forall^\infty k. \log 2 (RN k k) / k \leq 2 - \delta + \eta$ 
  by (metis (lifting) eventually-mono)
then show ?thesis
  by (simp add: δ-def η-def delta'-def)
qed

```

Main theorem 1.1: the exponent is approximately 3.9987

theorem *Main-1-1*:

obtains $\varepsilon::\text{real}$ **where** $\varepsilon > 0 \ \forall^\infty k. RN k k \leq (4 - \varepsilon) ^ k$

proof

let $? \varepsilon = 0.00134::\text{real}$

have $\forall^\infty k. k > 0 \wedge \log 2 (RN k k) / k \leq 2 - \text{delta}'$

unfolding *eventually-conj-iff* **using** *Aux-1-1 eventually-gt-at-top* **by** *blast*

then have $\forall^\infty k. RN k k \leq (2 \text{ powr } (2 - \text{delta}')) ^ k$

proof (*eventually-elim*)

case (*elim k*)

then have $\log 2 (RN k k) \leq (2 - \text{delta}') * k$

by (*meson of-nat-0-less-iff pos-divide-le-eq*)

then have $RN k k \leq 2 \text{ powr } ((2 - \text{delta}') * k)$

by (*smt (verit, best) Transcendental.log-le-iff powr-ge-zero*)

then show $RN k k \leq (2 \text{ powr } (2 - \text{delta}')) ^ k$

by (*simp add: mult.commute powr-power*)

qed

moreover have $2 \text{ powr } (2 - \text{delta}') \leq 4 - ? \varepsilon$

unfolding *delta'-def* **by** (*approximation 25*)

ultimately show $\forall^\infty k. \text{real } (RN k k) \leq (4 - ? \varepsilon) ^ k$

by (*smt (verit) power-mono powr-ge-zero eventually-mono*)

qed *auto*

end

References

- [1] M. Campos, S. Griffiths, R. Morris, and J. Sahasrabudhe. An exponential improvement for diagonal Ramsey, 2023. arXiv, 2303.09521.