

The Balog–Szemerédi–Gowers Theorem

Angeliki Koutsoukou-Argyraiki, Mantas Bakšys, and Chelsea Edmonds
University of Cambridge
{ak2110, mb2412, cle47}@cam.ac.uk

February 6, 2026

Abstract

We formalise the Balog–Szemerédi–Gowers Theorem, a profound result in additive combinatorics which played a central role in Gowers’s proof deriving the first effective bounds for Szemerédi’s Theorem [2]. The proof is of great mathematical interest given that it involves an interplay between different mathematical areas, namely applications of graph theory and probability theory to additive combinatorics involving algebraic objects. This interplay is what made the process of the formalisation, for which we had to develop formalisations of new background material in the aforementioned areas, more rich and technically challenging. We demonstrate how locales, Isabelle’s module system, can be employed to handle such interplays. To treat the graph-theoretic aspects of the proof, we make use of a new, more general undirected graph theory library developed recently by Chelsea Edmonds, which is both flexible and extensible [1]. For the formalisation we followed a proof presented in the 2022 lecture notes by Timothy Gowers "Introduction to Additive Combinatorics" for Part III of the Mathematical Tripos taught at the University of Cambridge [3]. In addition to the main theorem, which, following our source, is formulated for difference sets, we also give an alternative version for sumsets which required a formalisation of an auxiliary triangle inequality following a proof by Yufei Zhao from his book "Graph Theory and Additive Combinatorics" [4]. We moreover formalise a few additional results in additive combinatorics that are not used in the proof of the main theorem. This is the first formalisation of the Balog–Szemerédi–Gowers Theorem in any proof assistant to our knowledge.

Contents

1	Miscellaneous technical lemmas	3
2	Background material for the graph-theoretic aspects of the main proof	5
2.1	On graphs with loops	5
2.2	On bipartite graphs	6
3	Auxiliary probability space results	10
4	A triangle inequality for sumsets	13
5	Background material in additive combinatorics	19
5.1	Additive quadruples and additive energy	20
5.2	On sums	21
5.3	On differences	25
6	Results on lower bounds on additive energy	30
7	Towards the proof of the Balog–Szemerédi–Gowers Theorem	33
8	Supplementary results related to intermediate lemmas used in the proof of the Balog–Szemerédi–Gowers Theorem	65

Acknowledgements

Angeliki Koutsoukou-Argyraiki is funded by the ERC Advanced Grant ALEXANDRIA (Project GA 742178) funded by the European Research Council and led by Lawrence C. Paulson (University of Cambridge, Department of Computer Science and Technology). Mantas Bakšys received funding for his internship supervised by Koutsoukou-Argyraiki by the Cambridge Mathematics Placements (CMP) Programme and by the ALEXANDRIA Project. Chelsea Edmonds is jointly funded by the Cambridge Trust (Cambridge Australia Scholarship) and a Cambridge Department of Computer Science and Technology Premium Research Studentship.

1 Miscellaneous technical lemmas

theory *Miscellaneous-Lemmas*

imports

HOL-Library.Indicator-Function

HOL-Analysis.Convex

begin

lemma *set-pairs-filter-subset*: $A \subseteq B \implies \{p . p \in A \times A \wedge P p\} \subseteq \{p . p \in B \times B \wedge P p\}$

by (*intro subsetI*) *blast*

lemma *card-set-ss-indicator*:

assumes $A \subseteq B$

assumes *finite B*

shows $\text{card } A = (\sum p \in B. \text{indicator } A p)$

proof –

obtain *C* **where** *ceq*: $C = B - A$ **by** *blast*

then have *beq*: $B = A \cup C$ **using** *assms* **by** *blast*

have *bint*: $A \cap C = \{\}$ **using** *ceq* **by** *blast*

have *finite*: *finite A* **using** *assms* *finite-subset* **by** *auto*

have *zero*: $\bigwedge p. p \in C \implies \text{indicator } (A) p = 0$

by (*simp add: ceq*)

then have $\text{card } A = (\sum p \in A. \text{indicator } (A) p)$

by *simp*

also have $\dots = (\sum p \in A. \text{indicator } A p) + (\sum p \in C. \text{indicator } A p)$

using *zero* **by** (*metis add-cancel-left-right sum.neutral*)

finally show $\text{card } A = (\sum p \in B. \text{indicator } A p)$ **using** *beq bint assms*

by (*metis add.commute ceq sum.subset-diff*)

qed

lemma *card-cartesian-prod-square*: $\text{finite } X \implies \text{card } (X \times X) = (\text{card } X)^{\wedge 2}$

using *card-cartesian-product* **by** (*simp add: power2-eq-square*)

lemma (*in ordered-ab-group-add*) *diff-strict1-mono*:

assumes $a > a' \ b \leq b'$

shows $a - b > a' - b'$

using *diff-strict-mono assms*

by (*metis local.diff-strict-right-mono local.dual-order.not-eq-order-implies-strict*)

lemma *card-cartesian-product-6*: $\text{card } (A \times A \times A \times A \times A \times A) = (\text{card } A)^{\wedge 6}$

proof –

have $\text{card } (A \times A \times A \times A \times A \times A) =$

$\text{card } A * \text{card } A * \text{card } A * \text{card } A * \text{card } A * \text{card } A$

using *card-cartesian-product mult.commute* **by** *metis*

then show *?thesis* **by** *algebra*

qed

lemma *card-cartesian-product3*: $\text{card } (X \times Y \times Z) = \text{card } X * \text{card } Y * \text{card } Z$
using *card-cartesian-product* **by** (*metis mult.commute mult.left-commute*)

lemma *card-le-image-div*:

fixes *A*:: 'a set **and** *B*:: 'b set **and** *f*:: 'a \Rightarrow 'b set **and** *r*:: real
assumes *finite B* **and** *pairwise* ($\lambda s t. \text{disjnt } (f s) (f t)$) *A* **and** $\forall d \in A. (\text{card } (f d)) \geq r$
and $\forall d \in A. f d \subseteq B$ **and** $r > 0$
shows $\text{card } A \leq \text{card } B / r$

proof (*cases finite A*)

assume *hA*: *finite A*

have *hpair-disj*: *pairwise disjnt* (*f* ' *A*) **using** *assms* **by** (*metis pairwiseD pairwise-imageI*)

have $r * \text{card } A = (\sum d \in A. r)$ **by** *simp*

also have $\dots \leq (\sum d \in A. \text{card } (f d))$ **using** *assms sum-mono* **by** *fastforce*

also have $\dots = \text{sum card } (f ' A)$ **using** *assms hA* **by** (*simp add: sum-card-image*)

also have $\dots = \text{card } (\bigcup d \in A. f d)$ **using** *assms hA hpair-disj*

by (*metis Sup-upper card-Union-disjoint finite-UN-I rev-finite-subset*)

also have $\dots \leq \text{card } B$ **using** *assms card-mono*

by (*metis UN-subset-iff of-nat-le-iff*)

finally have $r * \text{card } A \leq \text{card } B$ **by** *linarith*

thus *?thesis* **using** *divide-le-eq assms* **by** (*simp add: mult-imp-le-div-pos mult-of-nat-commute*)

next

assume $\neg \text{finite } A$

thus *?thesis* **using** *assms* **by** *auto*

qed

lemma *list-middle-eq*:

$\text{length } xs = \text{length } ys \Longrightarrow \text{hd } xs = \text{hd } ys \Longrightarrow \text{last } xs = \text{last } ys$

$\Longrightarrow \text{butlast } (tl xs) = \text{butlast } (tl ys) \Longrightarrow xs = ys$

apply (*induct xs ys rule: list-induct2, simp*)

by (*metis append-butlast-last-id butlast.simps(1) butlast.simps(2) butlast-tl hd-Cons-tl*)

impossible-Cons le-refl list.sel(3) neq-Nil-conv)

lemma *list2-middle-singleton*:

assumes $\text{length } xs = 3$

shows $\text{butlast } (tl xs) = [xs ! 1]$

proof (*simp add: list-eq-iff-nth-eq assms*)

have *l*: $\text{length } (\text{butlast } (tl xs)) = 1$ **using** *length-butlast length-tl assms* **by** *simp*

then have $\text{butlast } (tl xs) ! 0 = (tl xs) ! 0$ **using** *nth-butlast[of 0 tl xs]* **by** *simp*

then show $\text{butlast } (tl xs) ! 0 = xs ! \text{Suc } 0$ **using** *nth-tl[of 0 xs] l* **by** *simp*

qed

lemma *le-powr-half-mult*:

```

fixes  $x\ y\ z::\ \text{real}$ 
assumes  $x^2 \leq y * z$  and  $0 \leq y$  and  $0 \leq z$ 
shows  $x \leq y^{1/2} * z^{1/2}$ 
using assms power2-eq-square
by (metis dual-order.trans linorder-linear powr-ge-zero powr-half-sqrt powr-mult
real-le-rsqrt
real-sqrt-le-0-iff)

```

```

lemma Cauchy-Schwarz-ineq-sum2:
fixes  $f\ g::\ 'a \Rightarrow \text{real}$  and  $A::\ 'a\ \text{set}$ 
shows  $(\sum d \in A. f\ d * g\ d) \leq$ 
 $(\sum d \in A. (f\ d)^2)^{1/2} * (\sum d \in A. (g\ d)^2)^{1/2}$ 
using Convex.Cauchy-Schwarz-ineq-sum[of f g A] le-powr-half-mult sum-nonneg
zero-le-power2
by (metis (mono-tags, lifting))

```

end

2 Background material for the graph-theoretic aspects of the main proof

This section includes a number of lemmas on project specific definitions for graph theory, building on the general undirected graph theory library [1]

```

theory Graph-Theory-Preliminaries
imports
  Miscellaneous-Lemmas
  Undirected-Graph-Theory.Bipartite-Graphs
  Undirected-Graph-Theory.Connectivity
  Random-Graph-Subgraph-Threshold.Ugraph-Misc
begin

```

2.1 On graphs with loops

```

context ulgraph

```

```

begin

```

```

definition degree-normalized::  $'a \Rightarrow 'a\ \text{set} \Rightarrow \text{real}$  where
  degree-normalized  $v\ S \equiv \text{card}(\text{neighbors-ss } v\ S) / (\text{card } S)$ 

```

```

lemma degree-normalized-le-1:  $\text{degree-normalized } x\ S \leq 1$ 

```

```

proof(cases finite S)
assume  $hA: \text{finite } S$ 
then have  $\text{card}(\text{neighbors-ss } x\ S) \leq \text{card } S$  using neighbors-ss-def card-mono
 $hA$ 
by fastforce
then show ?thesis using degree-normalized-def divide-le-eq-1

```

by (*metis antisym-conv3 of-nat-le-iff of-nat-less-0-iff*)
 next
 case *False*
 then show *?thesis using degree-normalized-def by auto*
 qed
 end

2.2 On bipartite graphs

context *bipartite-graph*
 begin

definition *codegree*:: 'a \Rightarrow 'a \Rightarrow nat **where**
codegree v u \equiv card {x \in V . vert-adj v x \wedge vert-adj u x}

lemma *codegree-neighbors*: *codegree v u* = card (*neighborhood v* \cap *neighborhood u*)
unfolding *codegree-def neighborhood-def*

proof –

have {x \in V . vert-adj v x \wedge vert-adj u x} = {va \in V . vert-adj v va} \cap {v \in V .
 vert-adj u v}

by *blast*

thus card {x \in V . vert-adj v x \wedge vert-adj u x} = card ({va \in V . vert-adj v va}
 \cap {v \in V . vert-adj u v})

by *auto*

qed

lemma *codegree-sym*: *codegree v u* = *codegree u v*
 by (*simp add: Int-commute codegree-neighbors*)

definition *codegree-normalized*:: 'a \Rightarrow 'a \Rightarrow 'a set \Rightarrow real **where**
codegree-normalized v u S \equiv *codegree v u* / card *S*

lemma *codegree-normalized-altX*:

assumes x \in X **and** x' \in X

shows *codegree-normalized x x' Y* = card (*neighbors-ss x Y* \cap *neighbors-ss x' Y*)
 / card *Y*

proof –

have ((*neighbors-ss x Y*) \cap (*neighbors-ss x' Y*)) = *neighborhood x* \cap *neighborhood*
x'

using *neighbors-ss-eq-neighborhoodX* *assms* **by** *auto*

then show *?thesis unfolding codegree-normalized-def*

using *codegree-def codegree-neighbors* **by** *presburger*

qed

lemma *codegree-normalized-altY*:

assumes $y \in Y$ **and** $y' \in Y$
shows $\text{codegree-normalized } y \ y' \ X = \text{card } (\text{neighbors-ss } y \ X \cap \text{neighbors-ss } y' \ X)$
 $/ \text{card } X$

proof –

have $\text{neighbors-ss } y \ X \cap \text{neighbors-ss } y' \ X = \text{neighborhood } y \cap \text{neighborhood } y'$
using $\text{neighbors-ss-eq-neighborhood } Y \ \text{assms}$ **by** auto
then show $?thesis$ **unfolding** $\text{codegree-normalized-def}$
using $\text{codegree-def } \text{codegree-neighbors}$ **by** presburger
qed

lemma $\text{codegree-normalized-sym}$: $\text{codegree-normalized } u \ v \ S = \text{codegree-normalized } v \ u \ S$

unfolding $\text{codegree-normalized-def}$ **using** codegree-sym **by** simp

definition bad-pair :: $'a \Rightarrow 'a \Rightarrow 'a \ \text{set} \Rightarrow \text{real} \Rightarrow \text{bool}$ **where**

$\text{bad-pair } v \ u \ S \ c \equiv \text{codegree-normalized } v \ u \ S < c$

lemma bad-pair-sym :

assumes $\text{bad-pair } v \ u \ S \ c$ **shows** $\text{bad-pair } u \ v \ S \ c$

using $\text{assms } \text{bad-pair-def } \text{codegree-normalized-def}$

by $(\text{simp } \text{add: } \text{codegree-normalized-sym})$

definition bad-pair-set :: $'a \ \text{set} \Rightarrow 'a \ \text{set} \Rightarrow \text{real} \Rightarrow ('a \times 'a) \ \text{set}$ **where**

$\text{bad-pair-set } S \ T \ c \equiv \{(u, v) \in S \times S. \text{bad-pair } u \ v \ T \ c\}$

lemma bad-pair-set-ss : $\text{bad-pair-set } S \ T \ c \subseteq S \times S$

by $(\text{auto } \text{simp } \text{add: } \text{bad-pair-set-def})$

lemma $\text{bad-pair-set-filter-alt}$:

$\text{bad-pair-set } S \ T \ c = \text{Set.filter } (\lambda p . \text{bad-pair } (\text{fst } p) (\text{snd } p) \ T \ c) (S \times S)$

using bad-pair-set-def **by** auto

lemma $\text{bad-pair-set-finite}$:

assumes $\text{finite } S$

shows $\text{finite } (\text{bad-pair-set } S \ T \ c)$

proof –

have $\text{finite } (S \times S)$ **using** $\text{finite-cartesian-product } \text{assms}$ **by** blast

thus $?thesis$ **using** $\text{bad-pair-set-filter-alt } \text{finite-filter}$ **by** auto

qed

lemma $\text{codegree-is-path-length-two}$:

$\text{codegree } x \ x' = \text{card } \{p . \text{connecting-path } x \ x' \ p \wedge \text{walk-length } p = 2\}$

unfolding codegree-def

proof –

define f :: $'a \ \text{list} \Rightarrow 'a$ **where** $f = (\lambda p . p!1)$

have $f\text{-inj}$: $\text{inj-on } f \ \{p . \text{connecting-path } x \ x' \ p \wedge \text{walk-length } p = 2\}$

unfolding $f\text{-def}$

proof $(\text{intro } \text{inj-onI}, \text{simp } \text{del: } \text{One-nat-def})$

fix $a\ b$ **assume** $ha: \text{connecting-path } x\ x'\ a \wedge \text{walk-length } a = 2$ **and**
 $hb: \text{connecting-path } x\ x'\ b \wedge \text{walk-length } b = 2$ **and** $1: a!1 = b!1$
then have $len: \text{length } a = 3\ \text{length } b = 3$ **using** walk-length-conv **by** auto
show $a = b$ **using** $\text{list2-middle-singleton } 1\ len\ \text{list-middle-eq } ha\ hb\ \text{connecting-path-def } len$ **by** metis
qed
have $f\text{-image}: f\ \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\} =$
 $\{xa \in V. \text{vert-adj } x\ xa \wedge \text{vert-adj } x'\ xa\}$
proof ($\text{intro subset-antisym}$)
show $f\ \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\}$
 $\subseteq \{xa \in V. \text{vert-adj } x\ xa \wedge \text{vert-adj } x'\ xa\}$
proof (intro subsetI)
fix a **assume** $a \in f\ \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\}$
then obtain p **where** $ha: p!1 = a$ **and** $hp: \text{connecting-path } x\ x'\ p$ **and** $hpl:$
 $\text{length } p = 3$
using $f\text{-def } \text{walk-length-conv}$ **by** auto
have $p!0 = x$ **using** $hd\text{-conv-nth}[of\ p]\ hpl\ hp\ \text{connecting-path-def}$ **by** fastforce

then have $va1: \text{vert-adj } x\ a$ **using** $\text{is-walk-index}[of\ 0\ p]\ hp\ \text{connecting-path-def}$
 is-gen-path-def
 $\text{vert-adj-def } ha\ hpl$ **by** auto
have $p!2 = x'$ **using** $\text{last-conv-nth}[of\ p]\ hpl\ hp\ \text{connecting-path-def}$ **by**
 fastforce
then have $\text{vert-adj } a\ x'$ **using** $\text{is-walk-index}[of\ 1\ p]\ hp\ \text{connecting-path-def}$
 is-gen-path-def
 $\text{vert-adj-def } ha\ hpl$ **by** ($\text{metis One-nat-def } le0\ lessI\ numeral-3\ eq-3$
 one-add-one)
then show $a \in \{a \in V. \text{vert-adj } x\ a \wedge \text{vert-adj } x'\ a\}$
using $va1\ \text{vert-adj-sym}$ **by** ($\text{simp add: vert-adj-imp-in } V$)
qed
show $\{xa \in V. \text{vert-adj } x\ xa \wedge \text{vert-adj } x'\ xa\}$
 $\subseteq f\ \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\}$
proof (intro subsetI)
fix a **assume** $ha: a \in \{xa \in V. \text{vert-adj } x\ xa \wedge \text{vert-adj } x'\ xa\}$
then have $a \in V$ **and** $x \in V$ **and** $x' \in V$ **and** $\text{vert-adj } x\ a$ **and** $\text{vert-adj } x'\ a$
using $\text{vert-adj-imp-in } V$ **by** auto
then have $\text{is-gen-path } [x, a, x']$
using $\text{is-walk-def } \text{vert-adj-def } \text{vert-adj-sym } ha\ \text{singleton-not-edge } \text{is-gen-path-def}$
by auto
then have $\text{connecting-path } x\ x'\ [x, a, x']$
unfolding $\text{connecting-path-def } \text{vert-adj-def } hd\text{-conv-nth } \text{last-conv-nth}$ **by**
 simp
moreover have $\text{walk-length } [x, a, x'] = 2$ **using** walk-length-conv **by** simp
ultimately show $a \in f\ \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\}$
using $f\text{-def}$ **by** force
qed
qed
then show $\text{card } \{xa \in V. \text{vert-adj } x\ xa \wedge \text{vert-adj } x'\ xa\} =$
 $\text{card } \{p . \text{connecting-path } x\ x'\ p \wedge \text{walk-length } p = 2\}$

using *f-inj card-image* **by** *fastforce*
qed

lemma *codegree-bipartite-eq*:

$\forall x \in X. \forall x' \in X. \text{codegree } x \ x' = \text{card } \{y \in Y. \text{vert-adj } x \ y \wedge \text{vert-adj } x' \ y\}$
unfolding *codegree-def* **using** *vert-adj-imp-inV X-vert-adj-Y*
by (*metis (no-types, lifting) Collect-cong*)

lemma (**in** *fin-bipartite-graph*) *bipartite-deg-square-eq*:

$\forall y \in Y. (\sum x' \in X. \sum x \in X. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y) = (\text{degree } y)^2$

proof

have *hX*: *finite X* **by** (*simp add: partitions-finite(1)*)
fix *y* **assume** *hy*: $y \in Y$
have *1*: $\forall x' \in X. \forall x \in X. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y = \text{indicator } (\{z. \text{vert-adj } x' \ z\} \cap \{z. \text{vert-adj } x \ z\}) \ y$
by (*metis (mono-tags, lifting) Int-Collect indicator-simps(1) indicator-simps(2) mem-Collect-eq*)
have *2*: $\forall x' \in X. \forall x \in X. (\text{indicator } (\{z. \text{vert-adj } x' \ z\} \cap \{z. \text{vert-adj } x \ z\}) \ y :: \text{nat}) = \text{indicator } \{z. \text{vert-adj } x' \ z\} \ y * \text{indicator } \{z. \text{vert-adj } x \ z\} \ y$ **using** *indicator-inter-arith*
by *auto*
have $(\sum x' \in X. \sum x \in X. (\text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y :: \text{nat})) =$
 $(\sum x' \in X. \sum x \in X. \text{indicator } (\{z. \text{vert-adj } x' \ z\} \cap \{z. \text{vert-adj } x \ z\}) \ y)$
using *1 sum.cong* **by** (*metis (no-types, lifting)*)
also have $\dots = (\sum x' \in X. \sum x \in X. \text{indicator } \{z. \text{vert-adj } x' \ z\} \ y * \text{indicator } \{z. \text{vert-adj } x \ z\} \ y)$ **using** *2 sum.cong* **by** *auto*
also have $\dots = \text{sum } (\lambda x. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y) \ X * \text{sum } (\lambda x. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y) \ X$
using *sum-product[of* $(\lambda x. (\text{indicator } \{z. \text{vert-adj } x \ z\} \ y :: \text{nat})) \ X$
 $(\lambda x. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y) \ X]$ **by** *auto*
finally have *3*: $(\sum x' \in X. \sum x \in X. (\text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y :: \text{nat})) = (\text{sum } (\lambda x. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y) \ X)^2$ **using** *power2-eq-square*
by (*metis (no-types, lifting)*)
have $\forall x \in X. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y = \text{indicator } \{x. \text{vert-adj } x \ y\} \ x$
by (*simp add: indicator-def*)
from this have $(\text{sum } (\lambda x. \text{indicator } \{z. \text{vert-adj } x \ z\} \ y) \ X) = \text{sum } (\lambda x. \text{indicator } \{x. \text{vert-adj } x \ y\} \ x) \ X$
using *sum.cong* **by** *fastforce*
also have $\dots = \text{card } (\{x \in X. \text{vert-adj } x \ y\})$ **using** *sum-indicator-eq-card hX*
by (*metis Collect-conj-eq Collect-mem-eq*)
finally show $(\sum x' \in X. \sum x \in X. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y) = (\text{degree } y)^2$
using *3 hy degree-neighbors-ssY neighbors-ss-def vert-adj-sym* **by** *presburger*
qed

lemma (in *fin-bipartite-graph*) *codegree-degree*:
 $(\sum x' \in X. \sum x \in X. (\text{codegree } x \ x')) = (\sum y \in Y. (\text{degree } y)^{\wedge 2})$

proof –

have *hX: finite X and hY: finite Y*
by (*simp-all add: partitions-finite*)
have $\forall x' \in X. \forall x \in X. \{z \in V. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} = Y \cap \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\}$
using *XY-union X-vert-adj-Y* **by** *fastforce*
from *this* **have** $(\sum x' \in X. \sum x \in X. (\text{codegree } x \ x')) = (\sum x' \in X. \sum x \in X. \text{card } (Y \cap \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\}))$
using *codegree-def sum.cong* **by** *auto*
also **have** $\dots = (\sum x' \in X. \sum x \in X. \sum y \in Y. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y)$
using *sum-indicator-eq-card hY* **by** *fastforce*
also **have** $\dots = (\sum x' \in X. \sum y \in Y. (\sum x \in X. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y))$
using *sum.swap* **by** (*metis (no-types)*)
also **have** $\dots = (\sum y \in Y. \sum x' \in X. (\sum x \in X. \text{indicator } \{z. \text{vert-adj } x \ z \wedge \text{vert-adj } x' \ z\} \ y))$
using *sum.swap* **by** *fastforce*
also **have** $\dots = (\sum y \in Y. (\text{degree } y)^{\wedge 2})$ **using** *bipartite-deg-square-eq sum.cong*
by *force*
finally **show** *?thesis* **by** *simp*
qed

lemma (in *fin-bipartite-graph*) *sum-degree-normalized-X-density*:
 $(\sum x \in X. \text{degree-normalized } x \ Y) / \text{card } X = \text{edge-density } X \ Y$
by (*smt (z3) card-all-edges-betw-neighbor card-edges-between-set degree-normalized-def divide-divide-eq-left' density-simp of-nat-mult of-nat-sum partitions-finite(1) partitions-finite(2) sum.cong sum-left-div-distrib*)

lemma (in *fin-bipartite-graph*) *sum-degree-normalized-Y-density*:
 $(\sum y \in Y. \text{degree-normalized } y \ X) / \text{card } Y = \text{edge-density } X \ Y$
using *bipartite-sym fin-bipartite-graph.sum-degree-normalized-X-density fin-bipartite-graph-def*

fin-graph-system-axioms edge-density-commute **by** *fastforce*

end

end

3 Auxiliary probability space results

theory *Prob-Space-Lemmas*

imports

Random-Graph-Subgraph-Threshold.Prob-Lemmas

begin

context *prob-space*

begin

lemma *expectation-uniform-count*:

assumes $M = \text{uniform-count-measure } X$ **and** *finite* X

shows $\text{expectation } f = (\sum x \in X. f x) / \text{card } X$

proof –

have $\text{expectation } f = (\sum x \in X. (1 / (\text{card } X)) * f x)$

using *assms uniform-count-measure-def bot-nat-0.extremum of-nat-0 of-nat-le-iff real-scaleR-def*

lebesgue-integral-point-measure-finite[of - $(\lambda x. 1 / \text{card } X) f]$

scaleR-sum-right sum-distrib-left zero-le-divide-1-iff **by** *metis*

then show *?thesis* **using** *sum-left-div-distrib* **by** *fastforce*

qed

A lemma to obtain a value for x where the inequality is satisfied

lemma *expectation-obtains-ge*:

fixes $f :: 'a \Rightarrow \text{real}$

assumes $M = \text{uniform-count-measure } X$ **and** *finite* X

assumes $\text{expectation } f \geq c$

obtains x **where** $x \in X$ **and** $f x \geq c$

proof –

have $ne: X \neq \{\}$

using *assms(1) subprob-not-empty* **by** *auto*

then have $ne0: \text{card } X > 0$

by (*simp add: assms(2) card-gt-0-iff*)

have $\exists x \in X. f x \geq c$

proof (*rule ccontr*)

assume $\neg (\exists x \in X. c \leq f x)$

then have $\forall x \in X. c > f x$ **by** *auto*

then have $(\sum x \in X. f x) < (\sum x \in X. c)$

by (*meson assms(2) ne sum-strict-mono*)

then have $lt: (\sum x \in X. f x) < (\text{card } X) * c$ **by** *simp*

have $\text{expectation } f = (\sum x \in X. f x) / \text{card } X$ **using** *expectation-uniform-count assms* **by** *auto*

then have $(\sum x \in X. f x) \geq (\text{card } X) * c$ **using** $ne0$ *assms*

by (*simp add: le-divide-eq mult-of-nat-commute*)

then show *False* **using** lt **by** *auto*

qed

then show *?thesis* **using** *that* **by** *auto*

qed

The following is the variation on the Cauchy-Schwarz inequality presented in Gowers's notes before Lemma 2.13 [3].

lemma *cauchy-schwarz-ineq-var*:

fixes $X :: 'a \Rightarrow \text{real}$

assumes *integrable* $M (\lambda x. (X x) ^ 2)$ **and** $X \in \text{borel-measurable } M$

shows $\text{expectation } (\lambda x. (X x)^{\wedge 2}) \geq (\text{expectation } (\lambda x. (X x)))^{\wedge 2}$

proof –

have $\text{expectation } (\lambda x. (X x)^{\wedge 2}) - (\text{expectation } (\lambda x. (X x)))^{\wedge 2} = \text{expectation } (\lambda x. (X x - \text{expectation } X)^{\wedge 2})$

using *variance-expectation* *assms(1)* *assms(2)* **by** *presburger*

then have $\text{expectation } (\lambda x. (X x)^{\wedge 2}) - (\text{expectation } (\lambda x. (X x)))^{\wedge 2} \geq 0$ **by** *simp*

thus *?thesis* **by** *simp*

qed

lemma *integrable-uniform-count-measure-finite*:

fixes $g :: 'a \Rightarrow 'b :: \{\text{banach, second-countable-topology}\}$

shows $\text{finite } A \implies \text{integrable } (\text{uniform-count-measure } A) g$

unfolding *uniform-count-measure-def* **by** (*simp add: integrable-point-measure-finite*)

lemma *cauchy-schwarz-ineq-var-uniform*:

fixes $X :: 'a \Rightarrow \text{real}$

assumes $M = \text{uniform-count-measure } S$

assumes *finite S*

shows $\text{expectation } (\lambda x. (X x)^{\wedge 2}) \geq (\text{expectation } (\lambda x. (X x)))^{\wedge 2}$

proof –

have *borel: X ∈ borel-measurable M* **using** *assms* **by** (*simp*)

have *integrable M X* **using** *assms* **by** (*simp add: integrable-uniform-count-measure-finite*)

then have *integrable M (λ x. (X x)²)* **using** *assms* **by** (*simp add: integrable-uniform-count-measure-finite*)

thus *?thesis* **using** *cauchy-schwarz-ineq-var borel* **by** *simp*

qed

An equation for expectation over a discrete random variables distribution:

lemma *expectation-finite-uniform-space*:

assumes $M = \text{uniform-count-measure } S$ **and** *finite S*

fixes $X :: 'a \Rightarrow \text{real}$

shows $\text{expectation } X = (\sum y \in X \text{ ' } S . \text{prob } \{x \in S . X x = y\} * y)$

proof –

have *Bochner-Integration.simple-bochner-integrable M X*

proof (*safe intro!*: *Bochner-Integration.simple-bochner-integrable.intros*)

show *simple-function M X* **unfolding** *simple-function-def* **using** *assms*

by (*auto simp add: space-uniform-count-measure*)

show *emeasure M {y ∈ space M. X y ≠ 0} = ∞* \implies *False*

using *emeasure-subprob-space-less-top* **by** (*auto*)

qed

then have $\text{expectation } X = \text{Bochner-Integration.simple-bochner-integral } M X$

using *simple-bochner-integrable-eq-integral* **by** *fastforce*

thus *?thesis* **using** *Bochner-Integration.simple-bochner-integral-def space-uniform-count-measure* **by** (*metis (no-types, lifting) Collect-cong assms(1) real-scaleR-def sum.cong*)

qed

lemma *expectation-finite-uniform-indicator*:

assumes $M = \text{uniform-count-measure } S$ **and** *finite* S

shows *expectation* $(\lambda x. \text{indicator } (T x) y) = \text{prob } \{x \in S . \text{indicator } (T x) y = 1\}$ (*is expectation* $?X = -$)

proof –

have *ss*: $?X \text{ ' } S \subseteq \{0, 1\}$

by (*intro subsetI, auto simp add: indicator-eq-1-iff*)

have *diff*: $\bigwedge y'. y' \in (\{0, 1\} - ?X \text{ ' } S) \implies \text{prob } \{x \in S . ?X x = y'\} = 0$

by (*metis (mono-tags, lifting) DiffD2 empty-Collect-eq image-eqI measure-empty*)

have *expectation* $?X = (\sum y \in ?X \text{ ' } S . \text{prob } \{x \in S . ?X x = y\} * y)$

using *expectation-finite-uniform-space assms* **by** *auto*

also have $\dots = (\sum y \in ?X \text{ ' } S . \text{prob } \{x \in S . ?X x = y\} * y) +$

$(\sum y \in (\{0, 1\} - ?X \text{ ' } S) . \text{prob } \{x \in S . ?X x = y\} * y)$

using *diff* **by** *auto*

also have $\dots = (\sum y \in \{0, 1\} . \text{prob } \{x \in S . ?X x = y\} * y)$

using *sum.subset-diff*[of $?X \text{ ' } S \{0, 1\} \lambda y. \text{prob } \{x \in S . ?X x = y\} * y$] *ss*

by *fastforce*

also have $\dots = \text{prob } \{x \in S . ?X x = 0\} * 0 + \text{prob } \{x \in S . ?X x = 1\} * 1$ **by**

auto

finally have *expectation* $?X = \text{prob } \{x \in S . ?X x = 1\} * 1$ **by** *auto*

thus *?thesis* **by** (*smt (verit) Collect-cong indicator-eq-1-iff*)

qed

end

end

4 A triangle inequality for sumsets

theory *Sumset-Triangle-Inequality*

imports

Pluenecke-Ruzsa-Inequality.Pluenecke-Ruzsa-Inequality

begin

context *additive-abelian-group*

begin

We show a useful triangle inequality for sumsets that does *not* follow from the Ruzsa triangle inequality. The proof follows the exposition in Zhao's book [4].

The following auxiliary lemma corresponds to Lemma 7.3.4 in Zhao's book [4].

lemma *triangle-ineq-sumsets-aux*:

fixes $X B Y :: 'a \text{ set}$

```

assumes  $hX$ : finite  $X$  and  $hB$ : finite  $B$  and  $hXG$ :  $X \subseteq G$  and  $hBG$ :  $B \subseteq G$ 
and
   $hXne$ :  $X \neq \{\}$  and  $hYX$ :  $\bigwedge Y. Y \subseteq X \implies Y \neq \{\} \implies \text{card}(\text{sumset } Y B) / \text{card } Y \geq$ 
   $\text{card}(\text{sumset } X B) / \text{card } X$  and  $hC$ : finite  $C$  and  $hCne$ :  $C \neq \{\}$  and  $hCG$ :
   $C \subseteq G$ 
shows  $\text{card}(\text{sumset } X (\text{sumset } C B)) / \text{card}(\text{sumset } X C) \leq \text{card}(\text{sumset } X B) / \text{card } X$ 
using  $hC$   $hCne$   $hCG$  proof (induct)
case empty
then show ?case by blast
next
case  $hcase$ : (insert  $c$   $C$ )
have  $hc$  :  $c \in G$  using  $hcase$  by auto
show  $\text{card}(\text{sumset } X B) / \text{card } X \geq$ 
   $\text{card}(\text{sumset } X (\text{sumset}(\text{insert } c C) B)) / \text{card}(\text{sumset } X (\text{insert } c C))$ 
proof(cases  $C = \{\}$ )
case True
then have  $\text{card}(\text{sumset } X (\text{insert } c C)) = \text{card } X$  using  $hc$   $hXG$   $hX$ 
by (simp add: card-sumset-singleton-eq le-iff-inf)
moreover have  $\text{card}(\text{sumset } X (\text{sumset}(\text{insert } c C) B)) \leq \text{card}(\text{sumset } X B)$ 
using  $hX$   $hB$   $hBG$   $hXG$ 
hc by (metis True card-sumset-le finite-sumset sumset-assoc sumset-commute)
ultimately show ?thesis by (simp add: divide-right-mono)
next
case  $hCne$ : False
have  $hCG$  :  $C \subseteq G$  using  $hcase$  by auto
have  $hstep$ :  $\text{card}(\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B)) \leq$ 
   $\text{card}(\text{sumset } X B) * \text{card}(\text{sumset } X \{c\} - \text{sumset } X C) / \text{card } X$ 
proof-
let ? $Y = \{x \in X. \text{sumset } \{x\} (\text{sumset } \{c\} B) \subseteq \text{sumset } X (\text{sumset } C B)\}$ 
have  $hYX$  : ? $Y \subseteq X$  and  $hY$ : finite ? $Y$  using finite-subset  $hX$  by auto
have  $hsub1$ :  $\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B) \subseteq$ 
   $\text{sumset}(\text{sumset } X \{c\} - \text{sumset } X C) B$ 
by (metis Diff-subset-conv Un-Diff-cancel sumset-assoc sumset-subset-Un1
sup.cobounded2)
have  $hcard1$  :  $\text{card}(\text{sumset } X B) = \text{card}(\text{sumset } X (\text{sumset } \{c\} B))$ 
by (metis card-sumset-singleton-eq finite-sumset  $hB$   $hX$   $hc$  sumset-Int-carrier
sumset-assoc
sumset-commute)
have  $hcard2$  :  $\text{card}(\text{sumset } ?Y B) = \text{card}(\text{sumset}(\text{sumset } ?Y \{c\}) B)$ 
using card-sumset-singleton-eq finite-sumset  $hB$   $hY$   $hc$  sumset-Int-carrier
sumset-assoc
sumset-commute by (smt (verit, ccfv-threshold))
have  $\text{sumset}(\text{sumset } ?Y \{c\}) B \subseteq \text{sumset } X (\text{sumset } C B)$ 
proof
fix  $d$  assume  $d \in \text{sumset}(\text{sumset } ?Y \{c\}) B$ 
then obtain  $a$   $b$  where  $ha$ :  $a \in ?Y$  and  $b \in B$  and  $hd$ :  $d = a \oplus c \oplus b$ 
by (smt (verit) empty-iff insert-iff sumset.cases)

```

then have $a \oplus c \oplus b \in \text{sumset } \{a\} (\text{sumset } \{c\} B)$
by (*smt (verit) associative composition-closed hBG hXG hc insertCI mem-Collect-eq subsetD sumset.sumsetI*)
then show $d \in \text{sumset } X (\text{sumset } C B)$ **using** *ha hd* **by** *blast*
qed
then have $\text{hdisj} : \text{disjnt } ((\text{sumset } X (\text{sumset } \{c\} B)) - (\text{sumset } X (\text{sumset } C B)))$
 $(\text{sumset } (\text{sumset } ?Y \{c\}) B)$
by (*auto simp add : disjnt-iff*)
have $\text{hsub2} : \text{sumset } (\text{sumset } X \{c\} - \text{sumset } X C) B \cup \text{sumset } (\text{sumset } ?Y \{c\}) B \subseteq$
 $\text{sumset } (\text{sumset } X \{c\}) B$ **by** (*simp add: sumset-mono*)
then have $\text{ineq1} : \text{card } (\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B))$
 $+ \text{card } (\text{sumset } ?Y B) \leq$
 $\text{card } (\text{sumset } X B)$
proof-
have $\text{card } (\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B)) + \text{card}$
 $(\text{sumset } ?Y B) =$
 $\text{card } ((\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B)) \cup \text{sumset}$
 $(\text{sumset } ?Y \{c\}) B)$
using *hdisj hcard2 card-Un-disjnt finite-sumset hX hYX finite-subset hB*
by (*metis (no-types, lifting) finite.emptyI finite.insertI finite-Diff*)
also have $\dots \leq \text{card } (\text{sumset } X (\text{sumset } \{c\} B))$ **using** *card-mono finite-sumset hX hB*
by (*metis (no-types, lifting) Diff-subset Un-subset-iff finite.emptyI finite.insertI*
hsub2 sumset-assoc)
finally show $\text{card } (\text{sumset } X (\text{sumset } \{c\} B) - \text{sumset } X (\text{sumset } C B)) +$
 $\text{card } (\text{sumset } ?Y B) \leq$
 $\text{card } (\text{sumset } X B)$ **using** *hcard1* **by** *auto*
qed
have $\text{ineq2} : \text{card } (\text{sumset } X \{c\} - \text{sumset } X C) \geq \text{card } X - \text{card } ?Y$
proof-
let $?Z = \{x \in X. \text{sumset } \{x\} \{c\} \subseteq \text{sumset } X C\}$
have $\text{hZY} : ?Z \subseteq ?Y$
by (*smt (verit, del-insts) Collect-mono-iff subset-refl sumset-assoc sumset-mono*)
have $\text{hinj} : \text{inj-on } (\lambda x. x \oplus c) (X - ?Z)$
proof (*intro inj-onI*)
fix $x y$ **assume** $x \in X - ?Z$ **and** $y \in X - ?Z$ **and** $h : x \oplus c = y \oplus c$
then have $x \in G$ **and** $y \in G$ **using** *hXG* **by** *auto*
then show $x = y$ **using** *h hc* **by** *simp*
qed
have $\text{himage} : (\lambda x. x \oplus c) ' (X - ?Z) = \text{sumset } X \{c\} - \text{sumset } X C$
proof
show $(\lambda x. x \oplus c) ' (X - ?Z) \subseteq \text{sumset } X \{c\} - \text{sumset } X C$
proof(*intro image-subsetI*)
fix x **assume** $\text{hx} : x \in X - ?Z$

```

    then have hxG : x ∈ G using hXG by auto
    then have hxc1: x ⊕ c ∈ sumset X {c} using hXG hc hx by auto
    have x ⊕ c ∈ sumset {x} {c} using hxG hc by auto
    then have hxc2: x ⊕ c ∉ sumset X C using hx hXG hc hCG
      using DiffD2 sumset.simps sumset.sumsetI by auto
    then show x ⊕ c ∈ sumset X {c} - sumset X C using hxc1 hxc2 by
simp
qed
show sumset X {c} - sumset X C ⊆ (λx. x ⊕ c) ‘ (X - ?Z)
proof
  fix d assume d ∈ sumset X {c} - sumset X C
  then obtain x where hd: d = x ⊕ c and hxc: x ⊕ c ∉ sumset X C and
hx: x ∈ X
    using sumset.cases by force
  then show d ∈ (λx. x ⊕ c) ‘ (X - ?Z) using hd hxc hx hXG hc by auto

  qed
  qed
  have hcard3: card (X - ?Z) = card (sumset X {c} - sumset X C)
    using card-image hinj himage by fastforce
  have card X = card ?Z + card (X - ?Z)
    by (simp add: card-Diff-subset card-mono hX)
  also have ... ≤ card ?Y + card (sumset X {c} - sumset X C)
    using hcard3 card-mono hZY hY by auto
  finally show ?thesis by simp
qed
have ineq3: card (sumset X B) - card (sumset ?Y B) ≤
  card (sumset X B) * (card X - card ?Y) / card X
proof (cases ?Y = {})
  case True
  then show ?thesis using card-eq-0-iff
    by (smt (verit) hX hXne minus-nat.diff-0 nonzero-mult-div-cancel-right
of-nat-eq-0-iff
  of-nat-mult sumset-empty(2))
next
  case hYne: False
  have card (sumset ?Y B) ≥ card (sumset X B) / card X * card ?Y using
assms(6)[OF hYX hYne]
  hX hYne hX hYX finite-subset divide-le-eq
  by (smt (z3) card-gt-0-iff hY mult-imp-div-pos-less of-nat-0-less-iff)
  moreover have card (sumset X B) / card X * card ?Y = (card (sumset X
B) * card ?Y) / card X
  by auto
  moreover have card (sumset X B) * card ?Y / card X * card X = card
(sumset X B) * card ?Y
  using hX by (simp add: field-simps)
  ultimately have card (sumset ?Y B) * card X ≥ card (sumset X B) * card
?Y
  using hX hXne of-nat-0-less-iff le-divide-eq

```

```

by (smt (z3) card-sumset-0-iff hBG hXG mult-cancel1 mult-cancel2 of-nat-le-0-iff
  of-nat-le-iff of-nat-mult)
then have real ((card (sumset X B) - card (sumset ?Y B)) * card X) ≤
  card (sumset X B) * (card X - card ?Y)
by (simp add: diff-mult-distrib diff-mult-distrib2)
thus ?thesis using le-divide-eq card-eq-0-iff hX hXne
by (smt (z3) of-nat-le-0-iff of-nat-mult)
qed
show ?thesis
proof-
  have real (card ((sumset X (sumset {c} B)) - (sumset X (sumset C B))))
≤
  card (sumset X B) - card (sumset ?Y B) using ineq1 by auto
  also have ... ≤ card (sumset X B) * (card X - card ?Y) / card X using
ineq3 by auto
  also have ... ≤ card (sumset X B) * card (sumset X {c} - sumset X C) /
card X using ineq2
  divide-le-cancel of-nat-less-0-iff of-nat-mono by (smt (verit, del-insts)
mult-le-mono2)
  finally show ?thesis by simp
qed
qed
have hinsert: real (card (sumset X (sumset (insert c C) B))) / real (card (sumset
X (insert c C))) =
  (card (sumset X (sumset {c} B) - sumset X (sumset C B)) + card (sumset X
(sumset C B))) /
  (card (sumset X {c} - sumset X C) + card (sumset X C))
  using sumset-insert2 card-Un-disjoint finite-sumset hX hB hC Diff-disjoint
Int-commute
  Un-commute finite.emptyI finite.insertI finite-Diff hcase.hyps(1)
by (smt (verit) Un-Diff-cancel2 insert-is-Un sumset-commute sumset-subset-Un2)
have hsplit: real (card (sumset X B)) * (card (sumset X {c} - sumset X C) +
card (sumset X C)) / card X =
  real (card (sumset X B)) * card (sumset X {c} - sumset X C) / card X +
  card (sumset X B) * card (sumset X C) / card X
  by (smt (verit, ccfv-threshold) add-divide-distrib add-mult-distrib2 of-nat-add
of-nat-mult)
have hind: card (sumset X B) * card (sumset X C) / card X ≥ card (sumset X
(sumset C B))
  using hcase(3)[OF hCne hCG] hXne hCne hcase(1) hX finite-sumset card-gt-0-iff
add-mult-distrib2 of-nat-mult card-eq-0-iff card-sumset-0-iff hCG
hXG of-nat-0-less-iff by (metis (no-types, opaque-lifting) divide-le-eq times-divide-eq-left)
have real (card (sumset X B)) * (card (sumset X {c} - sumset X C) + card
(sumset X C)) / card X ≥
  (card (sumset X (sumset {c} B) - sumset X (sumset C B)) + card (sumset X
(sumset C B)))
  using hsplit hind hstep by simp
then have card (sumset X B) / card X ≥

```

```

    (card (sumset X (sumset {c} B) - sumset X (sumset C B)) + card (sumset X
    (sumset C B))) /
    (card (sumset X {c} - sumset X C) + card (sumset X C)) using card-sumset-0-iff
    hCG hXG
    card-eq-0-iff hC hX hXne hCne divide-self le-divide-eq
    by (smt (z3) add-is-0 hcase.hyps(1) of-nat-le-0-iff times-divide-eq-left)
    thus real (card (sumset X (sumset (insert c C) B))) / real (card (sumset X
    (insert c C))) ≤
    real (card (sumset X B)) / real (card X) using hinsert by auto
qed
qed

```

The following inequality is the result corresponding to Corollary 7.3.6 in Zhao's book [4].

lemma *triangle-ineq-sumsets*:

assumes hA : finite A **and** hB : finite B **and** hC : finite C **and**

hAG : $A \subseteq G$ **and** hBG : $B \subseteq G$ **and** hCG : $C \subseteq G$

shows $\text{card } A * \text{card } (\text{sumset } B C) \leq \text{card } (\text{sumset } A B) * \text{card } (\text{sumset } A C)$

proof(cases $A = \{\}$)

case *True*

then show *?thesis* **by** *simp*

next

case $hAne$: *False*

show $\text{card } A * \text{card } (\text{sumset } B C) \leq \text{card } (\text{sumset } A B) * \text{card } (\text{sumset } A C)$

proof(cases $B = \{\}$)

case *True*

then show *?thesis* **by** *simp*

next

case $hBne$: *False*

define KS **where** $KS \equiv (\lambda X. \text{card } (\text{sumset } X C) / \text{real } (\text{card } X)) \text{ ' } (Pow A - \{\{\}\})$

define K **where** $K \equiv Min KS$

define X **where** $X \equiv @X. X \in Pow A - \{\{\}\} \wedge K = \text{card } (\text{sumset } X C) / \text{real } (\text{card } X)$

obtain KS : finite KS $KS \neq \{\}$

using KS -def hA $hAne$ **by** *blast*

then have $K \in KS$

using K -def Min -in **by** *blast*

then have $\exists X. X \in Pow A - \{\{\}\} \wedge K = \text{card } (\text{sumset } X C) / \text{real } (\text{card } X)$

using KS -def **by** *blast*

then obtain $X \in Pow A - \{\{\}\}$ **and** Keq : $K = \text{card } (\text{sumset } X C) / \text{real } (\text{card } X)$

by (*metis* (*mono-tags*, *lifting*) X -def *someI-ex*)

then have hX : $X \subseteq A$ $X \neq \{\}$

by *auto*

have $hXmin$: $\bigwedge Y. Y \subseteq A \implies Y \neq \{\} \implies$

$\text{card } (\text{sumset } X C) / \text{card } X \leq \text{card } (\text{sumset } Y C) / \text{card } Y$

using K -def KS -def Keq Min -le $KS(1)$ **by** *auto*

```

then have hXAineq: card (sumset X C) / card X ≤ card (sumset A C) / card
A
  by (metis hAne subset-refl)
have haux: real (card (sumset X (sumset B C))) / real (card (sumset X B))
  ≤ real (card (sumset X C)) / real (card X) using triangle-ineq-sumsets-aux[of
X C B]
  hXmin hX hA hAG finite-subset hB hC hBne hBG hC hCG subset-trans by
metis
have hXAsumset : real (card (sumset X B)) ≤ card (sumset A B)
  using hX(1) card-mono hA finite-sumset hB order-refl sumset-mono
  by (metis of-nat-le-iff)
have card (sumset B C) ≤ card (sumset X (sumset B C)) using assms hX
  finite-sumset hAG card-le-sumset
  by (metis bot.extremum-uniqueI dual-order.trans infinite-super subsetD subsetI

  sumset-subset-carrier)
also have ... ≤ (card (sumset X C) / card X) * card (sumset X B) using haux
divide-le-eq
  card-sumset-0-iff hBne hX hB hA finite-subset card-0-eq by (smt (verit) hCG
mult-eq-0-iff
  of-nat-0-eq-iff of-nat-0-le-iff sumset-assoc sumset-subset-carrier)
also have ... ≤ (card (sumset A C) / card A) * card (sumset A B)
  using hXAineq hXAsumset by (meson divide-nonneg-nonneg mult-mono'
of-nat-0-le-iff)
finally have card (sumset B C) ≤ (card (sumset A C) * card (sumset A B))
/ card A by simp
then have card (sumset B C) * card A ≤ card (sumset A C) * card (sumset
A B)
  using le-divide-eq hAne hA card-gt-0-iff by (smt (verit, ccfv-threshold)
card-0-eq of-nat-le-0-iff of-nat-le-iff of-nat-mult)
thus card A * card (sumset B C) ≤ card (sumset A B) * card (sumset A C)
  by (simp add: mult.commute)
qed
qed

end
end

```

5 Background material in additive combinatorics

This section outlines some background definitions and basic lemmas in additive combinatorics based on the notes by Gowers [3].

theory Additive-Combinatorics-Preliminaries

imports

Pluenecke-Ruzsa-Inequality.Pluenecke-Ruzsa-Inequality

begin

5.1 Additive quadruples and additive energy

context *additive-abelian-group*

begin

definition *additive-quadruple*:: 'a ⇒ 'a ⇒ 'a ⇒ 'a ⇒ bool **where**
additive-quadruple a b c d ≡ a ∈ G ∧ b ∈ G ∧ c ∈ G ∧ d ∈ G ∧ a ⊕ b = c ⊕ d

lemma *additive-quadruple-aux*:
assumes *additive-quadruple a b c d*
shows d = a ⊕ b ⊖ c
by (*metis additive-quadruple-def assms associative commutative inverse-closed invertible invertible-right-inverse2*)

lemma *additive-quadruple-diff*:
assumes *additive-quadruple a b c d*
shows a ⊖ c = d ⊖ b
by (*smt (verit, del-Insts) additive-quadruple-def assms associative commutative composition-closed inverse-closed invertible invertible-inverse-inverse invertible-right-inverse2*)

definition *additive-quadruple-set*:: 'a set ⇒ ('a × 'a × 'a × 'a) set **where**
additive-quadruple-set A ≡ {(a, b, c, d) | a b c d. a ∈ A ∧ b ∈ A ∧ c ∈ A ∧ d ∈ A ∧
additive-quadruple a b c d}

lemma *additive-quadruple-set-sub*:
additive-quadruple-set A ⊆ {(a, b, c, d) | a b c d. d = a ⊕ b ⊖ c ∧ a ∈ A ∧ b ∈ A ∧
c ∈ A ∧ d ∈ A} **using** *additive-quadruple-set-def additive-quadruple-def additive-quadruple-aux*
by *auto*

definition *additive-energy*:: 'a set ⇒ real **where**
additive-energy A ≡ card (*additive-quadruple-set A*) / (card A)³

lemma *card-ineq-aux-quadruples*:
assumes *finite A*
shows card (*additive-quadruple-set A*) ≤ (card A)³

proof –

define *f*:: 'a × 'a × 'a × 'a ⇒ 'a × 'a × 'a **where** *f* = (λ (a, b, c, d) . (a, b, c))
have *hinj*: *inj-on f* {(a, b, c, d) | a b c d. d = a ⊕ b ⊖ c ∧ a ∈ A ∧ b ∈ A ∧ c ∈ A ∧ d ∈ A}
unfolding *inj-on-def f-def* **by** *auto*
moreover **have** *himage*: *f* ‘ {(a, b, c, d) | a b c d. d = a ⊕ b ⊖ c ∧ a ∈ A ∧ b ∈ A ∧ c ∈ A ∧ d ∈ A} ⊆ A × A × A

unfolding *f-def* **by** *auto*
ultimately have $\text{card } (\text{additive-quadruple-set } A) \leq \text{card } (\{(a, b, c, d) \mid a \ b \ c \ d. \ d = a \oplus b \oplus c \wedge a \in A \wedge b \in A \wedge c \in A \wedge d \in A\})$
using *card-mono inj-on-finite[of f] assms additive-quadruple-set-sub finite-SigmaI*
by (*metis (no-types, lifting)*)
also have $\dots \leq \text{card } (A \times A \times A)$ **using** *himage hinj assms card-inj-on-le finite-SigmaI*
by (*metis (no-types, lifting)*)
finally show *?thesis* **by** (*simp add: card-cartesian-product power3-eq-cube*)
qed

lemma *additive-energy-upper-bound: additive-energy A ≤ 1*

proof (*cases finite A*)
assume *hA: finite A*
show *?thesis* **unfolding** *additive-energy-def* **using** *card-ineq-aux-quadruples hA*

card-cartesian-product power3-eq-cube **by** (*simp add: divide-le-eq*)
next
assume *infinite A*
thus *?thesis* **unfolding** *additive-energy-def* **by** *simp*
qed

5.2 On sums

definition *f-sum:: 'a ⇒ 'a set ⇒ nat* **where**
 $f\text{-sum } d \ A \equiv \text{card } \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$

lemma *pairwise-disjnt-sum-1:*
 $\text{pairwise } (\lambda s \ t. \ \text{disjnt } ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ s) \ ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ t)) \ (\text{sumset } A \ A)$
unfolding *disjnt-def* **by** (*intro pairwiseI*) (*auto*)

lemma *pairwise-disjnt-sum-2:*
 $\text{pairwise } \text{disjnt } ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}) \ '(\text{sumset } A \ A))$
unfolding *disjnt-def* **by** (*intro pairwiseI*) (*auto*)

lemma *sum-Union-span:*
assumes $A \subseteq G$
shows $\bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ '(\text{sumset } A \ A)) = A \times A$

proof
show $\bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ '(\text{sumset } A \ A)) \subseteq A \times A$ **by** *blast*
next
show $A \times A \subseteq \bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ '(\text{sumset } A \ A))$

A A)
proof (*intro subsetI*)
fix x **assume** $hxA: x \in A \times A$
then obtain $y z$ **where** $hxyz: x = (y, z)$ **and** $hy: y \in A$ **and** $hz: z \in A$ **by** *blast*
show $x \in (\bigcup d \in (\text{sumset } A \ A). \{(a, b) \mid a \in A \wedge b \in A \wedge a \oplus b = d\})$
using $hy \ hz \ \text{assms } hxA \ hxyz$ **by** *auto*
qed
qed

lemma *f-sum-le-card*:
assumes *finite* A **and** $A \subseteq G$
shows $f\text{-sum } d \ A \leq \text{card } A$

proof –
define $f:: ('a \times 'a) \Rightarrow 'a$ **where** $f \equiv (\lambda (a, b). a)$
have *inj-on* $f \ \{(a, b) \mid a \in A \wedge b \in A \wedge a \oplus b = d\}$
unfolding *f-def* **proof** (*intro inj-onI*)
fix $x \ y$ **assume** $x \in \{(a, b) \mid a \in A \wedge b \in A \wedge a \oplus b = d\}$ **and**
 $y \in \{(a, b) \mid a \in A \wedge b \in A \wedge a \oplus b = d\}$ **and**
 $hcase: (\text{case } x \ \text{of } (a, b) \Rightarrow a) = (\text{case } y \ \text{of } (a, b) \Rightarrow a)$
then obtain $x1 \ x2 \ y1 \ y2$ **where** $hx: x = (x1, x2)$ **and** $hy: y = (y1, y2)$ **and** $h1:$
 $x1 \oplus x2 = d$ **and**
 $h2: y1 \oplus y2 = d$ **and** $hx1: x1 \in A$ **and** $hx2: x2 \in A$ **and** $hy1: y1 \in A$ **and**
 $hy2: y2 \in A$ **by** *blast*
have $hxsub: x2 = d \ominus x1$
using $h1 \ hx1 \ hx2$ **assms** **by** (*metis additive-abelian-group.inverse-closed composition-closed*
additive-abelian-group-axioms commutative invertible invertible-left-inverse2
subsetD)
have $hysub: y2 = d \ominus y1$
using $h2 \ hy1 \ hy2$ **assms** **by** (*metis inverse-closed commutative composition-closed*
 $hy1 \ hy2$
invertible invertible-left-inverse2 subset-iff)
show $x = y$ **using** $hx \ hy \ hxsub \ hysub \ hcase$ **by** *auto*
qed
moreover **have** $f \ \{(a, b) \mid a \in A \wedge b \in A \wedge a \oplus b = d\} \subseteq A$ **using** *f-def*
by *auto*
ultimately show *?thesis* **using** *card-mono assms f-sum-def card-image[of f]*
by (*metis (mono-tags, lifting)*)
qed

lemma *f-sum-card*:
assumes $A \subseteq G$ **and** $hA: \text{finite } A$
shows $(\sum d \in (\text{sumset } A \ A). (f\text{-sum } d \ A)) = (\text{card } A)^{\wedge 2}$

proof –
have $fin: \forall X \in ((\lambda d. \{(a, b) \mid a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ \text{sumset } A \ A).$ *finite* X
proof

fix X **assume** $hX: X \in (\lambda d. \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\})$ ‘
 $(\text{sumset } A \ A)$
then obtain d **where** $hXd: X = \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$
by *blast*
show *finite* X **using** $hA \ hXd$ *finite-subset finite-cartesian-product*
by $(\text{smt } (\text{verit}, \text{best}) \text{ mem-Collect-eq mem-Sigma-iff rev-finite-subset subrelI})$
qed
have $(\sum d \in (\text{sumset } A \ A). \ f\text{-sum } d \ A) = \text{card } (A \times A)$
unfolding *f-sum-def*
using *sum-card-image[of sumset A A (λ d .{(a, b) | a b. a ∈ A ∧ b ∈ A ∧ (a ⊕ b = d)})]*
pairwise-disjnt-sum-1 hA finite-sumset card-Union-disjoint[of ((λd. {(a, b) | a b. a ∈ A ∧ b ∈ A ∧ a ⊕ b = d}) ‘ sumset A A)]
fin pairwise-disjnt-sum-2 hA finite-sumset sum-Union-span assms **by** *auto*
thus *?thesis* **using** *card-cartesian-product power2-eq-square* **by** *metis*
qed

lemma *f-sum-card-eq*:

assumes $A \subseteq G$

shows $\forall x \in \text{sumset } A \ A. \ (f\text{-sum } x \ A) \hat{=} =$

$\text{card } \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}$

proof

fix x **assume** $x \in \text{sumset } A \ A$

define C **where** $hC: C = \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$

$\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}$

define $f :: 'a \times 'a \times 'a \times 'a \Rightarrow ('a \times 'a) \times ('a \times 'a)$ **where** $f = (\lambda (a, b, c, d). ((a, b), (c, d)))$

have $h\text{finj}: \text{inj-on } f \ C$ **unfolding** *f-def* **by** (intro inj-onI) *(auto)*

have $f \text{ ‘ } C = \{((a, b), (c, d)) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus b = x \wedge c \oplus d = x\}$

proof

show $f \text{ ‘ } C \subseteq \{((a, b), (c, d)) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus b = x \wedge c \oplus d = x\}$

unfolding *f-def* hC **by** *auto*

next

show $\{((a, b), c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus b = x \wedge c \oplus d = x\} \subseteq f \text{ ‘ } C$

proof

fix z **assume** $z \in \{((a, b), (c, d)) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus b = x \wedge c \oplus d = x\}$

then obtain $a \ b \ c \ d$ **where** $hz: z = ((a, b), (c, d))$ **and** $ha: a \in A$ **and** $hb: b \in A$ **and** $hc: c \in A$ **and** $hd: d \in A$

and $hab: a \oplus b = x$ **and** $hcd: c \oplus d = x$ **by** *blast*

then have $habcd: (a, b, c, d) \in C$ **using** *additive-quadruple-def* $assms \ hC$ **by** *auto*

show $z \in f \text{ ‘ } C$ **using** $hz \ f\text{-def} \ habcd$ **by** *force*

qed
qed
moreover have $\{((a, b), c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus b = x \wedge c \oplus d = x\} =$
 $\{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = x\} \times \{(c, d) \mid c \ d. \ c \in A \wedge d \in A \wedge c \oplus d = x\}$ **by** *blast*
ultimately have $\text{card } C = \text{card } (\{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = x\})^{\wedge 2}$
using *hfinj card-image[of f] card-cartesian-product* **by** (*metis (no-types, lifting) Sigma-cong power2-eq-square*)
thus $(f\text{-sum } x \ A)^{\wedge 2} = \text{card } (\{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\})$ **using** *hC f-sum-def* **by** *auto*
qed

lemma pairwise-disjoint-sum:
pairwise $(\lambda s \ t. \ \text{disjnt } ((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \ s)$
 $((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \ t))$ (*sumset A A*)
unfolding *disjnt-def* **by** (*intro pairwiseI*) (*auto*)

lemma pairwise-disjnt-quadruple-sum:
pairwise $\text{disjnt } ((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \ '(\text{sumset } A \ A))$
unfolding *disjnt-def* **by** (*intro pairwiseI*) (*auto*)

lemma quadruple-sum-Union-eq:
 $\bigcup ((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \ '(\text{sumset } A \ A)) =$
 $\text{additive-quadruple-set } A$

proof
show $(\bigcup x \in \text{sumset } A \ A. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a$
 $b \ c \ d \wedge$
 $a \oplus b = x \wedge c \oplus d = x\}) \subseteq \text{additive-quadruple-set } A$
unfolding *additive-quadruple-set-def* **by** (*intro Union-least*) (*auto*)

next
show $\text{additive-quadruple-set } A \subseteq (\bigcup x \in \text{sumset } A \ A. \ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\})$
unfolding *additive-quadruple-set-def* *additive-quadruple-def* **by** (*intro subsetI*) (*auto*)
qed

lemma *f-sum-card-quadruple-set*:

assumes $hAG: A \subseteq G$ **and** $hA: \text{finite } A$

shows $(\sum d \in (\text{sumset } A \ A). (f\text{-sum } d \ A)^{\wedge 2}) = \text{card } (\text{additive-quadruple-set } A)$

proof –

have $fin: \forall X \in ((\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \text{ ' } (\text{sumset } A \ A)). \text{finite } X$

proof

fix X **assume** $X \in (\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$

$\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \text{ ' } \text{sumset } A \ A$

then obtain x **where** $hX: X = \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$

$\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}$

by *blast*

show *finite* X **using** hA hX *finite-subset* *finite-cartesian-product*

by (*smt* (*verit*, *best*) *mem-Collect-eq* *mem-Sigma-iff* *rev-finite-subset* *subrelI*)

qed

have $(\sum d \in \text{sumset } A \ A. (f\text{-sum } d \ A)^2) =$

$\text{card } (\bigcup ((\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \text{ ' } (\text{sumset } A \ A)))$

using *f-sum-card-eq* hAG *sum-card-image*[*of* *sumset* $A \ A$ $(\lambda x. \{(a, b, c, d) \mid a \ b$

$c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\})$]

pairwise-disjoint-sum *card-Union-disjoint*[*of* $(\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge$

$c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus b = x \wedge c \oplus d = x\}) \text{ ' } (\text{sumset } A \ A)$]

$(\text{sumset } A \ A)$]

fin *pairwise-disjnt-quadruple-sum* hA *finite-sumset* **by** *auto*

then show *?thesis* **using** *quadruple-sum-Union-eq* **by** *auto*

qed

lemma *f-sum-card-quadruple-set-additive-energy*: **assumes** $A \subseteq G$ **and** *finite* A

shows $(\sum d \in \text{sumset } A \ A. (f\text{-sum } d \ A)^{\wedge 2}) = \text{additive-energy } A * (\text{card } A)^{\wedge 2}$

using *assms* *f-sum-card-quadruple-set* *additive-energy-def* **by** *force*

definition *popular-sum*:: $'a \Rightarrow \text{real} \Rightarrow 'a \ \text{set} \Rightarrow \text{bool}$ **where**

$\text{popular-sum } d \ \vartheta \ A \equiv f\text{-sum } d \ A \geq \vartheta * \text{of-real } (\text{card } A)$

definition *popular-sum-set*:: $\text{real} \Rightarrow 'a \ \text{set} \Rightarrow 'a \ \text{set}$ **where**

$\text{popular-sum-set } \vartheta \ A \equiv \{d \in \text{sumset } A \ A. \text{popular-sum } d \ \vartheta \ A\}$

5.3 On differences

The following material is directly analogous to the material given previously on sums. All definitions and lemmas are the corresponding ones for

differences. E.g. $f\text{-diff}$ corresponds to $f\text{-sum}$.

definition $f\text{-diff}:: 'a \Rightarrow 'a \text{ set} \Rightarrow \text{nat}$ **where**

$$f\text{-diff } d \ A \equiv \text{card } \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$$

lemma $\text{pairwise-disjnt-diff-1}$:

$\text{pairwise } (\lambda s \ t. \ \text{disjnt } ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ s) \ ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ t)) \ (\text{differenceset } A \ A)$
using disjnt-def **by** (intro pairwiseI) (auto)

lemma $\text{pairwise-disjnt-diff-2}$:

$\text{pairwise disjnt } ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}) \ ' (\text{differenceset } A \ A))$
unfolding disjnt-def **by** (intro pairwiseI) (auto)

lemma diff-Union-span :

assumes $A \subseteq G$

shows $\bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ ' (\text{differenceset } A \ A)) = A \times A$

proof

show $\bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ ' (\text{differenceset } A \ A)) \subseteq A \times A$

by blast

next

show $A \times A \subseteq \bigcup ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \oplus b = d)\}) \ ' (\text{differenceset } A \ A))$

proof (intro subsetI)

fix $x \ y$ **assume** $hxA: x \in A \times A$

then obtain $y \ z$ **where** $hxyz: x = (y, z)$ **and** $hy: y \in A$ **and** $hz: z \in A$ **by** blast

show $x \in (\bigcup d \in (\text{differenceset } A \ A). \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\})$

using $hy \ hz \ \text{assms } hxA \ hxyz$ **by** auto

qed

qed

lemma $f\text{-diff-le-card}$:

assumes $\text{finite } A$ **and** $A \subseteq G$

shows $f\text{-diff } d \ A \leq \text{card } A$

proof –

define $f:: ('a \times 'a) \Rightarrow 'a$ **where** $f \equiv (\lambda (a, b). \ a)$

have $\text{inj-on } f \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$

unfolding $f\text{-def}$ **proof** (intro inj-onI)

fix $x \ y$ **assume** $x \in \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$ **and**

$y \in \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \oplus b = d\}$ **and**

$hcase: (\text{case } x \ \text{of } (a, b) \Rightarrow a) = (\text{case } y \ \text{of } (a, b) \Rightarrow a)$

then obtain $x1 \ x2 \ y1 \ y2$ **where** $hx: x = (x1, x2)$ **and** $hy: y = (y1, y2)$ **and** $h1: x1 \oplus x2 = d$ **and**

$h2: y1 \oplus y2 = d$ **and** $hx1: x1 \in A$ **and** $hx2: x2 \in A$ **and** $hy1: y1 \in A$ **and** $hy2: y2 \in A$ **by** blast

have h_{xsub} : $x2 = x1 \ominus d$
using $h1$ *assms associative commutative composition-closed $hx1$ $hx2$*
by (*smt (verit, best) inverse-closed invertible invertible-left-inverse2 subset-iff*)
have h_{ysub} : $y2 = y1 \ominus d$
using $h2$ *assms associative commutative composition-closed $hy1$ $hy2$*
by (*smt (verit, best) inverse-closed invertible invertible-left-inverse2 subset-iff*)
show $x = y$ **using** hx hy h_{xsub} h_{ysub} $hcase$ **by** *auto*
qed
moreover have $f \cdot \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \ominus b = d\} \subseteq A$ **using** $f-def$
by *auto*
ultimately show *?thesis* **using** $card-mono$ *assms $f-diff-def$ $card-image$ [of f]*
by (*metis (mono-tags, lifting)*)
qed

lemma $f-diff-card$:

assumes $A \subseteq G$ **and** hA : *finite* A

shows $(\sum d \in (differenceset\ A\ A). \ f-diff\ d\ A) = (card\ A) \wedge 2$

proof –

have fin : $\forall X \in ((\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \ominus b = d)\}) \cdot (differenceset\ A\ A)).$

finite X

proof

fix X **assume** hX : $X \in (\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \ominus b = d\}) \cdot (differenceset\ A\ A)$

then obtain d **where** hXd : $X = \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \ominus b = d\}$

and

$d \in (differenceset\ A\ A)$ **by** *blast*

have hXA : $X \subseteq A \times A$ **using** hXd **by** *blast*

show *finite* X **using** hXA hA *finite-subset* **by** *blast*

qed

have $(\sum d \in (differenceset\ A\ A). \ f-diff\ d\ A) = card\ (A \times A)$

unfolding $f-diff-def$ **using** *sum-card-image*[of $differenceset\ A\ A$

$(\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge (a \ominus b = d)\})$] *pairwise-disjnt-diff-1*

card-Union-disjoint[of $(\lambda d. \ \{(a, b) \mid a \ b. \ a \in A \wedge b \in A \wedge a \ominus b = d\}) \cdot (differenceset\ A\ A)$]

fin *pairwise-disjnt-diff-2* *diff-Union-span* *assms* hA *finite-minusset* *finite-sumset*

by *auto*

thus *?thesis* **using** *card-cartesian-product* *power2-eq-square* **by** *metis*

qed

lemma $f-diff-card-eq$:

assumes $A \subseteq G$

shows $\forall x \in differenceset\ A\ A. \ (f-diff\ x\ A) \wedge 2 =$

$card\ \{(a, b, c, d) \mid a \ b \ c \ d. \ a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$

$additive-quadruple\ a \ b \ c \ d \wedge a \ominus c = x \wedge d \ominus b = x\}$

proof

fix x **assume** $x \in differenceset\ A\ A$

define C **where** $hC: C = \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}$
define $f: 'a \times 'a \times 'a \times 'a \Rightarrow ('a \times 'a) \times ('a \times 'a)$ **where** $f = (\lambda (a, b, c, d). ((a, c), (d, b)))$
have $hfinj: inj\text{-on } f \ C$ **using** $f\text{-def}$ **by** $(intro \ inj\text{-onI})$ $(auto)$
have $f' \ C = \{((a, c), (d, b)) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus c = x \wedge d \oplus b = x\}$
proof
show $f' \ C \subseteq \{((a, c), (d, b)) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus c = x \wedge d \oplus b = x\}$
unfolding $f\text{-def } hC$ **by** $auto$
next
show $\{((a, c), d, b) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus c = x \wedge d \oplus b = x\} \subseteq f' \ C$
proof
fix z **assume** $z \in \{((a, c), (d, b)) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus c = x \wedge d \oplus b = x\}$
then obtain $a \ b \ c \ d$ **where** $hz: z = ((a, c), (d, b))$ **and** $ha: a \in A$ **and** $hb: b \in A$ **and** $hc: c \in A$ **and** $hd: d \in A$
and $hab: a \oplus c = x$ **and** $hcd: d \oplus b = x$ **by** $blast$
have $\text{additive-quadruple } a \ b \ c \ d$
using $assms$ **by** $(metis \ (no\text{-types}, \ lifting) \ ha \ hb \ hc \ hd \ \text{additive-quadruple-def} \ \text{associative} \ \text{commutative composition-closed} \ hab \ hcd \ \text{inverse-closed} \ \text{invertible} \ \text{invertible-right-inverse2} \ \text{subset-eq})$
then have $habcd: (a, b, c, d) \in C$ **using** $hab \ hcd \ hC \ ha \ hb \ hc \ hd$ **by** $blast$
show $z \in f' \ C$ **using** $hz \ f\text{-def} \ habcd \ \text{image-iff}$ **by** $fastforce$
qed
moreover have $\{((a, c), (d, b)) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge a \oplus c = x \wedge d \oplus b = x\} = \{(a, c) \mid a \ c. a \in A \wedge c \in A \wedge a \oplus c = x\} \times \{(d, b) \mid d \ b. d \in A \wedge b \in A \wedge d \oplus b = x\}$ **by** $blast$
moreover have $\text{card } C = \text{card } (f' \ C)$ **using** $hfinj \ \text{card-image}[of \ f]$ **by** $auto$
ultimately have $\text{card } C = \text{card } (\{(a, c) \mid a \ c. a \in A \wedge c \in A \wedge a \oplus c = x\})^2$
using $hfinj \ \text{card-image}[of \ f] \ \text{card-cartesian-product} \ \text{Sigma-cong} \ \text{power2-eq-square}$ **by** $(smt \ (verit, \ best))$
thus $(f\text{-diff } x \ A)^2 = \text{card } \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}$
using $f\text{-diff-def } hC$ **by** $simp$
qed

lemma $\text{pairwise-disjoint-diff}$:

$\text{pairwise } (\lambda s \ t. \ \text{disjnt } ((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \ s) \ ((\lambda x. \ \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \ t)) \ (\text{differenceset } A \ A)$
unfolding disjnt-def **by** $(intro \ \text{pairwiseI})$ $(auto)$

lemma *pairwise-disjnt-quadruple-diff*:

pairwise disjnt $((\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \text{ ' (differenceset } A \ A))$

unfolding *disjnt-def* **by** *(intro pairwiseI)* *(auto)*

lemma *quadruple-diff-Union-eq*:

$\bigcup ((\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \text{ ' (differenceset } A \ A)) =$

additive-quadruple-set A

proof

show $(\bigcup x \in \text{differenceset } A \ A. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge$
 $d \in A \wedge$

additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \subseteq \text{additive-quadruple-set}$
 A

unfolding *additive-quadruple-set-def* **by** *(intro Union-least)* *(auto)*

next

show *additive-quadruple-set* $A \subseteq (\bigcup x \in \text{differenceset } A \ A.$

$\{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\})$

proof *(intro subsetI)*

fix x **assume** $x \in \text{additive-quadruple-set } A$

then obtain $x_1 \ x_2 \ x_3 \ x_4$ **where** $hx: x = (x_1, x_2, x_3, x_4)$ **and** $hx1: x_1 \in A$
and $hx2: x_2 \in A$ **and** $hx3: x_3 \in A$

and $hx4: x_4 \in A$ **and** $hxadd: \text{additive-quadruple } x_1 \ x_2 \ x_3 \ x_4$

using *additive-quadruple-set-def* **by** *auto*

have $hxmem: (x_1, x_2, x_3, x_4) \in \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in$
 $A \wedge d \in A \wedge$

additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x_1 \oplus x_3 \wedge d \oplus b = x_1 \oplus x_3\}$

using *additive-quadruple-diff* $hx1 \ hx2 \ hx3 \ hx4 \ hxadd$ **by** *auto*

show $x \in (\bigcup x \in \text{differenceset } A \ A. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in$
 $A \wedge d \in A \wedge$

additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\})$

using $hx \ hxmem \ hx1 \ hx3$ *additive-quadruple-def* $hxadd$ **by** *auto*

qed

qed

lemma *f-diff-card-quadruple-set*:

assumes $hAG: A \subseteq G$ **and** $hA: \text{finite } A$

shows $(\sum d \in (\text{differenceset } A \ A). (f\text{-diff } d \ A)^{\wedge 2}) = \text{card } (\text{additive-quadruple-set}$
 $A)$

proof –

have $fin: \forall X \in ((\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in A \wedge b \in A \wedge c \in A \wedge d \in A$
 \wedge

additive-quadruple $a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}) \text{ ' (differenceset } A \ A)).$
finite X

proof
fix X **assume** $X \in (\lambda x. \{(a, b, c, d) \mid a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\})$ ‘*differenceset* $A \ A$
then obtain x **where** $hX: X = \{(a, b, c, d) \mid a \in A \wedge b \in A \wedge c \in$
 $A \wedge d \in A \wedge$
 $\text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b = x\}$ **and** $x \in \text{differenceset}$
 $A \ A$ **by blast**
show *finite* X **using** $hX \ hA$ *finite-subset finite-cartesian-product*
by (*smt (verit, best) mem-Collect-eq mem-Sigma-iff rev-finite-subset subrelI*)
qed
have $(\sum_{d \in \text{differenceset } A \ A. (f\text{-diff } d \ A)^2}) = \text{card } (\bigcup ((\lambda x. \{(a, b, c, d) \mid a \ b$
 $c \ d. a \in A \wedge$
 $b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b =$
 $x\})$ ‘(*differenceset* $A \ A$))
using *f-diff-card-eq hAG sum-card-image[of differenceset* $A \ A (\lambda x. \{(a, b, c, d)$
 $\mid a \ b \ c \ d.$
 $a \in A \wedge b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x$
 $\wedge d \oplus b = x\})]$
pairwise-disjoint-diff card-Union-disjoint[of $(\lambda x. \{(a, b, c, d) \mid a \ b \ c \ d. a \in$
 $A \wedge$
 $b \in A \wedge c \in A \wedge d \in A \wedge \text{additive-quadruple } a \ b \ c \ d \wedge a \oplus c = x \wedge d \oplus b$
 $= x\})$ ‘
 $(\text{differenceset } A \ A)]$ *fin pairwise-disjnt-quadruple-diff hA finite-minusset fi-*
nite-sumset **by auto**
thus *?thesis using quadruple-diff-Union-eq* **by auto**
qed

lemma *f-diff-card-quadruple-set-additive-energy*: **assumes** $A \subseteq G$ **and** *finite* A
shows $(\sum_{d \in \text{differenceset } A \ A. (f\text{-diff } d \ A)^2}) = \text{additive-energy } A * (\text{card}$
 $A)^3$
using *assms f-diff-card-quadruple-set additive-energy-def* **by force**

definition *popular-diff*:: $'a \Rightarrow \text{real} \Rightarrow 'a \text{ set} \Rightarrow \text{bool}$ **where**
 $\text{popular-diff } d \ \vartheta \ A \equiv f\text{-diff } d \ A \geq \vartheta * \text{of-real } (\text{card } A)$

definition *popular-diff-set*:: $\text{real} \Rightarrow 'a \text{ set} \Rightarrow 'a \text{ set}$ **where**
 $\text{popular-diff-set } \vartheta \ A \equiv \{d \in \text{differenceset } A \ A. \text{popular-diff } d \ \vartheta \ A\}$

end
end

6 Results on lower bounds on additive energy

theory *Additive-Energy-Lower-Bounds*

imports

Additive-Combinatorics-Preliminaries

Miscellaneous-Lemmas

begin

context *additive-abelian-group*

begin

The following corresponds to Proposition 2.11 in Gowers's notes [3].

proposition *additive-energy-lower-bound-sumset*: **fixes** $C :: \text{real}$

assumes *finite A and $A \subseteq G$ and $(\text{card } (\text{sumset } A A)) \leq C * \text{card } A$ and $\text{card } A \neq 0$*

shows *additive-energy $A \geq 1/C$*

proof –

have $(\text{card } A)^{\wedge 2} = (\sum x \in \text{sumset } A A. \text{real } (f\text{-sum } x A))$

using *assms f-sum-card by (metis of-nat-sum)*

also have $\dots \leq (\text{card}(\text{sumset } A A))^{\text{powr}(1/2)} * (\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2})^{\text{powr}(1/2)}$

using *Cauchy-Schwarz-ineq-sum2[$\text{of } \lambda d. 1 \lambda d. f\text{-sum } d A$] by auto*

also have $\dots \leq ((C * (\text{card } A))^{\text{powr}(1/2)}) * ((\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2}))^{\text{powr}(1/2)}$

by *(metis mult.commute mult-left-mono assms(3) of-nat-0-le-iff powr-ge-zero powr-mono2*

zero-le-divide-1-iff zero-le-numeral)

finally have $((\text{card } A)^{\wedge 2})^{\wedge 2} \leq (((C * (\text{card } A))^{\text{powr}(1/2)}) * ((\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2}))^{\text{powr}(1/2)})^{\wedge 2}$

by *(metis of-nat-0-le-iff of-nat-power-eq-of-nat-cancel-iff power-mono)*

then have $(\text{card } A)^{\wedge 4} \leq (((C * (\text{card } A)) * ((\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2})))^{\text{powr}(1/2)})^{\wedge 2}$

by *(smt (verit) assms of-nat-0-le-iff powr-mult*

mult.left-commute power2-eq-square power3-eq-cube power4-eq-xxxx power-commutes)

then have $(\text{card } A)^{\wedge 4} \leq ((C * (\text{card } A)) * ((\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2})))^{\wedge 2}$

using *assms powr-half-sqrt of-nat-0 of-nat-le-0-iff power-mult-distrib*

real-sqrt-pow2 by (smt (verit, best) powr-mult)

moreover have *additive-energy $A = (\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2}) / (\text{card } A)^{\wedge 3}$*

using *additive-energy-def f-sum-card-quadruple-set assms by simp*

moreover then have *additive-energy $A * (\text{card } A)^{\wedge 3} = (\sum x \in \text{sumset } A A. (f\text{-sum } x A)^{\wedge 2})$*

using *assms by simp*

ultimately have *(additive-energy $A) \geq ((\text{card } A)^{\wedge 4}) / (C * (\text{card } A)^{\wedge 4})$*

using *additive-energy-upper-bound*

additive-abelian-group-axioms assms divide-le-eq divide-le-eq-1-pos mult.left-commute

mult-left-mono of-nat-0-eq-iff of-nat-0-le-iff power-eq-0-iff power3-eq-cube power4-eq-xxxx

linorder-not-less mult.assoc mult-zero-left of-nat-0-less-iff of-nat-mult

order-trans-rules(23) times-divide-eq-right by (smt (verit) card-sumset-0-iff

div-by-1 mult-cancel-left1 nonzero-mult-div-cancel-left nonzero-mult-divide-mult-cancel-right

$\text{nonzero-mult-divide-mult-cancel-right2}$ of-nat-1 of-nat-le-0-iff times-divide-eq-left)
then show ?thesis **by** (simp add: assms)
qed

An analogous version of Proposition 2.11 where the assumption is on a difference set is given below. The proof is identical to the proof of *additive-energy-lower-bound-sumset* above (with the obvious modifications).

proposition *additive-energy-lower-bound-differenceset*: **fixes** $C::\text{real}$
assumes *finite A and* $A \subseteq G$ **and** $(\text{card}(\text{differenceset } A \ A)) \leq C * \text{card } A$ **and**
 $\text{card } A \neq 0$
shows *additive-energy A* $\geq 1/C$

proof –

have $(\text{card } A)^2 = (\sum x \in \text{differenceset } A \ A. \text{real } (f\text{-diff } x \ A))$
using *assms f-diff-card by* (metis of-nat-sum)
also have $\dots \leq (\text{card}(\text{differenceset } A \ A))^{\text{powr } (1/2)} * (\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2)^{\text{powr } (1/2)}$
using *Cauchy-Schwarz-ineq-sum2*[of $\lambda d. 1 \ \lambda d. f\text{-diff } d \ A$] **by** *auto*
also have $\dots \leq ((C * (\text{card } A))^{\text{powr } (1/2)}) * ((\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2))^{\text{powr } (1/2)}$
by (metis *mult.commute mult-left-mono assms(3) of-nat-0-le-iff powr-ge-zero powr-mono2 zero-le-divide-1-iff zero-le-numeral*)
finally have $((\text{card } A)^2)^2 \leq (((C * (\text{card } A))^{\text{powr } (1/2)}) * ((\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2))^{\text{powr } (1/2)})^2$
by (metis *of-nat-0-le-iff of-nat-power-eq-of-nat-cancel-iff power-mono*)
then have $(\text{card } A)^4 \leq (((C * (\text{card } A)) * ((\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2)))^{\text{powr } (1/2)})^2$
by (smt (verit) *assms of-nat-0-le-iff powr-mult mult.left-commute power2-eq-square power3-eq-cube power4-eq-xxxx power-commutes*)
then have $(\text{card } A)^4 \leq ((C * (\text{card } A)) * ((\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2)))^2$
using *assms powr-half-sqrt of-nat-0 of-nat-le-0-iff power-mult-distrib real-sqrt-pow2* **by** (smt (verit, best) *powr-mult*)
moreover have *additive-energy A* $= (\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2) / (\text{card } A)^3$
using *additive-energy-def f-diff-card-quadruple-set assms* **by** *simp*
moreover then have *additive-energy A* $* (\text{card } A)^3 = (\sum x \in \text{differenceset } A \ A. (f\text{-diff } x \ A)^2)$
using *assms* **by** *simp*
ultimately have $(\text{additive-energy } A) \geq ((\text{card } A)^4) / (C * (\text{card } A)^4)$
using *additive-energy-upper-bound additive-abelian-group-axioms assms divide-le-eq divide-le-eq-1-pos mult.left-commute*

mult-left-mono of-nat-0-eq-iff of-nat-0-le-iff power-eq-0-iff power3-eq-cube
 power4-eq-xxxx
 linorder-not-less *mult.assoc mult-zero-left of-nat-0-less-iff of-nat-mult*
 $\text{order-trans-rules(23)}$ *times-divide-eq-right* **by** (smt (verit) *card-sumset-0-iff*)

```

    div-by-1 mult-cancel-left1 nonzero-mult-div-cancel-left nonzero-mult-divide-mult-cancel-right
    nonzero-mult-divide-mult-cancel-right2 of-nat-1 of-nat-le-0-iff times-divide-eq-left)
  then show ?thesis by (simp add: assms)
qed

end
end

```

7 Towards the proof of the Balog–Szemerédi–Gowers Theorem

```

theory Balog-Szemeredi-Gowers-Main-Proof

```

```

  imports

```

```

    Prob-Space-Lemmas

```

```

    Graph-Theory-Preliminaries

```

```

    Sumset-Triangle-Inequality

```

```

    Additive-Combinatorics-Preliminaries

```

```

begin

```

```

context additive-abelian-group

```

```

begin

```

After having introduced all the necessary preliminaries in the imported files, we are now ready to follow the chain of the arguments for the main proof as in Gowers’s notes [3].

The following lemma corresponds to Lemma 2.13 in Gowers’s notes [3].

```

lemma (in fin-bipartite-graph) proportion-bad-pairs-subset-bipartite:

```

```

  fixes c::real

```

```

  assumes c > 0

```

```

  obtains X' where X' ⊆ X and card X' ≥ density * card X / sqrt 2 and
  card (bad-pair-set X' Y c) / (card X')^2 ≤ 2 * c / density^2

```

```

proof (cases density = 0)

```

```

  case True

```

```

  then show ?thesis using that[of {}] bad-pair-set-def by auto

```

```

next

```

```

  case False

```

```

  then have dgt0: density > 0 using density-simp by auto

```

```

  let ?M = uniform-count-measure Y

```

```

  interpret P: prob-space ?M

```

```

  by (simp add: Y-not-empty partitions-finite prob-space-uniform-count-measure)

```

```

  have sp: space ?M = Y

```

```

  by (simp add: space-uniform-count-measure)

```

```

  have avg-degree: P.expectation (λ y . card (neighborhood y)) = density * (card X)

```

```

proof –

```

```

have density = (∑ y ∈ Y . degree y)/(card X * card Y)
using edge-size-degree-sumY density-simp by simp
then have d: density * (card X) = (∑ y ∈ Y . degree y)/(card Y)
using card-edges-between-set edge-size-degree-sumY partitions-finite(1) parti-
tions-finite(2) by auto
have P.expectation (λ y . card (neighborhood y)) = P.expectation (λ y . degree
y)
using alt-deg-neighborhood by simp
also have ... = (∑ y ∈ Y . degree y)/(card Y) using P.expectation-uniform-count
by (simp add: partitions-finite(2))
finally show ?thesis using d by simp
qed

then have card-exp-gt: P.expectation (λ y . (card (neighborhood y))^2) ≥ den-
sity^2 * (card X)^2
proof -
have P.expectation (λ y . (card (neighborhood y))^2) ≥ (P.expectation (λ y .
card (neighborhood y)))^2
using P.cauchy-schwarz-ineq-var-uniform partitions-finite(2) by auto
thus ?thesis using avg-degree
by (metis of-nat-power power-mult-distrib)
qed

define B where B ≡ bad-pair-set X Y c
define B' where B' ≡ λ y . bad-pair-set (neighborhood y) Y c
have finB: finite B using bad-pair-set-finite partitions-finite B-def by auto
have ∧ x . x ∈ X ⇒ x ∈ V using partitions-ss(1) by auto
have card B ≤ (card (X × X)) using B-def bad-pair-set-ss partitions-finite
card-mono finite-cartesian-product-iff by metis
then have card-B: card B ≤ (card X)^2
by (metis card-cartesian-prod-square partitions-finite(1))

have ∧ x x' . (x, x') ∈ B ⇒ P.prob {y ∈ Y . {x, x'} ⊆ neighborhood y} < c
proof -
fix x x' assume assm: (x, x') ∈ B
then have x ∈ X x' ∈ X unfolding B-def bad-pair-set-def bad-pair-def by
auto
then have card-eq: card {v ∈ V . vert-adj v x ∧ vert-adj v x'} = card {y ∈ Y .
vert-adj y x ∧ vert-adj y x'}
by (metis (no-types, lifting) X-vert-adj-Y vert-adj-edge-iff2 vert-adj-imp-in V)

have ltc: card {v ∈ V . vert-adj v x ∧ vert-adj v x'}/(card Y) < c
using assm by (auto simp add: B-def bad-pair-set-def bad-pair-def code-
gree-normalized-def codegree-def vert-adj-sym)
have {y ∈ Y . {x, x'} ⊆ neighborhood y} = {y ∈ Y . vert-adj y x ∧ vert-adj y
x'}
using bad-pair-set-def bad-pair-def neighborhood-def vert-adj-imp-in V vert-adj-imp-in V
by auto
then have P.prob {y ∈ Y . {x, x'} ⊆ neighborhood y} = card {y ∈ Y . vert-adj

```

$y \ x \wedge \text{vert-adj } y \ x' / \text{card } Y$
using *measure-uniform-count-measure partitions-finite(2)* **by** *fastforce*
thus $P.\text{prob } \{y \in Y . \{x, x'\} \subseteq \text{neighborhood } y\} < c$ **using** *card-eq ltc* **by** *simp*
qed
then have $\bigwedge x \ x' . (x, x') \in B \implies P.\text{prob } \{y \in Y . (x, x') \in B' \ y\} < c$
by (*simp add: B-def B'-def bad-pair-set-def*)
then have prob: $\bigwedge p . p \in B \implies P.\text{prob } \{y \in Y . \text{indicator } (B' \ y) \ p = 1\} \leq c$
unfolding *indicator-def* **by** *fastforce*

have $d\text{simp}: (\text{density}^{\wedge 2} - (\text{density}^{\wedge 2} / (2 * c)) * c) * (\text{card } X)^{\wedge 2} = (\text{density}^{\wedge 2} / 2)$
 $* (\text{card } X)^{\wedge 2}$
using *assms* **by** (*simp add: algebra-simps*)
then have $gt0: (\text{density}^{\wedge 2} / 2) * (\text{card } X)^{\wedge 2} > 0$
using *dgt0* **by** (*metis density-simp division-ring-divide-zero half-gt-zero linorder-neqE-linordered-idom*
of-nat-less-0-iff of-nat-mult power2-eq-square zero-less-mult-iff)
have $Cgt0: (\text{density}^{\wedge 2} / (2 * c)) > 0$ **using** *dgt0* *assms* **by** *auto*
have $\bigwedge y . y \in Y \implies \text{card } (B' \ y) = (\sum p \in B . \text{indicator } (B' \ y) \ p)$
proof –
fix y **assume** $y \in Y$
then have $\text{neighborhood } y \subseteq X$ **by** (*simp add: neighborhood-subset-oppY*)
then have $ss: B' \ y \subseteq B$ **unfolding** *B-def B'-def bad-pair-set-def*
using *set-pairs-filter-subset* **by** *blast*
then show $\text{card } (B' \ y) = (\sum p \in B . \text{indicator } (B' \ y) \ p)$
using *card-set-ss-indicator[of B' y B]* *finB* **by** *auto*
qed
then have $P.\text{expectation } (\lambda y . \text{card } (B' \ y)) = P.\text{expectation } (\lambda y . (\sum p \in B .$
 $\text{indicator } (B' \ y) \ p))$
by (*metis (mono-tags, lifting) P.prob-space-axioms of-nat-sum partitions-finite(2)*

prob-space.expectation-uniform-count real-of-nat-indicator sum.cong)
also have $\dots = (\sum p \in B . P.\text{expectation } (\lambda y . \text{indicator } (B' \ y) \ p))$
by (*rule Bochner-Integration.integral-sum[of B ?M \lambda p y . indicator (B' y) p]*)
(auto simp add: P.integrable-uniform-count-measure-finite partitions-finite(2))
finally have $P.\text{expectation } (\lambda y . \text{card } (B' \ y)) = (\sum p \in B . P.\text{prob } \{y \in Y .$
 $\text{indicator } (B' \ y) \ p = 1\})$
using *P.expectation-finite-uniform-indicator[of Y B']* **using** *partitions-finite(2)*
by (*smt (verit, best) sum.cong*)
then have $P.\text{expectation } (\lambda y . \text{card } (B' \ y)) \leq (\sum p \in B . c)$
using *prob sum-mono[of B \lambda p . P.prob {y \in Y . indicator (B' y) p = 1} \lambda p.*
c]
by (*simp add: indicator-eq-1-iff*)
then have $lt1: P.\text{expectation } (\lambda y . \text{card } (B' \ y)) \leq c * (\text{card } B)$ **using** *finB*
by (*simp add: mult-of-nat-commute*)

have $c * (\text{card } B) \leq c * (\text{card } X)^{\wedge 2}$ **using** *assms card-B* **by** *auto*
then have $P.\text{expectation } (\lambda y . \text{card } (B' \ y)) \leq c * (\text{card } X)^{\wedge 2}$
using *lt1* **by** *linarith*
then have $\bigwedge C . C > 0 \implies C * P.\text{expectation } (\lambda y . \text{card } (B' \ y)) \leq C * c *$
 $(\text{card } X)^{\wedge 2}$

by auto
then have $\bigwedge C . C > 0 \implies (P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2) - C * (P.\text{expectation } (\lambda y. \text{card } (B' y))))$
 $\geq (\text{density}^2 * (\text{card } X)^2) - (C * c * (\text{card } X)^2)$
using *card-exp-gt diff-strict1-mono* **by** (*smt (verit)*)
then have $\bigwedge C . C > 0 \implies (P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2) - C * (P.\text{expectation } (\lambda y. \text{card } (B' y))))$
 $\geq (\text{density}^2 - C * c) * (\text{card } X)^2$
by (*simp add: field-simps*)

then have $(P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2) - (\text{density}^2 / (2 * c))) * (P.\text{expectation } (\lambda y. \text{card } (B' y)))$
 $\geq (\text{density}^2 - (\text{density}^2 / (2 * c)) * c) * (\text{card } X)^2$
using *Cgt0 assms* **by** *blast*
then have $P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2) - (\text{density}^2 / (2 * c)) * (P.\text{expectation } (\lambda y. \text{card } (B' y)))$
 $\geq (\text{density}^2 / 2) * (\text{card } X)^2$ **using** *dsimp* **by** *linarith*
then have $P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2) - (P.\text{expectation } (\lambda y. (\text{density}^2 / (2 * c)) * \text{card } (B' y)))$
 $\geq (\text{density}^2 / 2) * (\text{card } X)^2$ **by** *auto*
then have $P.\text{expectation } (\lambda y. (\text{card } (\text{neighborhood } y))^2 - ((\text{density}^2 / (2 * c)) * \text{card } (B' y)))$
 $\geq (\text{density}^2 / 2) * (\text{card } X)^2$
using *Bochner-Integration.integral-diff* [of *?M* $(\lambda y. (\text{card } (\text{neighborhood } y))^2)$ $(\lambda y. (\text{density}^2 / (2 * c)) * \text{card } (B' y))$]
P.integrable-uniform-count-measure-finite partitions-finite(2) **by** *fastforce*

then obtain *y* **where** *yin*: $y \in Y$ **and** *ineq*: $(\text{card } (\text{neighborhood } y))^2 - ((\text{density}^2 / (2 * c)) * \text{card } (B' y)) \geq (\text{density}^2 / 2) * (\text{card } X)^2$
using *P.expectation-obtains-ge partitions-finite*(2) **by** *blast*

let *?X'* = *neighborhood y*
have *ss*: $?X' \subseteq X$
using *yin* **by** (*simp add: neighborhood-subset-oppY*)
have *local.density*² / (2 * c) * *real* (card (B' y)) ≥ 0
using *assms density-simp* **by** *simp*
then have *d1*: $(\text{card } ?X')^2 \geq (\text{density}^2 / 2) * (\text{card } X)^2$
using *ineq* **by** *linarith*
then have $(\text{card } ?X') \geq \text{sqrt}(((\text{density}) * (\text{card } X))^2 / 2)$
by (*simp add: field-simps real-le-lsqrt*)
then have *den*: $((\text{card } ?X') \geq (\text{density} * (\text{card } X) / (\text{sqrt } 2)))$
by (*smt (verit, del-Insts) divide-nonneg-nonneg divide-nonpos-nonneg real-sqrt-divide*
real-sqrt-ge-0-iff real-sqrt-unique zero-le-power2)
have *xgt0*: $(\text{card } ?X') > 0$ **using** *dgt0 gt0*
using *d1 gr0I* **by** *force*
then have $(\text{card } ?X')^2 \geq (\text{density}^2 / (2 * c)) * \text{card } (B' y)$
using *gt0 ineq* **by** *simp*
then have $(\text{card } ?X')^2 / (\text{density}^2 / (2 * c)) \geq \text{card } (B' y)$

```

using Cgt0 by (metis mult.commute pos-le-divide-eq)
then have ((2 * c)/(density^2)) ≥ card (B' y)/(card ?X')^2
using pos-le-divide-eq xgt0 by (simp add: field-simps)
thus ?thesis using that[of ?X] den ss B'-def by auto
qed

```

The following technical probability lemma corresponds to Lemma 2.14 in Gowers's notes [3].

```

lemma (in prob-space) expectation-condition-card-1:
  fixes X::'a set and f::'a ⇒ real and δ::real
  assumes finite X and ∀ x ∈ X. f x ≤ 1 and M = uniform-count-measure X
and expectation f ≥ δ
  shows card {x ∈ X. (f x ≥ δ / 2)} ≥ δ * card X / 2
proof (cases δ ≥ 0)
  assume hδ: δ ≥ 0
  have ineq1: real (card (X - {x ∈ X. δ ≤ f x * 2})) * δ ≤ real (card X) * δ
  using card-mono assms Diff-subset hδ mult-le-cancel-right nat-le-linear of-nat-le-iff

  by (smt (verit, best))
  have ineq2: ∀ x ∈ X - {x. x ∈ X ∧ (f x ≥ δ/2)}. f x ≤ δ / 2 by auto
  have expectation f * card X = (∑ x ∈ X. f x)
  using assms(1) expectation-uniform-count assms(3) by force
  also have ... = (∑ x ∈ {x. x ∈ X ∧ (f x ≥ δ/2)}. f x)
    + (∑ x ∈ X - {x. x ∈ X ∧ (f x ≥ δ/2)}. f x)
  using assms
  by (metis (no-types, lifting) add.commute mem-Collect-eq subsetI sum.subset-diff)
  also have ... ≤ (∑ x ∈ {x. x ∈ X ∧ (f x ≥ δ/2)}. 1) +
    (∑ x ∈ X - {x. x ∈ X ∧ (f x ≥ δ/2)}. δ / 2)
  using assms sum-mono ineq2 by (smt (verit, ccfv-SIG) mem-Collect-eq)
  also have ... ≤ card ({x. x ∈ X ∧ (f x ≥ δ/2)}) + (card X) * δ / 2
  using ineq1 by auto
  finally have δ * card X ≤ card {x. x ∈ X ∧ (f x ≥ δ/2)} + (δ/2)*(card X)
  using ineq1 mult-of-nat-commute assms(4) mult-right-mono le-trans
  by (smt (verit, del-Insts) of-nat-0-le-iff times-divide-eq-left)
  then show ?thesis by auto
next
  assume ¬ δ ≥ 0
  thus ?thesis by (smt (verit, del-Insts) divide-nonpos-nonneg mult-nonpos-nonneg
of-nat-0-le-iff)
qed

```

The following technical probability lemma corresponds to Lemma 2.15 in Gowers's notes.

```

lemma (in prob-space) expectation-condition-card-2:
  fixes X::'a set and β::real and α::real and f:: 'a ⇒ real
  assumes finite X and ∧ x. x ∈ X ⇒ f x ≤ 1 and β > 0 and α > 0
and expectation f ≥ 1 - α and M = uniform-count-measure X
  shows card {x ∈ X. f x ≥ 1 - β} ≥ (1 - α / β) * card X

```

proof–
have $hcard$: $card \{x \in X. 1 - \beta \leq f x\} \leq card X$ **using** *card-mono* *assms(1)*
by *fastforce*
have $h\beta$: $\forall x \in X - \{x. x \in X \wedge (f x \geq 1 - \beta)\}. f x \leq 1 - \beta$ **by** *auto*
have $expectation f * card X = (\sum x \in X. f x)$
using *assms(1)* *expectation-uniform-count* *assms(6)* **by** *force*
then have $(1 - \alpha) * card X \leq (\sum x \in X. f x)$ **using** *assms*
by *(metis mult.commute sum-bounded-below sum-constant)*
also have $... = (\sum x \in \{x. x \in X \wedge (f x \geq 1 - \beta)\}. f x) +$
 $(\sum x \in X - \{x. x \in X \wedge (f x \geq 1 - \beta)\}. f x)$ **using** *assms*
by *(metis (no-types, lifting) add.commute mem-Collect-eq subsetI sum.subset-diff)*
also have $... \leq (\sum x \in \{x. x \in X \wedge (f x \geq 1 - \beta)\}. 1) +$
 $(\sum x \in X - \{x. x \in X \wedge (f x \geq 1 - \beta)\}. (1 - \beta))$
using *assms* $h\beta$ *sum-mono* **by** *(smt (verit, ccfv-SIG) mem-Collect-eq)*
also have $... = card \{x. x \in X \wedge (f x \geq 1 - \beta)\} + (1 - \beta) * card (X - \{x. x$
 $\in X \wedge (f x \geq 1 - \beta)\})$
by *auto*
also have $... = (card \{x. x \in X \wedge (f x \geq 1 - \beta)\} +$
 $card (X - \{x. x \in X \wedge (f x \geq 1 - \beta)\})) - \beta * card (X - \{x. x \in X \wedge (f x \geq$
 $1 - \beta)\})$
using *left-diff-distrib*
by *(smt (verit, ccfv-threshold) mult.commute mult.right-neutral of-nat-1 of-nat-add*
of-nat-mult)
also have heq : $... = card X - \beta * card (X - \{x. x \in X \wedge (f x \geq 1 - \beta)\})$
using *assms(1)* *card-Diff-subset[of {x. x \in X \wedge (f x \geq 1 - \beta)} X]* $hcard$ **by**
auto
finally have $(1 - \alpha) * card X \leq card X - \beta * card (X - \{x. x \in X \wedge (f x \geq$
 $1 - \beta)\})$ **by** *blast*
then have $-(1 - \alpha) * card X + card X \geq \beta * card (X - \{x. x \in X \wedge (f x \geq$
 $1 - \beta)\})$ **by** *linarith*
then have $- card X + \alpha * card X + card X \geq \beta * card(X - \{x. x \in X \wedge (f x$
 $\geq 1 - \beta)\})$
using *add.assoc* *add.commute*
add.right-neutral *add-0* *add-diff-cancel-right'* *add-diff-eq* *add-uminus-conv-diff*
diff-add-cancel
distrib-right *minus-diff-eq* *mult.commute* *mult-1* *of-int-minus* *of-int-of-nat-eq*
uminus-add-conv-diff
cancel-comm-monoid-add-class.diff-cancel **by** *(smt (verit, del-insts) mult-cancel-right2)*
then have $\alpha * card X \geq \beta * card(X - \{x. x \in X \wedge (f x \geq 1 - \beta)\})$ **by** *auto*
then have $\alpha * card X / \beta \geq card(X - \{x. x \in X \wedge (f x \geq 1 - \beta)\})$ **using**
assms
by *(smt (verit, ccfv-SIG) mult.commute pos-divide-less-eq)*
then show *?thesis* **by** *(smt (verit) heq combine-common-factor left-diff-distrib'*
mult-of-nat-commute
nat-mult-1-right *of-nat-1* *of-nat-add* *of-nat-mult* *times-divide-eq-left* *scale-minus-left)*
qed

The following lemma corresponds to Lemma 2.16 in Gowers's notes [3].
For the proof, we will apply Lemma 2.13 (*proportion-bad-pairs-subset-bipartite*),
the technical probability Lemmas 2.14 (*expectation-condition-card-1*) and

2.15 (*expectation-condition-card-2*) as well as background material on graphs with loops and bipartite graphs that was previously presented.

lemma (in *fin-bipartite-graph*) *walks-of-length-3-subsets-bipartite*:

obtains X' **and** Y' **where** $X' \subseteq X$ **and** $Y' \subseteq Y$ **and**

$\text{card } X' \geq (\text{edge-density } X \ Y)^2 * \text{card } X / 16$ **and**

$\text{card } Y' \geq \text{edge-density } X \ Y * \text{card } Y / 4$ **and**

$\forall x \in X'. \forall y \in Y'. \text{card } \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\} \geq$
 $(\text{edge-density } X \ Y)^6 * \text{card } X * \text{card } Y / 2^{13}$

proof (*cases edge-density X Y > 0*)

let $?\delta = \text{edge-density } X \ Y$

assume $h\delta: ?\delta > 0$

interpret $P1: \text{prob-space uniform-count-measure } X$

by (*simp add: X-not-empty partitions-finite(1) prob-space-uniform-count-measure*)

have $hP1exp: P1.\text{expectation } (\lambda x. \text{degree-normalized } x \ Y) \geq ?\delta$

using $P1.\text{expectation-uniform-count partitions-finite sum-degree-normalized-X-density}$

by *auto*

let $?X1 = \{x \in X. (\text{degree-normalized } x \ Y \geq ?\delta/2)\}$

have $hX1X: ?X1 \subseteq X$ **and** $hX1card: \text{card } ?X1 \geq ?\delta * (\text{card } X)/2$

and $hX1degree: \forall x \in ?X1. \text{degree-normalized } x \ Y \geq ?\delta / 2$ **using**

$P1.\text{expectation-condition-card-1 partitions-finite degree-normalized-le-1 hP1exp}$

by *auto*

have $hX1cardpos: \text{card } ?X1 > 0$ **using** $hX1card \ h\delta \ X\text{-not-empty}$

by (*smt (verit, del-insts) divide-divide-eq-right divide-le-0-iff density-simp gr0I less-eq-real-def mult-is-0 not-numeral-le-zero of-nat-le-0-iff of-nat-less-0-iff*)

interpret $H: \text{fin-bipartite-graph } (?X1 \cup Y) \{e \in E. e \subseteq (?X1 \cup Y)\} ?X1 \ Y$

proof (*unfold-locales, simp add: partitions-finite*)

have $\text{disjoint } \{?X1, Y\}$ **using** $hX1X \ \text{partition-on-def } \text{partition}$

by (*metis (no-types, lifting) disjnt-subset2 disjnt-sym ne pairwise-insert singletonD*)

moreover have $\{\} \notin \{?X1, Y\}$ **using** $hX1cardpos \ Y\text{-not-empty}$

by (*metis (no-types, lifting) card.empty insert-iff neq0-conv singleton-iff*)

ultimately show $\text{partition-on } (?X1 \cup Y) \{?X1, Y\}$ **using** partition-on-def

by *auto*

next

show $?X1 \neq Y$ **using** $ne \ \text{partition}$ **by** (*metis Int-absorb1 Y-not-empty hX1X part-intersect-empty*)

next

show $\bigwedge e. e \in \{e \in E. e \subseteq ?X1 \cup Y\} \implies e \in \text{all-bi-edges } \{x \in X. \text{edge-density } X \ Y / 2 \leq$

$\text{degree-normalized } x \ Y\} \ Y$

using $Un\text{-iff } Y\text{-verts-not-adj edge-betw-indiv in-mono insert-subset mem-Collect-eq}$

$\text{subset-refl that vert-adj-def all-bi-edges-def[of } ?X1 \ Y] \ \text{in-mk-uedge-img-iff}$

by (*smt (verit, ccfv-threshold) all-edges-betw-I all-edges-between-bi-subset*)

next

show $\text{finite } (?X1 \cup Y)$ **using** $hX1X$ **by** (*simp add: partitions-finite(1) partitions-finite(2)*)

qed

have *neighborhood-unchanged*: $\forall x \in ?X1. \text{neighbors-ss } x \ Y = H.\text{neighbors-ss } x \ Y$
using *neighbors-ss-def* *H.neighbors-ss-def* *vert-adj-def* *H.vert-adj-def* **by** *auto*
then have *degree-unchanged*: $\forall x \in ?X1. \text{degree } x = H.\text{degree } x$
using *H.degree-neighbors-ssX* *degree-neighbors-ssX* **by** *auto*
have *hHdensity*: $(H.\text{edge-density } ?X1 \ Y) \geq ?\delta / 2$
proof–
have $?\delta / 2 = (\sum x \in ?X1. (?\delta / 2)) / \text{card } ?X1$ **using** *hX1cardpos* **by** *auto*
also have $\dots \leq (\sum x \in ?X1. \text{degree-normalized } x \ Y) / \text{card } ?X1$
using *sum-mono* *hX1degree* *hX1cardpos* *divide-le-cancel*
by (*smt* (*z3*) *H.X-not-empty* *H.partitions-finite*(1)
calculation *divide-pos-pos* *hδ* *sum-pos* *zero-less-divide-iff*)
also have $\dots = (H.\text{edge-density } ?X1 \ Y)$
using *H.degree-normalized-def* *degree-normalized-def* *degree-unchanged* *sum.cong*
H.degree-neighbors-ssX *degree-neighbors-ssX* *H.sum-degree-normalized-X-density*
by *auto*
finally show *?thesis* **by** *simp*
qed
have *hδ3pos*: $?\delta^3 / 128 > 0$ **using** *hδ* **by** *auto*
then obtain *X2* **where** *hX2subX1*: $X2 \subseteq ?X1$ **and** *hX2card*: $\text{card } X2 \geq$
 $(H.\text{edge-density } ?X1 \ Y) * (\text{card } ?X1) / (\text{sqrt } 2)$ **and** *hX2badtemp*: $(\text{card } (H.\text{bad-pair-set } X2 \ Y \ (?\delta^3 / 128))) / \text{real } ((\text{card } X2)^2)$
 $\leq 2 * (?\delta^3 / 128) / (H.\text{edge-density } ?X1 \ Y)^2$ **using** *H.proportion-bad-pairs-subset-bipartite*
by *blast*
have $(H.\text{edge-density } ?X1 \ Y) * (\text{card } ?X1) / (\text{sqrt } 2) > 0$ **using** *hHdensity*
hX1cardpos *hδ* *hX2card*
by *auto*
then have *hX2cardpos*: $\text{card } X2 > 0$ **using** *hX2card* **by** *auto*
then have *hX2finite*: *finite* *X2* **using** *card-ge-0-finite* **by** *auto*
have *hX2bad*: $(\text{card } (H.\text{bad-pair-set } X2 \ Y \ (?\delta^3 / 128))) \leq (?\delta / 16) * (\text{card } X2)^2$
proof–
have *hpos*: $\text{real } ((\text{card } X2)^2) > 0$ **using** *hX2cardpos* **by** *auto*
have *trivial*: $(3::\text{nat}) - 2 = 1$ **by** *simp*
then have *hδpow*: $?\delta^3 / ?\delta^2 = ?\delta^1$ **using** *power-diff* *hδ*
by (*metis* *div-greater-zero-iff* *less-numeral-extra*(3) *numeral-Bit1-div-2* *zero-less-numeral*)

have $\text{card } (H.\text{bad-pair-set } X2 \ Y \ (?\delta^3 / 128)) \leq (2 * (?\delta^3 / 128) /$
 $(H.\text{edge-density } ?X1 \ Y)^2) * (\text{card } X2)^2$ **using** *hX2badtemp* *hX2cardpos* **by** (*simp* *add*: *field-simps*)
also have $\dots \leq (2 * (?\delta^3 / 128) / (?\delta / 2)^2) * (\text{card } X2)^2$
using *hδ* *hHdensity* *divide-left-mono* *frac-le* *hpos* **by** (*smt* (*verit*) *divide-pos-pos*

edge-density-ge0 *le-divide-eq* *power-mono* *zero-le-divide-iff* *zero-less-power*)
also have $\dots = (?\delta^3 / ?\delta^2) * (1/16) * (\text{card } X2)^2$ **by** (*simp* *add*:
field-simps)
also have $\dots = (?\delta / 16) * (\text{card } X2)^2$ **using** *hδpow* **by** *auto*
finally show *?thesis* **by** *simp*

```

qed
let ?E-loops = mk-edge ‘ {(x, x') | x x'. x ∈ X2 ∧ x' ∈ X2 ∧
  (H.codegree-normalized x x' Y) ≥ ?δ ^ 3 / 128}
interpret Γ: ulgraph X2 ?E-loops
proof(unfold-locales)
  show ∧e. e ∈ ?E-loops ⇒ e ⊆ X2 by auto
next
  have ∧a b. a ∈ X2 ⇒ b ∈ X2 ⇒ 0 < card {a, b}
    by (meson card-0-eq finite.emptyI finite-insert insert-not-empty neq0-conv)
  moreover have ∧ a b. a ∈ X2 ⇒ b ∈ X2 ⇒ card {a, b} ≤ 2
    by (metis card-2-iff dual-order.refl insert-absorb2 is-singletonI
      is-singleton-altdef one-le-numeral)
  ultimately show ∧e. e ∈ ?E-loops ⇒ 0 < card e ∧ card e ≤ 2 by auto
qed
have hΓ-edges: ∧ a b. a ∈ X2 ⇒ b ∈ X2 ⇒
  {a, b} ∈ ?E-loops ↔ H.codegree-normalized a b Y ≥ ?δ^3/128
proof
  fix a b assume {a, b} ∈ ?E-loops
  then show H.codegree-normalized a b Y ≥ ?δ^3/128
    using in-mk-uedge-img-iff[of a b {(x, x') | x x'. x ∈ X2 ∧ x' ∈ X2 ∧
  (H.codegree-normalized x x' Y) ≥ ?δ ^ 3 / 128}] doubleton-eq-iff H.codegree-normalized-sym

    by auto
  next
  fix a b assume a ∈ X2 and b ∈ X2 and H.codegree-normalized a b Y ≥
  ?δ^3/128
  then show {a, b} ∈ ?E-loops using in-mk-uedge-img-iff[of a b {(x, x') | x
  x'. x ∈ X2 ∧
  x' ∈ X2 ∧ (H.codegree-normalized x x' Y) ≥ ?δ ^ 3 / 128}] H.codegree-normalized-sym
by auto
qed
interpret P2: prob-space uniform-count-measure X2
  using hX2finite hX2cardpos prob-space-uniform-count-measure by fastforce
  have hP2exp: P2.expectation (λ x. Γ.degree-normalized x X2) ≥ 1 - ?δ / 16
proof-
  have hΓall: Γ.all-edges-between X2 X2 = (X2 × X2) - H.bad-pair-set X2 Y
  (?δ^3 / 128)
proof
  show Γ.all-edges-between X2 X2 ⊆ X2 × X2 - H.bad-pair-set X2 Y (?δ ^
  3 / 128)
proof
  fix x assume x ∈ Γ.all-edges-between X2 X2
  then obtain a b where a ∈ X2 and b ∈ X2 and x = (a, b) and
  H.codegree-normalized a b Y ≥ ?δ^3 / 128
  using Γ.all-edges-between-def in-mk-uedge-img-iff hΓ-edges
by (smt (verit, del-insts) Γ.all-edges-betw-D3 Γ.wellformed-alt-snd edge-density-commute

    mk-edge.cases mk-edge.simps that)
  then show x ∈ X2 × X2 - H.bad-pair-set X2 Y (?δ ^ 3 / 128)

```

```

    using H.bad-pair-set-def H.bad-pair-def by auto
  qed
next
  show  $X2 \times X2 - H.bad-pair-set X2 Y (?\delta \wedge 3 / 128) \subseteq \Gamma.all-edges-between X2 X2$ 
    using H.bad-pair-set-def H.bad-pair-def  $\Gamma.all-edges-between-def$   $h\Gamma-edges$  by
  auto
  qed
  then have  $h\Gamma all-le: card (\Gamma.all-edges-between X2 X2) \geq (1 - ?\delta / 16) * (card X2 * card X2)$ 
  proof-
    have  $hbadsb: H.bad-pair-set X2 Y (?\delta \wedge 3 / 128) \subseteq X2 \times X2$  using
  H.bad-pair-set-def by auto
    have  $(1 - ?\delta / 16) * (card X2 * card X2) = card (X2 \times X2) - ?\delta / 16 * (card X2) \wedge 2$ 
    using card-cartesian-product power2-eq-square
    by (metis Rings.ring-distrib(4) more-arith-simps(6) mult-of-nat-commute)
    also have  $\dots \leq card (X2 \times X2) - card (H.bad-pair-set X2 Y (?\delta \wedge 3 / 128))$ 
    using hX2bad by auto
    also have  $\dots = card (X2 \times X2 - H.bad-pair-set X2 Y (?\delta \wedge 3 / 128))$  using
  card-Diff-subset
    finite-cartesian-product[of X2 X2] hX2finite hbadsb
    by (metis (mono-tags, lifting) finite-subset)
    finally show  $card(\Gamma.all-edges-between X2 X2) \geq (1 - ?\delta/16) * (card X2 * card X2)$ 
    using h $\Gamma all$  by simp
  qed
  have  $1 - ?\delta / 16 = ((1 - ?\delta / 16) * (card X2 * card X2)) / (card X2 * card X2)$ 
  using hX2cardpos by auto
  also have  $\dots \leq card (\Gamma.all-edges-between X2 X2) / (card X2 * card X2)$ 
  using h $\Gamma all-le$  hX2cardpos divide-le-cancel of-nat-less-0-iff by fast
  also have  $\dots = (\sum x \in X2. real (card (\Gamma.neighbors-ss x X2))) / card X2 / card X2$ 
  using  $\Gamma.card-all-edges-betw-neighbor$ [of X2 X2] hX2finite by (auto simp add:
  field-simps)
  also have  $\dots = (\sum x \in X2. \Gamma.degree-normalized x X2) / card X2$ 
  unfolding  $\Gamma.degree-normalized-def$ 
  using sum-divide-distrib[of  $\lambda x. real (card (\Gamma.neighbors-ss x X2))$ ] X2 card
  X2] by auto
  also have  $\dots = P2.expectation (\lambda x. \Gamma.degree-normalized x X2)$ 
  using P2.expectation-uniform-count hX2finite by auto
  finally show ?thesis by simp
qed
let ?X' =  $\{x \in X2. \Gamma.degree-normalized x X2 \geq 1 - ?\delta / 8\}$ 
have  $hX'subX2: ?X' \subseteq X2$  by blast
have  $hX'cardX2: card ?X' \geq card X2 / 2$  using hX2finite divide-self h $\delta$ 
P2.expectation-condition-card-2 [of X2 ( $\lambda x. \Gamma.degree-normalized x X2$ )] ? $\delta / 8$ 

```

```

? $\delta$  / 16]
  hP2exp  $\Gamma$ .degree-normalized-le-1 by auto
interpret P3: prob-space uniform-count-measure Y
by (simp add: Y-not-empty partitions-finite(2) prob-space-uniform-count-measure)
have hP3exp: P3.expectation ( $\lambda$  y. H.degree-normalized y X2)  $\geq$  ? $\delta$  / 2
proof -
  have hHdegree-normalized:  $\bigwedge$  x. x  $\in$  X2  $\implies$  (? $\delta$  / 2)  $\leq$  H.degree-normalized
x Y
  using hX1degree degree-normalized-def H.degree-normalized-def neighbor-
hood-unchanged
  hX2subX1 subsetD by (metis (no-types, lifting))
  have ? $\delta$  / 2 = ( $\sum$  x  $\in$  X2. (? $\delta$  / 2)) / card X2 using hX2cardpos by auto
  also have ...  $\leq$  ( $\sum$  x  $\in$  X2. real (card (H.neighbors-ss x Y))) / card Y / card
X2
  using hHdegree-normalized sum-mono divide-le-cancel hX2cardpos of-nat-0-le-iff
H.degree-normalized-def sum.cong sum-divide-distrib by (smt (verit, best))
  also have ... = (card (H.all-edges-between Y X2)) / card X2 / card Y
  using H.card-all-edges-between-commute H.card-all-edges-betw-neighbor hX2fi-
nite
  H.partitions-finite(2) by auto
  also have ... = ( $\sum$  y  $\in$  Y. real (card(H.neighbors-ss y X2))) / card X2 / card
Y using
  H.card-all-edges-betw-neighbor hX2finite H.partitions-finite(2) by auto
  also have ... = ( $\sum$  y  $\in$  Y. H.degree-normalized y X2) / card Y using
H.degree-normalized-def
  sum.cong sum-divide-distrib by (smt (verit, best))
  also have ... = P3.expectation ( $\lambda$  y. H.degree-normalized y X2)
  using P3.expectation-uniform-count H.partitions-finite(2) by auto
  finally show ?thesis by linarith
qed
let ?Y' = {x  $\in$  Y. H.degree-normalized x X2  $\geq$  ? $\delta$  / 4}
have hY'subY: ?Y'  $\subseteq$  Y by blast
then have hY'card: card ?Y'  $\geq$  ? $\delta$  * card Y / 4
  using P3.expectation-condition-card-1[of Y ( $\lambda$  y. H.degree-normalized y X2)
? $\delta$  / 2] H.partitions-finite(2)
  hP3exp H.degree-normalized-le-1 by auto

have hX2adjcard:  $\bigwedge$  x y. x  $\in$  ?X'  $\implies$  y  $\in$  ?Y'  $\implies$ 
card {x'  $\in$  X2.  $\Gamma$ .vert-adj x x'  $\wedge$  vert-adj y x'}  $\geq$  ? $\delta$  / 8 * card X2
proof -
  fix x y assume hx: x  $\in$  ?X' and hy: y  $\in$  ?Y'
  have hinter: {x'  $\in$  X2.  $\Gamma$ .vert-adj x x'  $\wedge$  vert-adj y x'} =
{x'  $\in$  X2.  $\Gamma$ .vert-adj x x'}  $\cap$  {x'  $\in$  X2. vert-adj y x'} by auto
  have huncardX2: card ({x'  $\in$  X2.  $\Gamma$ .vert-adj x x'}  $\cup$  {x'  $\in$  X2. vert-adj y x'})
 $\leq$  card X2
  using card-mono hX2finite by fastforce
  have fin1: finite {x'  $\in$  X2.  $\Gamma$ .vert-adj x x'} and fin2: finite {x'  $\in$  X2. vert-adj
y x'}
  using hX2finite by auto

```

have $\{x' \in X2. \Gamma.\text{vert-adj } x \ x'\} = \Gamma.\text{neighbors-ss } x \ X2$
using $\Gamma.\text{vert-adj-def vert-adj-def } \Gamma.\text{neighbors-ss-def } hX2\text{sub}X1 \ \Gamma.\text{neighbors-ss-def}$
by *auto*
then have $h\text{card}1: \text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x'\} \geq (1 - ?\delta/8) * \text{card } X2$
using *hx*
 $\Gamma.\text{degree-normalized-def divide-le-eq } hX2\text{cardpos}$ **by** (*simp add: hX2card le-divide-eq*)
have $\{x' \in X2. \text{vert-adj } y \ x'\} = H.\text{neighbors-ss } y \ X2$ **using** $H.\text{vert-adj-def vert-adj-def}$
 $H.\text{neighbors-ss-def } hy \ hY'\text{sub}Y \ hX2\text{sub}X1 \ H.\text{neighbors-ss-def}$ **by** *auto*
then have $h\text{card}2: \text{card } \{x' \in X2. \text{vert-adj } y \ x'\} \geq (?\delta / 4) * \text{card } X2$
using $hy \ H.\text{degree-normalized-def divide-le-eq } hX2\text{cardpos}$ **by** (*simp add: hX2card le-divide-eq*)
have $?\delta / 8 * \text{card } X2 = (1 - ?\delta / 8) * \text{card } X2 + ?\delta/4 * \text{card } X2 - \text{card } X2$
by (*simp add: algebra-simps*)
also have $\dots \leq \text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x'\} + \text{card } \{x' \in X2. \text{vert-adj } y \ x'\}$
 $\text{card } (\{x' \in X2. \Gamma.\text{vert-adj } x \ x'\} \cup \{x' \in X2. \text{vert-adj } y \ x'\})$
using $h\text{uncard}X2 \ h\text{card}1 \ h\text{card}2$ **by** *linarith*
also have $\dots = \text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}$
using $\text{card-Un-Int } fin1 \ fin2 \ hinter$ **by** *fastforce*
finally show $\text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} \geq ?\delta / 8 * \text{card } X2$
by *linarith*
qed
have $hY\text{pos}: \text{real } (\text{card } Y) > 0$ **using** $Y.\text{not-empty partitions-finite}(2)$ **by** *auto*
have $\bigwedge x \ y. x \in ?X' \implies y \in ?Y' \implies \text{card } \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\} \geq$
 $(?\delta \wedge 3 / 128 * (\text{card } Y)) * ((?\delta / 8) * (\text{card } X2))$
proof–
fix $x \ y$ **assume** $hx: x \in ?X'$ **and** $hy: y \in ?Y'$
then have $hxV: x \in V$ **and** $hyV: y \in V$ **using** $hY'\text{sub}Y \ hX'\text{sub}X2 \ hX2\text{sub}X1 \ hX1X \text{partitions-ss}(1)$
 $\text{partitions-ss}(2) \ \text{subset}D$ **by** *auto*
define $f:: 'a \Rightarrow 'a \text{ list set}$ **where** $f \equiv (\lambda a. ((\lambda z. z @ [y]) ' \{p. \text{connecting-path } x \ a \ p \wedge \text{walk-length } p = 2\}))$
have $h\text{-norm}: \bigwedge a. a \in \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} \implies \text{codegree-normalized } x \ a \ Y \geq ?\delta \wedge 3 / 128$
using $\Gamma.\text{vert-adj-def codegree-normalized-sym } hx \ hX'\text{sub}X2 \ \text{subset}D$
 $\text{codegree-normalized-alt}X \ H.\text{codegree-normalized-alt}X \ \text{neighborhood-unchanged}$
 $h\Gamma.\text{edges } hX2\text{sub}X1 \ hX1X$ **by** (*smt (verit, del-insts) mem-Collect-eq*)
have $\text{inj-concat}: \text{inj } (\lambda z. z @ [y])$ **using** inj-def **by** *blast*
then have $\text{card-f-eq}: \bigwedge a. \text{card } (f \ a) = \text{card } \{p. \text{connecting-path } x \ a \ p \wedge \text{walk-length } p = 2\}$
using $f.\text{def } \text{card-image } \text{inj-eq}$ **by** (*smt (verit) inj-onI*)
then have $\text{card-f-ge}: \bigwedge a. a \in \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} \implies \text{card } (f \ a) \geq ?\delta \wedge 3 / 128 * \text{card } Y$

using *codegree-is-path-length-two codegree-normalized-def hYpos f-def h-norm*
by (*simp add: field-simps*)
have *f-disjoint*: $\text{pairwise } (\lambda s t. \text{disjnt } (f s) (f t)) \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$
proof (*intro pairwiseI*)
fix *s t* **assume** $s \in \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$ **and**
 $t \in \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$ **and** $s \neq t$
moreover **have** $\bigwedge a l. l \in f a \implies l! 2 = a$
proof–
fix *a l* **assume** $l \in f a$
then **obtain** *z* **where** *hz*: $z \in \{p. \text{connecting-path } x a p \wedge \text{walk-length } p = 2\}$ **and** *hlz*: $l = z @ [y]$
using *f-def* **by** *blast*
then **have** $z! 2 = a$ **using** *walk-length-conv connecting-path-def last-conv-nth*
by (*metis (mono-tags, lifting) diff-add-inverse length-tl list.sel(2) mem-Collect-eq*

 $\text{nat-1-add-1 one-eq-numeral-iff rel-simps}(18))$
then **show** $l! 2 = a$ **using** *hlz nth-append hz walk-length-conv less-diff-conv*
mem-Collect-eq
by (*metis (mono-tags, lifting) nat-1-add-1 one-less-numeral-iff rel-simps(9)*)
qed
ultimately **show** $\text{disjnt } (f s) (f t)$ **by** (*metis disjnt-iff*)
qed
have *hwalk-subset*: $\{p. \text{connecting-walk } x k p \wedge \text{walk-length } p = n\} \subseteq \{p. \text{set } p \subseteq V \wedge \text{length } p = n + 1\}$ **for** *n k*
using *connecting-walk-def is-walk-def walk-length-conv* **by** *auto*
have *finite-walk*: $\text{finite } \{p. \text{connecting-walk } x k p \wedge \text{walk-length } p = n\}$ **for** *n k*
using *finV finite-lists-length-eq finite-subset hwalk-subset[of k n] rev-finite-subset*
by *blast*
have *f-finite*: $\bigwedge A. A \in (f ' \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}) \implies \text{finite } A$
proof–
fix *A* **assume** $A \in (f ' \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\})$
then **obtain** *a* **where** $a \in \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$ **and**
hA: $A = f a$ **by** *blast*
have $\{p. \text{connecting-path } x a p \wedge \text{walk-length } p = 2\} \subseteq \{p. \text{connecting-walk } x a p \wedge \text{walk-length } p = 2\}$
using *connecting-path-walk* **by** *blast*
then **have** $\text{finite } \{p. \text{connecting-path } x a p \wedge \text{walk-length } p = 2\}$
using *finite-walk finite-subset connecting-path-walk* **by** *blast*
then **show** $\text{finite } A$ **using** *f-def hA* **by** *auto*
qed
moreover **have** *f-image-sub*:
 $(\bigcup x \in \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}. f x) \subseteq \{p. \text{connecting-walk } x y p \wedge \text{walk-length } p = 3\}$
proof (*intro Union-least*)
fix *X* **assume** $X \in f ' \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$
then **obtain** *a* **where** *ha*: $a \in \{x' \in X2. \Gamma.\text{vert-adj } x x' \wedge \text{vert-adj } y x'\}$ **and**
haX: $f a = X$ **by** *blast*

have $\bigwedge z. \text{connecting-path } x \ a \ z \implies \text{walk-length } z = 2 \implies \text{connecting-walk } x \ y \ (z \ @ \ [y])$
proof–
fix z **assume** $hpath: \text{connecting-path } x \ a \ z$ **and** $hlen: \text{walk-length } z = 2$
then obtain y' **where** $z \ ! \ 1 = y'$ **by** *blast*
then have $hz: z = [x, y', a]$ **using** *list2-middle-singleton walk-length-conv connecting-path-def hpath hlen add-diff-cancel-left' append-butlast-last-id butlast.simps connecting-path-walk connecting-walk-def diff-diff-add diff-le-self*

is-walk-not-empty is-walk-tl last-ConsL last-tl list.expand list.sel list.simps(3)

nat-1-add-1 le-imp-diff-is-add
by (*metis (no-types, lifting) arith-simps(45) arithmetic-simps(2) numerals(1)*)
moreover have $hwalk: \text{connecting-walk } x \ a \ z$ **using** *connecting-path-walk hpath* **by** *blast*
then show $\text{connecting-walk } x \ y \ (z \ @ \ [y])$ **using** *connecting-walk-def is-walk-def*

connecting-path-def is-gen-path-def is-walk-def ha hz hwalk hyV vert-adj-sym vert-adj-def **by** *auto*
qed
then show $X \subseteq \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\}$
using *haX f-def walk-length-conv* **by** *auto*
qed
ultimately have $hUn-le: \text{card } (\bigcup x \in \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}. f \ x) \leq \text{card } \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\}$
using *card-mono finite-walk[of y 3]* **by** *blast*
have *disjoint* ($f \ ' \ \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}$) **using** *f-disjoint pairwise-def*
pairwise-image[of disjnt f \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}] **by**
(*metis (mono-tags, lifting)*)
then have $\text{card } (\bigcup (f \ ' \ \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\})) = \text{sum } \text{card } (f \ ' \ \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\})$
using *card-Union-disjoint[of f \ ' \ \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}]*
f-finite **by** *blast*
also have $\dots = (\sum a \in \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\}. \text{card } (f \ a))$
using *sum-card-image[OF - f-disjoint] hX2finite finite-subset* **by** *fastforce*
also have $\dots \geq \text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} * ?\delta^{\wedge 3} / 128 * \text{card } Y$
using *sum-mono[of \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} (\lambda a. ?\delta^{\wedge 3} / 128 * \text{card } Y) (\lambda a. \text{card } (f \ a))]*
card-f-ge **by** *auto*
finally have $\text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} * ?\delta^{\wedge 3} / 128 * \text{card } Y \leq \text{card } \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\}$
using *hUn-le* **by** *linarith*
then have $(?\delta^{\wedge 3} / 128 * (\text{card } Y)) * \text{card } \{x' \in X2. \Gamma.\text{vert-adj } x \ x' \wedge \text{vert-adj } y \ x'\} \leq \text{card } \{p. \text{connecting-walk } x \ y \ p \wedge \text{walk-length } p = 3\}$
by *argo*
then show $(?\delta^{\wedge 3} / 128 * (\text{card } Y)) * ((?\delta / 8) * (\text{card } X2)) \leq \text{card } \{p.$

connecting-walk $x y p \wedge \text{walk-length } p = 3$
using $hX2adjcard[OF \ hx \ hy] \ hYpos \ mult\text{-left-mono} \ h\delta 3pos \ mult\text{-pos-pos}$
by (*smt* (*verit*, *del-insts*))
qed
moreover have $hX2cardX: \text{card } X2 \geq (\delta^2 / 8) * (\text{card } X)$
proof–
have $\text{card } X2 \geq (H.\text{edge-density } ?X1 \ Y / \text{sqrt } 2) * (\text{card } ?X1)$
using $hX2card$ **by** (*simp* *add: algebra-simps*)
moreover have $(H.\text{edge-density } ?X1 \ Y / \text{sqrt } 2) * (\text{card } ?X1) \geq (\delta / (2 * \text{sqrt } 2)) * \text{card } ?X1$
using $hHdensity \ hX1cardpos$ **by** (*simp* *add: field-simps*)
moreover have $(\delta / (2 * \text{sqrt } 2)) * \text{card } ?X1 \geq (\delta / 4) * \text{card } ?X1$
using $\text{sqrt2-less-2} \ hX1cardpos \ h\delta$ **by** (*simp* *add: field-simps*)
moreover have $(\delta / 4) * \text{card } ?X1 \geq (\delta / 4) * (\delta / 2 * \text{card } X)$
using $hX1card \ h\delta$ **by** (*simp* *add: field-simps*)
moreover have $(\delta / 4) * (\delta / 2 * \text{card } X) = (\delta^2 / 8) * (\text{card } X)$ **using**
power2-eq-square
by (*metis* (*no-types*, *opaque-lifting*) *Groups.mult-ac(2)* *Groups.mult-ac(3)*)
divide-divide-eq-left
num-double numeral-times-numeral times-divide-eq-left times-divide-eq-right
ultimately show *?thesis* **by** *linarith*
qed
moreover have $(\delta^3 / 128 * (\text{card } Y)) * ((\delta / 8) * (\text{card } X2)) \geq \delta^6 * \text{card } X * \text{card } Y / 2^{13}$
proof–
have *hinter*: $(\delta / 8) * (\delta^2 / 8 * \text{card } X) \leq (\delta / 8) * (\text{card } X2)$
using $hX2cardX \ h\delta$ **by** (*simp* *add: algebra-simps*)
have $\delta^6 * \text{real}(\text{card } X * \text{card } Y) / 2^{13} = \delta^3 * \delta * \delta^2 * \text{real}(\text{card } X * \text{card } Y) / (128 * 8 * 8)$ **by** *algebra*
also have $\dots = (\delta^3 / 128 * (\text{card } Y)) * ((\delta / 8) * (\delta^2 / 8 * \text{card } X))$
by *auto*
also have $\dots \leq (\delta^3 / 128 * (\text{card } Y)) * ((\delta / 8) * (\text{card } X2))$
using *hinter* $hYpos \ h\delta$ *power3-eq-cube*
by (*smt* (*verit*) $\langle 0 < \text{edge-density } X \ Y \ \delta^3 / 128 \rangle$ *mult-left-mono* *zero-compare-simps(6)*)
finally show *?thesis* **by** *simp*
qed
moreover have $hX'card: \text{card } ?X' \geq \delta^2 * \text{card } X / 16$ **using** $hX'cardX2$
 $hX2cardX$ **by** *auto*
moreover have $hX'subX: ?X' \subseteq X$ **using** $hX'subX2 \ hX2subX1 \ hX1X$ **by** *auto*
ultimately show *?thesis* **using** $hY'card \ hX'card \ hY'subY \ hX'subX$ **that**
by (*smt* (*verit*, *best*))
next
assume $\neg 0 < \text{edge-density } X \ Y$
then have $\text{edge-density } X \ Y = 0$ **by** (*smt* (*verit*, *ccfv-threshold*) *edge-density-ge0*)
then show *?thesis* **using** *that* **by** *auto*
qed

The following lemma corresponds to Lemma 2.17 in Gowers's notes [3].
Note that here we have set(*additive-energy* $A = 2 * c$ (instead of (*ad-*

ditive-energy $A = c$ as in the notes) and we are accordingly considering c -popular differences (instead of $c/2$ -popular differences as in the notes) so that we will still have $(\vartheta = \text{additive-energy } A / 2)$.

lemma *popular-differences-card*: **fixes** $A::'a$ set **and** $c::\text{real}$
assumes *finite* A **and** $A \subseteq G$ **and** *additive-energy* $A = 2 * c$
shows $\text{card} (\text{popular-diff-set } c A) \geq c * \text{card } A$

proof(*cases* $\text{card } A \neq 0$)
assume $hA: \text{card } A \neq 0$
have $hc: c \geq 0$ **using** *assms additive-energy-def of-nat-0-le-iff*
by (*smt (verit, best) assms(3) divide-nonneg-nonneg of-nat-0-le-iff*)
have $(2 * c) * (\text{card } A)^3 = (\sum d \in (\text{differenceset } A A). (f\text{-diff } d A)^2)$
using *assms f-diff-card-quadruple-set-additive-energy by auto*
also have $\dots = ((\sum d \in (\text{popular-diff-set } c A). (f\text{-diff } d A)^2))$
 $+ ((\sum d \in ((\text{differenceset } A A) - (\text{popular-diff-set } c A)). (f\text{-diff } d A)^2))$
using *popular-diff-set-def assms finite-minusset finite-sumset by (metis (no-types, lifting)*
add commute mem-Collect-eq subsetI sum.subset-diff)
also have $\dots \leq ((\text{card} (\text{popular-diff-set } c A)) * (\text{card } A)^2)$
 $+ c * \text{card } A * ((\sum d \in (\text{differenceset } A A - (\text{popular-diff-set } c A)). (f\text{-diff } d A)))$
proof –
have $\forall d \in ((\text{differenceset } A A) - (\text{popular-diff-set } c A)). (f\text{-diff } d A)^2 \leq$
 $(c * \text{card } A) * (f\text{-diff } d A)$
proof
fix d **assume** $hd1: d \in \text{differenceset } A A - \text{popular-diff-set } c A$
have $h\text{nonneg}: f\text{-diff } d A \geq 0$ **by** *auto*
have $\neg \text{popular-diff } d c A$ **using** $hd1$ *popular-diff-set-def* **by** *blast*
from this have $f\text{-diff } d A \leq c * \text{card } A$ **using** *popular-diff-def* **by** *auto*
thus $\text{real} ((f\text{-diff } d A)^2) \leq c * \text{real} (\text{card } A) * \text{real} (f\text{-diff } d A)$
using *power2-eq-square hnonneg mult-right-mono of-nat-0 of-nat-le-iff of-nat-mult*
by *metis*
qed
moreover have $\forall d \in (\text{differenceset } A A). f\text{-diff } d A \leq (\text{card } A)^2$
using *f-diff-def finite-minusset finite-sumset assms*
by (*metis f-diff-le-card le-antisym nat-le-linear power2-nat-le-imp-le*)
ultimately have $((\sum d \in ((\text{differenceset } A A) - \text{popular-diff-set } c A). (f\text{-diff } d A)^2)) \leq$
 $((\sum d \in ((\text{differenceset } A A) - \text{popular-diff-set } c A). (c * \text{card } A) * (f\text{-diff } d A)))$
using *assms finite-minusset finite-sumset sum-distrib-left sum-mono* **by** *fast-force*
then have $((\sum d \in ((\text{differenceset } A A) - \text{popular-diff-set } c A). (f\text{-diff } d A)^2))$
 \leq
 $(c * \text{card } A) * ((\sum d \in ((\text{differenceset } A A) - \text{popular-diff-set } c A). (f\text{-diff } d A)))$
by (*metis (no-types) of-nat-sum sum-distrib-left*)
moreover have $(\sum d \in \text{popular-diff-set } c A. (f\text{-diff } d A)^2) \leq$

$(\sum d \in \text{popular-diff-set } c \ A. (\text{card } A)^{\wedge 2})$ **using** *f-diff-le-card* *assms* *sum-mono*
assms *popular-diff-set-def*
by (*metis* (*no-types*, *lifting*) *power2-nat-le-eq-le*)
moreover then have *ddd*: $(\sum d \in \text{popular-diff-set } c \ A. (f\text{-diff } d \ A)^{\wedge 2}) \leq$
 $(\text{card } (\text{popular-diff-set } c \ A)) * (\text{card } A)^{\wedge 2}$
using *sum-distrib-right* **by** *simp*
ultimately show *?thesis* **by** *linarith*
qed
also have $\dots \leq ((\text{card } (\text{popular-diff-set } c \ A)) * (\text{card } A)^{\wedge 2}) + (c * \text{card } A) *$
 $(\text{card } A)^{\wedge 2}$
proof-
have $(\sum d \in (\text{differenceset } A \ A - \text{popular-diff-set } c \ A). (f\text{-diff } d \ A)) \leq$
 $(\sum d \in \text{differenceset } A \ A. (f\text{-diff } d \ A))$ **using** *DiffD1* *subsetI* *assms* *sum-mono2*

finite-minusset *finite-sumset* *zero-le* **by** *metis*
then have $(c * \text{card } A) * ((\sum d \in (\text{differenceset } A \ A - \text{popular-diff-set } c \ A).$
 $(f\text{-diff } d \ A)))$
 $\leq (c * \text{card } A) * (\text{card } A)^{\wedge 2}$
using *f-diff-card* *hc* *le0* *mult-left-mono* *of-nat-0* *of-nat-mono* *zero-le-mult-iff*
assms **by** *metis*
then show *?thesis* **by** *linarith*
qed
finally have $(2 * c) * (\text{card } A)^{\wedge 3} \leq ((\text{card } (\text{popular-diff-set } c \ A)) * (\text{card } A)^{\wedge 2})$
 $+$
 $(c * \text{card } A) * (\text{card } A)^{\wedge 2}$ **by** *linarith*
then have $(\text{card } (\text{popular-diff-set } c \ A)) \geq$
 $((2 * c) * (\text{card } A)^{\wedge 3} - (c * \text{card } A) * (\text{card } A)^{\wedge 2}) / ((\text{card } A)^{\wedge 2})$
using *hA* **by** (*simp* *add*: *field-simps*)
moreover have $((2 * c) * (\text{card } A)^{\wedge 3} - (c * \text{card } A) * (\text{card } A)^{\wedge 2}) / ((\text{card } A)^{\wedge 2})$
 $= 2 * c * \text{card } A - c * \text{card } A$
using *hA* **by** (*simp* *add*: *power2-eq-square* *power3-eq-cube*)
ultimately show *?thesis* **by** *linarith*
next
assume $\neg \text{card } A \neq 0$
thus *?thesis* **by** *auto*
qed

The following lemma corresponds to Lemma 2.18 in Gowers's notes [3]. It includes the key argument of the main proof and its proof applies Lemmas 2.16 (*walks-of-length-3-subsets-bipartite*) and 2.17 (*popular-differences-card*). In the proof we will use an appropriately defined bipartite graph as an intermediate/auxiliary construct so as to apply lemma *walks-of-length-3-subsets-bipartite*. As each vertex set of the bipartite graph is constructed to be a copy of a finite subset of an Abelian group, we need flexibility regarding types, which is what prompted the introduction and use of the new graph theory library [1] (that does not impose any type restrictions e.g. by representing vertices as natural numbers).

lemma *obtains-subsets-differenceset-card-bound*:

fixes $A::'a$ set **and** $c::real$
assumes finite A **and** $c>0$ **and** $A \neq \{\}$ **and** $A \subseteq G$ **and** additive-energy $A = 2 * c$
obtains B **and** A' **where** $B \subseteq A$ **and** $B \neq \{\}$ **and** $card\ B \geq c^4 * card\ A / 16$
and $A' \subseteq A$ **and** $A' \neq \{\}$ **and** $card\ A' \geq c^2 * card\ A / 4$
and $card\ (differenceset\ A'\ B) \leq 2^{13} * card\ A / c^{15}$

proof–
let $?X = A \times \{0::nat\}$
let $?Y = A \times \{1::nat\}$
let $?E = mk-edge\ \{(x, y) \mid x\ y.\ x \in ?X \wedge y \in ?Y \wedge (popular-diff\ (fst\ y \ominus\ fst\ x)\ c\ A)\}$
interpret $H: fin-bipartite-graph\ ?X \cup ?Y\ ?E\ ?X\ ?Y$
proof (*unfold-locales, auto simp add: partition-on-def assms(3) assms(1) disjoint-def*)
show $\{\} = A \times \{0\} \implies False$ **using** *assms(3)* **by** *auto*
next
show $\{\} = A \times \{Suc\ 0\} \implies False$ **using** *assms(3)* **by** *auto*
next
show $A \times \{0\} = A \times \{Suc\ 0\} \implies False$ **using** *assms(3)* **by** *fastforce*
next
fix $x\ y$ **assume** $x \in A$ **and** $y \in A$ **and** *popular-diff* $(y \ominus x)\ c\ A$
thus $\{(x, 0), (y, Suc\ 0)\} \in all-bi-edges\ (A \times \{0\})\ (A \times \{Suc\ 0\})$
using *all-bi-edges-def*[of $A \times \{0\}\ A \times \{Suc\ 0\}$]
by (*simp add: in-mk-edge-img*)
qed
have *edges1*: $\forall a \in A. \forall b \in A. (\{(a, 0), (b, 1)\} \in ?E \longleftrightarrow popular-diff\ (b \ominus a)\ c\ A)$
by (*auto simp add: in-mk-uedge-img-iff*)
have *hXA*: $card\ A = card\ ?X$ **by** (*simp add: card-cartesian-product*)
have *hYA*: $card\ A = card\ ?Y$ **by** (*simp add: card-cartesian-product*)
have *hA*: $card\ A \neq 0$ **using** *assms card-0-eq* **by** *blast*
have *edge-density*: $H.edge-density\ ?X\ ?Y \geq c^2$
proof–
define $f:: 'a \Rightarrow ('a \times nat)$ **edge set** **where** $f \equiv (\lambda x. \{\{(a, 0), (b, 1)\} \mid a\ b.\ a \in A \wedge b \in A \wedge b \ominus a = x\})$
have *f-disj*: *pairwise* $(\lambda s\ t. disjnt\ (f\ s)\ (f\ t))\ (popular-diff-set\ c\ A)$
proof (*intro pairwiseI*)
fix $x\ y$ **assume** *hx*: $x \in popular-diff-set\ c\ A$ **and** *hy*: $y \in popular-diff-set\ c\ A$
and *hxy*: $x \neq y$
show *disjnt* $(f\ x)\ (f\ y)$
proof–
have $\forall a. \neg (a \in f\ x \wedge a \in f\ y)$
proof (*intro allI notI*)
fix a **assume** $a \in f\ x \wedge a \in f\ y$
then obtain $z\ w$ **where** *hazw*: $a = \{(z, 0), (w, 1)\}$ **and** *hx*: $\{(z, 0), (w, 1)\} \in f\ x$
and *hy*: $\{(z, 0), (w, 1)\} \in f\ y$ **using** *f-def* **by** *blast*
have $w \ominus z = x$ **using** *f-def hx* **by** (*simp add: doubleton-eq-iff*)

```

    moreover have  $w \ominus z = y$  using f-def hy by (simp add: doubleton-eq-iff)
    ultimately show False using hxy by auto
  qed
  thus ?thesis using disjnt-iff by auto
  qed
  have f-sub-edges:  $\forall d \in \text{popular-diff-set } c \ A. (f \ d) \subseteq ?E$ 
    using popular-diff-set-def f-def edges1 by auto
  have f-union-sub:  $(\bigcup d \in \text{popular-diff-set } c \ A. (f \ d)) \subseteq ?E$  using popular-diff-set-def f-def edges1 by auto
  have f-disj2: disjoint (f ‘ (popular-diff-set c A)) using f-disj pairwise-image[of disjnt f popular-diff-set c A] by (simp add: pairwise-def)
  have f-finite:  $\bigwedge B. B \in f \ ‘ \text{popular-diff-set } c \ A \implies \text{finite } B$ 
    using finite-subset f-sub-edges H.fin-edges by auto
  have card-eq-f-diff:  $\forall d \in \text{popular-diff-set } c \ A. \text{card } (f \ d) = f\text{-diff } d \ A$ 
  proof
    fix d assume d  $\in \text{popular-diff-set } c \ A$ 
    define g::  $('a \times 'a) \Rightarrow ('a \times \text{nat}) \text{ edge}$  where g =  $(\lambda (a, b). \{(b, 0), (a, 1)\})$ 
    have g-inj: inj-on g  $\{(a, b) \mid a \ b. a \in A \wedge b \in A \wedge a \ominus b = d\}$ 
    proof (intro inj-onI)
      fix x y assume  $x \in \{(a, b) \mid a \ b. a \in A \wedge b \in A \wedge a \ominus b = d\}$  and
         $y \in \{(a, b) \mid a \ b. a \in A \wedge b \in A \wedge a \ominus b = d\}$  and hg:  $g \ x = g \ y$ 
      then obtain a1 a2 b1 b2 where hx:  $x = (a1, a2)$  and hy:  $y = (b1, b2)$ 
    by blast
    thus  $x = y$  using g-def hg hx hy by (simp add: doubleton-eq-iff)
  qed
  have g-image:  $g \ ‘ \{(a, b) \mid a \ b. a \in A \wedge b \in A \wedge a \ominus b = d\} = f \ d$  using
f-def g-def by auto
  show  $\text{card } (f \ d) = f\text{-diff } d \ A$  using card-image g-inj g-image f-diff-def by
fastforce
  qed
  have  $c \wedge 2 * (\text{card } A) \wedge 2 = c * (\text{card } A) * (c * (\text{card } A))$  using power2-eq-square
    by (metis of-nat-power power-mult-distrib)
  also have  $\dots \leq (\text{card } (\text{popular-diff-set } c \ A)) * (c * (\text{card } A))$ 
    using assms popular-differences-card hA by force
  also have  $\dots \leq (\sum d \in \text{popular-diff-set } c \ A. f\text{-diff } d \ A)$  using sum-mono
popular-diff-set-def
  popular-diff-def by (smt (verit, ccfv-SIG) mem-Collect-eq of-nat-sum of-real-of-nat-eq
    sum-constant)
  also have  $\dots = (\sum d \in \text{popular-diff-set } c \ A. \text{card } (f \ d))$ 
    using card-eq-f-diff sum.cong by auto
  also have  $\dots = \text{sum card } (f \ ‘ (\text{popular-diff-set } c \ A))$ 
    using f-disj sum-card-image[of popular-diff-set c A f] popular-diff-set-def
finite-minusset finite-sumset assms(1) finite-subset by auto
  also have  $\dots = \text{card } (\bigcup d \in \text{popular-diff-set } c \ A. (f \ d))$ 
    using card-Union-disjoint[of f ‘ (popular-diff-set c A)] f-disj2 f-finite by auto
  also have  $\dots \leq \text{card } ?E$  using card-mono f-union-sub H.fin-edges by auto

```

finally have $c^{\wedge} 2 * (\text{card } A)^{\wedge} 2 \leq \text{card } ?E$ **by** *linarith*
then have $c^{\wedge} 2 * (\text{card } A)^{\wedge} 2 \leq \text{card } (H.\text{all-edges-between } ?X ?Y)$
using *H.card-edges-between-set* **by** *auto*
moreover have $H.\text{edge-density } ?X ?Y = \text{card } (H.\text{all-edges-between } ?X ?Y) /$
 $(\text{card } A)^{\wedge} 2$
using *H.edge-density-def power2-eq-square hXA hYA*
by *(smt (verit, best))*
ultimately have $(c^{\wedge} 2 * (\text{card } A)^{\wedge} 2) / (\text{card } A)^{\wedge} 2 \leq H.\text{edge-density } ?X$
 $?Y$ **using** *hA*
divide-le-cancel **by** *(smt (verit, del-insts) H.edge-density-ge0 <c² * real ((card*
 $A)^2) =$
 $c * \text{real } (\text{card } A) * (c * \text{real } (\text{card } A))$ *divide-divide-eq-right zero-le-divide-iff)*
thus *?thesis using hA assms(2)* **by** *auto*
qed
obtain X' **and** Y' **where** $X'_{\text{sub}}: X' \subseteq ?X$ **and** $Y'_{\text{sub}}: Y' \subseteq ?Y$ **and**
 $hX': \text{card } X' \geq (H.\text{edge-density } ?X ?Y)^{\wedge} 2 * (\text{card } ?X) / 16$ **and**
 $hY': \text{card } Y' \geq (H.\text{edge-density } ?X ?Y) * (\text{card } ?Y) / 4$ **and**
 $hwalks: \forall x \in X'. \forall y \in Y'. \text{card } (\{p. H.\text{connecting-walk } x y p \wedge H.\text{walk-length}$
 $p = 3\})$
 $\geq (H.\text{edge-density } ?X ?Y)^{\wedge} 6 * \text{card } ?X * \text{card } ?Y / 2^{\wedge} 13$
using *H.walks-of-length-3-subsets-bipartite <c>0* **by** *auto*
have $((c^{\wedge} 2)^{\wedge} 2) * (\text{card } A) \leq (H.\text{edge-density } ?X ?Y)^{\wedge} 2 * (\text{card } A)$
using *edge-density assms(2) hA power-mono zero-le-power2 mult-le-cancel-right*
by *(smt (verit) of-nat-less-of-nat-power-cancel-iff of-nat-zero-less-power-iff*
 $\text{power2-less-eq-zero-iff power-0-left})$
then have $\text{card } X' \geq (c^{\wedge} 4) * (\text{card } A) / 16$ **using** *hX' divide-le-cancel*
 hXA **by** *fastforce*
have $c^{\wedge} 2 * (\text{card } A) / 4 \leq (H.\text{edge-density } ?X ?Y) * \text{card } ?Y / 4$ **using** *hYA*
 hA *edge-density*
mult-le-cancel-right **by** *simp*
then have $\text{card } Y' \geq c^{\wedge} 2 * (\text{card } A) / 4$ **using** *hY' by linarith*
have $(H.\text{edge-density } ?X ?Y)^{\wedge} 6 * (\text{card } ?X * \text{card } ?Y) / 2^{\wedge} 13 \geq (c^{\wedge} 2)^{\wedge} 6 *$
 $((\text{card } A)^{\wedge} 2) / 2^{\wedge} 13$ **using**
 $hXA hYA \text{ power2-eq-square edge-density divide-le-cancel mult-le-cancel-right } hA$
by *(smt (verit, ccfv-SIG) of-nat-power power2-less-0 power-less-imp-less-base*
 $\text{zero-le-power})$
then have $\text{card-walks: } \forall x \in X'. \forall y \in Y'.$
 $\text{card } (\{p. H.\text{connecting-walk } x y p \wedge H.\text{walk-length } p = 3\}) \geq (c^{\wedge} 12) * ((\text{card}$
 $A)^{\wedge} 2) / 2^{\wedge} 13$
using *hwalks* **by** *fastforce*

let $?B = (\lambda (a, b). a) \text{ ' } X'$
let $?C = (\lambda (a, b). a) \text{ ' } Y'$
have $hBA: ?B \subseteq A$ **and** $hCA: ?C \subseteq A$ **using** $Y'_{\text{sub}} X'_{\text{sub}}$ **by** *auto*
have $\text{inj-on-}X': \text{inj-on } (\lambda (a, b). a) X'$ **using** X'_{sub} **by** *(intro inj-onI) (auto)*
have $\text{inj-on-}Y': \text{inj-on } (\lambda (a, b). a) Y'$ **using** Y'_{sub} **by** *(intro inj-onI) (auto)*
have $hBX': \text{card } ?B = \text{card } X'$ **and** $hCY': \text{card } ?C = \text{card } Y'$
using *card-image inj-on-X' inj-on-Y'* **by** *auto*
then have $\text{card } B: \text{card } ?B \geq (c^{\wedge} 4) * (\text{card } A) / 16$ **and** $\text{card } C: \text{card } ?C \geq c^{\wedge} 2 *$

```

(card A)/4
  using cardX' cardY' by auto
  have card-ineq1:  $\bigwedge x y. x \in ?B \implies y \in ?C \implies \text{card } (\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}) \geq (c \wedge 12) * ((\text{card } A) \wedge 2) / 2 \wedge 13$ 
  proof-
    fix x y assume hx:  $x \in ?B$  and hy:  $y \in ?C$ 
    have hxA:  $x \in A$  and hyA:  $y \in A$  using hx hy hBA hCA by auto
    define f:: 'a  $\times$  'a  $\implies$  ('a  $\times$  nat) list
      where f  $\equiv$  ( $\lambda (z, w). [(x, 0), (z, 1), (w, 0), (y, 1)]$ )
    have f-inj-on: inj-on f  $\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}$  using f-def by (intro inj-onI) (auto)
    have f-image:  $f \text{ ' } \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\} = \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$ 
    proof
      show  $f \text{ ' } \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\} \subseteq \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$ 
      proof
        fix p assume hp:  $p \in f \text{ ' } \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}$ 
        then obtain z w where hz:  $z \in A$  and hw:  $w \in A$  and hzx:  $\text{popular-diff } (z \ominus x) c A$  and hzw:  $\text{popular-diff } (z \ominus w) c A$  and hyw:  $\text{popular-diff } (y \ominus w) c A$  and hp:  $p = [(x, 0), (z, 1), (w, 0), (y, 1)]$  using f-def hp by fast
        then have hcon:  $H.\text{connecting-walk } (x, 0) (y, 1) p$ 
          unfolding H.connecting-walk-def H.is-walk-def
          using hxA hyA H.vert-adj-def H.vert-adj-sym edges1 by simp
        thus  $p \in \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$ 
          using hp H.walk-length-conv by auto
        qed
      next
        show  $\{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\} \subseteq f \text{ ' } \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}$ 
        proof (intro subsetI)
          fix p assume hp:  $p \in \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$ 
          then have len:  $\text{length } p = 4$  using H.walk-length-conv by auto
          have hpsub:  $\text{set } p \subseteq A \times \{0\} \cup A \times \{1\}$  using hp H.connecting-walk-def H.is-walk-def
          by auto

```

```

then have fst-sub: fst ' set  $p \subseteq A$  by auto
have h1A: fst (p!1)  $\in A$  and h2A: fst (p!2)  $\in A$  using fst-sub len by auto
have hpnum:  $p = [p!0, p!1, p!2, p!3]$ 
proof (auto simp add: list-eq-iff-nth-eq len)
  fix k assume  $k < (4 :: nat)$ 
  then have  $k = 0 \vee k = 1 \vee k = 2 \vee k = 3$  by auto
  thus  $p ! k = [p ! 0, p ! Suc\ 0, p ! 2, p ! 3] ! k$  by fastforce
qed
then have set (H.walk-edges p) =  $\{\{p!0, p!1\}, \{p!1, p!2\}, \{p!2, p!3\}\}$ 
using
  comp-sgraph.walk-edges.simps(2) comp-sgraph.walk-edges.simps(3)
  by (metis empty-set list.simps(15))
then have h1:  $\{p!0, p!1\} \in ?E$  and h2:  $\{p!2, p!1\} \in ?E$  and h3:  $\{p!2,$ 
p!3 $\} \in ?E$ 
  using hp H.connecting-walk-def H.is-walk-def len by auto
  have hxp:  $p!0 = (x, 0)$  using hp len hd-conv-nth H.connecting-walk-def
H.is-walk-def
  by fastforce
  have hyp:  $p!3 = (y, 1)$  using hp len last-conv-nth H.connecting-walk-def
H.is-walk-def
  by fastforce
have h1p:  $p!1 = (fst\ (p!1), 1)$ 
proof -
  have  $p!1 \in A \times \{0\} \cup A \times \{1\}$  using hpnum hpsub
  by (metis (no-types, lifting) insertCI list.simps(15) subsetD)
  then have hsplit:  $snd\ (p!1) = 0 \vee snd\ (p!1) = 1$  by auto
  then have  $snd\ (p!1) = 1$ 
  proof (cases  $snd\ (p!1) = 0$ )
    case True
    then have 1:  $\{(x, 0), (fst\ (p!1), 0)\} \in ?E$  using h1 hxp doubleton-eq-iff
by (smt (verit, del-Insts) surjective-pairing)
    have hY:  $(fst\ (p!1), 0) \notin ?Y$  and hX:  $(x, 0) \in ?X$  using hxA by
auto
    then have 2:  $\{(x, 0), (fst\ (p!1), 0)\} \notin ?E$  using H.X-vert-adj-Y
H.vert-adj-def by meson
    then show ?thesis using 1 2 by blast
  next
  case False
  then show ?thesis using hsplit by auto
qed
thus  $(p ! 1) = (fst\ (p ! 1), 1)$ 
by (metis (full-types) split-pairs)
qed
have h2p:  $p!2 = (fst\ (p!2), 0)$ 
proof -
  have  $p!2 \in A \times \{0\} \cup A \times \{1\}$  using hpnum hpsub
  by (metis (no-types, lifting) insertCI list.simps(15) subsetD)
  then have hsplit:  $snd\ (p!2) = 0 \vee snd\ (p!2) = 1$  by auto
  then have  $snd\ (p!2) = 0$ 

```

```

proof(cases snd (p!2) = 1)
  case True
  then have 1: {(fst (p!2), 1), (y, 1)} ∈ ?E using h3 hyp doubleton-eq-iff
    by (smt (verit, del-insts) surjective-pairing)
    have hY: (y, 1) ∉ ?X and hX: (fst (p!2), 1) ∈ ?Y using hyA h2A
by auto
    then have 2: {(fst (p!2), 1), (y, 1)} ∉ ?E using H.Y-vert-adj-X
H.vert-adj-def
    by meson
    then show ?thesis using 1 2 by blast
  next
  case False
  then show ?thesis using hsplit by auto
  qed
  thus (p ! 2) = (fst (p ! 2), 0)
    by (metis (full-types) split-pairs)
  qed
  have hpop1: popular-diff ((fst (p!1)) ⊖ x) c A using edges1 h1 hxp h1p
hxA h1A
    by (smt (z3))
  have hpop2: popular-diff((fst (p!1)) ⊖ (fst (p!2))) c A using edges1 h2
h1p h2p h1A h2A
    by (smt (z3))
  have hpop3: popular-diff (y ⊖ (fst (p!2))) c A using edges1 h3 h2p hyp
hyA h2A
    by (smt (z3))
  thus p ∈ f ' {(z, w) | z w. z ∈ A ∧ w ∈ A ∧ popular-diff (z ⊖ x) c A ∧
popular-diff (z ⊖ w) c A ∧ popular-diff (y ⊖ w) c A} using f-def hpnum
hxp h1p h2p hyp
    h1A h2A hpop1 hpop2 hpop3 by force
  qed
qed
  have hx1: (x, 0) ∈ X' and hy2: (y, 1) ∈ Y' using hx X'sub hy Y'sub by
auto
  have card {(z, w) | z w. z ∈ A ∧ w ∈ A ∧ popular-diff (z ⊖ x) c A ∧
popular-diff (z ⊖ w) c A ∧ popular-diff (y ⊖ w) c A} =
card {p. H.connecting-walk (x, 0) (y, 1) p ∧ H.walk-length p = 3}
using card-image f-inj-on f-image by fastforce
  thus card {(z, w) | z w. z ∈ A ∧ w ∈ A ∧ popular-diff (z ⊖ x) c A ∧
popular-diff (z ⊖ w) c A ∧ popular-diff (y ⊖ w) c A} ≥ c ^ 12 * ((card
A) ^ 2) / 2 ^ 13
using hx1 hy2 card-walks by auto
  qed
  have card-ineq2: ∧ x y z w. x ∈ ?B ⇒ y ∈ ?C ⇒ (z, w) ∈ {(z, w) | z w. z ∈
A ∧ w ∈ A ∧
popular-diff (z ⊖ x) c A ∧ popular-diff (z ⊖ w) c A ∧ popular-diff (y ⊖ w) c
A} ⇒
card {(p, q, r, s, t, u) | p q r s t u. p ∈ A ∧ q ∈ A ∧ r ∈ A ∧ s ∈ A ∧ t ∈ A ∧
u ∈ A ∧

```

$p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w \} \geq c^{\wedge 3} * \text{card } A^{\wedge 3}$
proof (auto)
fix $x\ x2\ y\ y2\ z\ w$ **assume** $(x, x2) \in X'$ **and** $(y, y2) \in Y'$ **and** $z \in A$ **and** $w \in A$ **and**
1: *popular-diff* $(z \ominus x) c A$ **and** 2: *popular-diff* $(z \ominus w) c A$ **and**
3: *popular-diff* $(y \ominus w) c A$
define $f:: 'a \times 'a \times 'a \times 'a \times 'a \times 'a \Rightarrow ('a \times 'a) \times ('a \times 'a) \times ('a \times 'a)$
where
 $f \equiv (\lambda (p, q, r, s, t, u). ((p, q), (r, s), (t, u)))$

have $f\text{-inj}: \text{inj-on } f \{ (p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w \}$ **using** $f\text{-def}$
by (intro inj-onI) (auto)
have $f\text{-image}: f \text{ ` } \{ (p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w \} =$
 $\{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \} \times$
 $\{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \} \times$
 $\{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = y \ominus w \}$ **using** $f\text{-def}$ **by** force
have $\text{card } \{ (p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w \} = \text{card}$
 $(\{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \} \times$
 $\{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \} \times \{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = y \ominus w \})$
using $\text{card-image } f\text{-inj } f\text{-image}$ **by** fastforce
moreover **have** $\text{card } (\{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \} \times$
 $\{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \} \times \{ (p, q). p \in A \wedge q \in A \wedge p \ominus q = y \ominus w \}) =$
 $\text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \} *$
 $\text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \} *$
 $\text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = y \ominus w \}$
using $\text{card-cartesian-product3}$ **by** auto
moreover **have** $c * \text{card } A \leq \text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \}$
using 1 *popular-diff-def f-diff-def* **by** auto
moreover **then** **have** $(c * \text{card } A) * (c * \text{card } A) \leq \text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \}$
 $* \text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \}$
using 2 *popular-diff-def f-diff-def mult-mono assms(2) mult-nonneg-nonneg of-nat-0-le-iff of-nat-mult* **by** fastforce
moreover **then** **have** $(c * \text{card } A) * (c * \text{card } A) * (c * \text{card } A) \leq \text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus x \}$
 $* \text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = z \ominus w \} *$
 $\text{card } \{ (p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \ominus q = y \ominus w \}$
using 3 *popular-diff-def f-diff-def mult-mono assms(2) mult-nonneg-nonneg of-nat-0-le-iff of-nat-mult* **by** fastforce

moreover have $c \wedge 3 * \text{card } A \wedge 3 = (c * \text{card } A) * ((c * \text{card } A) * (c * \text{card } A))$
by (*simp add: power3-eq-cube algebra-simps*)
ultimately show $c \wedge 3 * \text{real } (\text{card } A) \wedge 3 \leq$
 $(\text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A$
 $\wedge u \in A \wedge$
 $p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})$ **by** *auto*
qed
have *card-ineq3*: $\bigwedge x \ y. x \in ?B \implies y \in ?C \implies \text{card } (\bigcup (z, w) \in \{(z, w) \mid z$
 $w. z \in A \wedge w \in A \wedge$
 $\text{popular-diff } (z \oplus x) \ c \ A \wedge \text{popular-diff } (z \oplus w) \ c \ A \wedge \text{popular-diff } (y \oplus w) \ c$
 $A\}$.
 $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\} \geq$
 $c \wedge 15 * ((\text{card } A) \wedge 5) / 2 \wedge 13$
proof-
fix $x \ y$ **assume** $hx: x \in ?B$ **and** $hy: y \in ?C$
have $hxG: x \in G$ **and** $hyG: y \in G$ **using** $hx \ hy \ hBA \ hCA$ *assms(4)* **by** *auto*
let $?f = (\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge$
 $r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u$
 $= y \oplus w\})$
have *h-pairwise-disjnt*: *pairwise* $(\lambda a \ b. \text{disjnt } (?f \ a) \ (?f \ b))$
 $\{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff } (z \oplus x) \ c \ A \wedge \text{popular-diff } (z \oplus$
 $w) \ c \ A \wedge$
 $\text{popular-diff } (y \oplus w) \ c \ A\}$
proof (*intro pairwiseI*)
fix $a \ b$ **assume** $a \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff } (z \oplus x) \ c \ A$
 \wedge
 $\text{popular-diff } (z \oplus w) \ c \ A \wedge \text{popular-diff } (y \oplus w) \ c \ A\}$ $b \in \{(z, w) \mid z \in$
 $A \wedge w \in A \wedge$
 $\text{popular-diff } (z \oplus x) \ c \ A \wedge \text{popular-diff } (z \oplus w) \ c \ A \wedge \text{popular-diff } (y \oplus w)$
 $c \ A\}$ **and**
 $a \neq b$
then obtain $a1 \ a2 \ b1 \ b2$ **where** $ha: a = (a1, a2)$ **and** $hb: b = (b1, b2)$ **and**
 $ha1: a1 \in G$ **and**
 $ha2: a2 \in G$ **and** $hb1: b1 \in G$ **and** $hb2: b2 \in G$ **and** $hne: (a1, a2) \neq (b1,$
 $b2)$
using *assms(4)* **by** *blast*
have $(\forall x. \neg (x \in (?f \ a) \wedge x \in (?f \ b)))$
proof(*intro allI notI*)
fix d **assume** $d \in (?f \ a) \wedge d \in (?f \ b)$
then obtain $p \ q \ r \ s \ t \ u$ **where** $d = (p, q, r, s, t, u)$ **and** $hpq1: p \oplus q =$
 $a1 \oplus x$ **and**
 $htu1: t \oplus u = y \oplus a2$ **and** $hpq2: p \oplus q = b1 \oplus x$ **and** $htu2: t \oplus u = y$
 $\oplus b2$
using $ha \ hb$ **by** *auto*
then have $y \oplus a2 = y \oplus b2$ **using** $htu1 \ htu2$ **by** *auto*
then have $2: a2 = b2$ **using** $ha2 \ hb2 \ hyG$
by (*metis additive-abelian-group.inverse-closed additive-abelian-group-axioms*)

```

      invertible invertible-inverse-inverse invertible-left-cancel)
    have 1: a1 = b1 using hpq1 hpq2 ha1 hb1 hxG by simp
    show False using 1 2 hne by auto
  qed
  thus disjnt (?f a) (?f b) using disjnt-iff[of (?f a) (?f b)] by auto
  qed
  have hfinite-walks:  $\bigwedge B. B \in ((\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})^c \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\}) \implies \text{finite } B$ 
  proof-
    fix B assume B  $\in ((\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})^c \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\})$ 
    then have B  $\subseteq A \times A \times A \times A \times A \times A$  by auto
    thus finite B using assms(1)
    by (auto simp add: finite-subset)
  qed
  have hdisj: disjoint (( $\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})^c \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\}$ ) using h-pairwise-disjnt pairwise-image[of disjnt ( $\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})^c \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\}$ )] by (simp add: pairwise-def)
  have  $\{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\} \subseteq A \times A$  by auto
  then have hwalks-finite: finite  $\{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\}$  using finite-subset assms(1)
  by fastforce
  have f-ineq:  $\forall a \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff}(z \oplus x) \ c \ A \wedge \text{popular-diff}(z \oplus w) \ c \ A \wedge \text{popular-diff}(y \oplus w) \ c \ A\}. c^{\wedge 3} * (\text{card } A)^{\wedge 3} \leq \text{card } (?f a)$  using card-ineq2 hx hy by auto
  have  $c^{\wedge 15} * ((\text{card } A)^{\wedge 5}) / 2^{\wedge 13} = (c^{\wedge 12} * (\text{card } A)^{\wedge 2} / 2^{\wedge 13}) * (c$ 

```

$\wedge^3 * \text{card } A \wedge^3$
by (*simp add: algebra-simps*)
also have ... $\leq \text{card } \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A$
 \wedge
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\} * (c \wedge^3 * (\text{card } A) \wedge^3)$
using *card-ineq1[of x y] hx hy mult-le-cancel-right hA* **by** (*smt (verit, best)*
assms(2)
mult-pos-pos of-nat-0-less-iff of-nat-le-0-iff zero-less-power)
also have ... $= (\sum a \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x)$
 $c A \wedge$
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}. (c \wedge^3 * (\text{card } A) \wedge^3)$
by *auto*
also have ... $\leq (\sum a \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A$
 \wedge
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}. \text{card } (?f a)$
using *sum-mono f-ineq* **by** (*smt (verit, del-insts) of-nat-sum*)
also have ... $= \text{sum card } (?f ' \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z$
 $\ominus x) c A \wedge$
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\})$
using *sum-card-image[of \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus
 $x) c A \wedge$
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\} ?f]$* *h-pairwise-disjnt*
hwalks-finite **by** *auto*
also have ... $= \text{card } (\bigcup (z, w) \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z$
 $\ominus x) c A \wedge$
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}. \{(p, q, r, s, t, u) \mid p q$
 $r s t u.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s$
 $= z \ominus w \wedge$
 $t \ominus u = y \ominus w\}$) **using** *card-Union-disjoint hfinite-walks hdisj* **by** (*metis*
(no-types, lifting))
finally show $c \wedge^{15} * \text{real } (\text{card } A \wedge^5) / 2 \wedge^{13} \leq \text{real } (\text{card } (\bigcup (z, w) \in \{(z,$
 $w) \mid z w.$
 $z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge$
 $\text{popular-diff } (y \ominus w) c A\}. \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r$
 $\in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus$
 $w\}))$ **by** *simp*
qed
have *hdsb*: $\forall d \in \text{differenceset } ?C ?B. \exists y \in ?C. \exists x \in ?B.$
 $(\bigcup (z, w) \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge$
 $\text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}.$
 $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w\})$
 $\subseteq \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
proof
fix *d* **assume** $d \in \text{differenceset } ?C ?B$

then obtain $y \times x$ **where** $hy: y \in ?C$ **and** $hx: x \in ?B$ **and** $hxy: d = y \ominus x$
using *sumset-def minusset-def hBA hCA assms(4) subset-trans*
by *(smt (verit, best) minusset.simps sumset.cases)*
have $hxG: x \in G$ **and** $hyG: y \in G$ **using** $hx\ hy\ hBA\ hCA\ assms(4)$ **by** *auto*
have $(\bigcup (z, w) \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) \text{ c } A \wedge \text{popular-diff } (z \ominus w) \text{ c } A \wedge \text{popular-diff } (y \ominus w) \text{ c } A\}). \{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s$
 $= z \ominus w \wedge t \ominus u = y \ominus w\})$
 $\subseteq \{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge$
 $u \in A \wedge$
 $d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
proof *(rule Union-least)*
fix X **assume** $X \in (\lambda(z, w). \{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s$
 $= z \ominus w \wedge$
 $t \ominus u = y \ominus w\})$ ‘ $\{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) \text{ c } A \wedge \text{popular-diff } (z \ominus w) \text{ c } A \wedge \text{popular-diff } (y \ominus w) \text{ c } A\}$
then obtain $z\ w$ **where** $hX: X = \{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u. p \in A \wedge q$
 $\in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus$
 $w\}$
and $hz: z \in A$ **and** $hw: w \in A$ **by** *auto*
have $hzG: z \in G$ **and** $hwG: w \in G$ **using** $hz\ hw\ assms(4)$ **by** *auto*
have $\{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus$
 $w\} \subseteq$
 $\{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u$
 $\in A \wedge$
 $d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
proof
fix e **assume** $e \in \{(p, q, r, s, t, u) \mid p\ q\ r\ s\ t\ u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w\}$
then obtain $p\ q\ r\ s\ t\ u$ **where** $p \ominus q = z \ominus x$ **and** $r \ominus s = z \ominus w$ **and** t
 $\ominus u = y \ominus w$
and $hp: p \in A$ **and** $hq: q \in A$ **and** $hr: r \in A$ **and** $hs: s \in A$ **and** $ht: t \in$
 A
and $hu: u \in A$ **and** $he: e = (p, q, r, s, t, u)$ **by** *blast*
then have $p \ominus q \ominus r \oplus s \oplus t \ominus u = (z \ominus x) \ominus (z \ominus w) \oplus (y \ominus w)$
by *(smt (z3) additive-abelian-group.inverse-closed additive-abelian-group-axioms*
 $assms(4)$
 $associative\ commutative-monoid.commutative\ commutative-monoid-axioms$
 $composition-closed$
 $invertible\ invertible-inverse-inverse\ monoid.inverse-composition-commute$
 $monoid-axioms\ subsetD)$
also have $\dots = (w \ominus x) \oplus (y \ominus w)$ **using** $hxG\ hyG\ hzG\ hwG$ *associative*
 $commutative$
 $inverse-composition-commute\ invertible-right-inverse2$ **by** *auto*

also have $\dots = y \ominus x$ **using** $hxG hwG hyG$ *associative commutative*
by (*simp add: monoid.invertible-right-inverse2 monoid-axioms*)
finally have $p \ominus q \ominus r \oplus s \oplus t \ominus u = d$ **using** hxy **by** *simp*
thus $e \in \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$ **using** $he \ hp \ hq \ hr \ hs \ ht \ hu$ **by** *auto*
qed
thus $X \subseteq \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
using hX **by** *auto*
qed
thus $\exists y \in (\lambda(a, b). a) \text{ ' } Y'. \exists x \in (\lambda(a, b). a) \text{ ' } X'. (\bigcup (z, w) \in \{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) \ c \ A \wedge \text{popular-diff } (z \ominus w) \ c \ A \wedge \text{popular-diff } (y \ominus w) \ c \ A\}).$
 $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w\} \subseteq \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
using $hx \ hy$ **by** *meson*
qed
have $pos: 0 < c \wedge 15 * \text{real } (card \ A \wedge 5) / 2 \wedge 13$ **using** $hA \langle c > 0 \rangle$ **by** *auto*
have $(5 :: nat) \leq 6$ **by** *auto*
then have $(card \ A \wedge 6 / card \ A \wedge 5) = (card \ A) \wedge (6 - 5)$
using $hA \text{ power-diff}$ **by** (*metis of-nat-eq-0-iff of-nat-power*)
then have $cardApow: (card \ A \wedge 6 / card \ A \wedge 5) = card \ A$ **using** *power-one-right*
by *simp*
moreover have $\forall d \in \text{differenceset } ?C \ ?B. card \ \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge (d = p \ominus q \ominus r \oplus s \oplus t \ominus u)\} \geq c \wedge 15 * (card \ A) \wedge 5 / 2 \wedge 13$
proof
fix d **assume** $d \in \text{differenceset } ?C \ ?B$
then obtain $x \ y$ **where** $hy: y \in ?C$ **and** $hx: x \in ?B$ **and** $hsub:$
 $(\bigcup (z, w) \in \{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) \ c \ A \wedge \text{popular-diff } (z \ominus w) \ c \ A \wedge \text{popular-diff } (y \ominus w) \ c \ A\}).$
 $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \ominus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w\}$
 $\subseteq \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$ **using** $hsub$ **by** *meson*
have $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\} \subseteq A \times A \times A \times A \times A \times A$ **by** *auto*
then have $fin: \text{finite } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \ominus q \ominus r \oplus s \oplus t \ominus u\}$
using *finite-subset assms(1) finite-cartesian-product* **by** *fastforce*

have $c^{15} * (\text{card } A)^5 / 2^{13} \leq \text{card } (\bigcup (z, w) \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-diff } (z \ominus x) c A \wedge \text{popular-diff } (z \ominus w) c A \wedge \text{popular-diff } (y \ominus w) c A\}).$
 $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \ominus x \wedge r \ominus s = z \ominus w \wedge t \ominus u = y \ominus w\}$
using *card-ineq3 hx hy by auto*
also have $\dots \leq \text{card } \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus r \oplus s \oplus t \ominus u\}$
using *hsub card-mono fin by auto*
finally show $c^{15} * (\text{card } A)^5 / 2^{13} \leq \text{card } \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus r \oplus s \oplus t \ominus u\}$ **by** *linarith*
qed
moreover have *pairwise* $(\lambda s t. \text{disjnt } ((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge (d = p \oplus q \ominus r \oplus s \oplus t \ominus u)\}) s) ((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge (d = p \oplus q \ominus r \oplus s \oplus t \ominus u)\}) t))$ *(differenceset ?C ?B)*
unfolding *disjnt-def by (intro pairwiseI) (auto)*
moreover have $\forall d \in \text{differenceset } ?C ?B. ((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge (d = p \oplus q \ominus r \oplus s \oplus t \ominus u)\}) d) \subseteq A \times A \times A \times A \times A \times A$
by *blast*
ultimately have $\text{card } (\text{differenceset } ?C ?B) \leq ((\text{card } A)^6) / (c^{15} * (\text{card } A)^5 / 2^{13})$
using *assms(1) hA finite-cartesian-product card-cartesian-product-6[of A]*
pos card-le-image-div[of A × A × A × A × A × A (λ d. {(p, q, r, s, t, u) | p q r s t u. p ∈ A ∧ q ∈ A ∧ r ∈ A ∧ s ∈ A ∧ t ∈ A ∧ u ∈ A ∧ (d = p ⊕ q ⊖ r ⊕ s ⊕ t ⊖ u)})] differenceset ?C ?B
 $(c^{15} * (\text{card } A)^5 / 2^{13})$ **by** *auto*
also have $\dots = (\text{card } A)^6 / \text{card } A^5 / (c^{15} / 2^{13})$
using *hA assms(3) field-simps by simp*
also have $\dots = (\text{card } A) / (c^{15} / 2^{13})$
using *cardApow by metis*
finally have *final: card (differenceset ?C ?B) ≤ 2¹³ * (1 / c¹⁵) * real (card A)*
by *argo*
have $0 < c^4 * \text{real } (\text{card } A) / 16$ **and** $0 < c^2 * \text{real } (\text{card } A) / 4$ **using** *assms(2) hA by auto*
then have $?B \neq \{\}$ **and** $?C \neq \{\}$ **using** *cardB cardC by auto*
then show *?thesis using hCA hBA cardC cardB final that by auto*
qed

We now show the main theorem, which is a direct application of lemma *obtains-subsets-differenceset-card-bound* and the Ruzsa triangle inequality. (The main theorem corresponds to Corollary 2.19 in Gowers's notes [3].)

theorem *Balog-Szemerédi-Gowers*: fixes $A::\text{a set}$ and $c::\text{real}$
 assumes afin : finite A and $A \neq \{\}$ and $c > 0$ and *additive-energy* $A = 2 * c$
 and ass : $A \subseteq G$
 obtains A' where $A' \subseteq A$ and $\text{card } A' \geq c^2 * \text{card } A / 4$ and
 $\text{card } (\text{differenceset } A' A') \leq 2^{30} * \text{card } A / c^{34}$
proof –
 obtain B and A' where bss : $B \subseteq A$ and bne : $B \neq \{\}$ and bge : $\text{card } B \geq (c^4)$
 $* (\text{card } A) / 16$
 and a2ss : $A' \subseteq A$ and a2ge : $\text{card } A' \geq (c^2) * (\text{card } A) / 4$
 and hcardle : $\text{card } (\text{differenceset } A' B) \leq 2^{13} * \text{card } A / c^{15}$
 using *assms obtains-subsets-differenceset-card-bound* by *metis*
 have Bg0 : $(\text{card } B :: \text{real}) > 0$ using bne afin bss *infinite-super* by *fastforce*
 have $(\text{card } B) * \text{card } (\text{differenceset } A' A') \leq$
 $\text{card } (\text{differenceset } A' B) * \text{card } (\text{differenceset } A' B)$
 using afin a2ss bss *infinite-super* ass *Ruzsa-triangle-ineq1* *card-minusset'* *differenceset-commute*
 $\text{sumset-subset-carrier}$ *subset-trans* *sumset-commute* by $(\text{smt } (\text{verit}, \text{best}))$
 then have $\text{card } B * \text{card } (\text{differenceset } A' A') \leq (\text{card } (\text{differenceset } A' B))^2$
 using bss *bss power2-eq-square* by *metis*
 then have $(\text{card } (\text{differenceset } A' A')) \leq (\text{card } (\text{differenceset } A' B))^2 / \text{card } B$
 using Bg0 *nonzero-mult-div-cancel-left* [of $\text{card } B$ $\text{card}(\text{differenceset } A' A')$]
divide-right-mono by $(\text{smt } (\text{verit}) \text{ of-nat-0 of-nat-mono of-nat-div-le-of-nat})$
 moreover have $(\text{card } (\text{differenceset } A' B))^2 \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2$
 $A))^2$
 using hcardle by *simp*
 ultimately have $(\text{card } (\text{differenceset } A' A')) \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / (\text{card } B)$
 using *pos-le-divide-eq* [OF Bg0] by *simp*
 moreover have $(c^4) * (\text{card } A) / 16 > 0$
 using *assms card-0-eq* by *fastforce*
 moreover have $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / (\text{card } B) =$
 $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 * (1/(\text{card } B))$ by *simp*
 moreover have $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 * (1/(\text{card } B)) \leq$
 $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / ((c^4) * (\text{card } A) / 16)$
 using bge *calculation*(2, 3) *frac-le less-eq-real-def zero-le-power2* by *metis*
 ultimately have $(\text{card } (\text{differenceset } A' A')) \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / ((c^4) * (\text{card } A) / 16)$
 by *linarith*
 then have $(\text{card } (\text{differenceset } A' A')) \leq (2^{30}) * (\text{card } A) / (c^{34})$
 using *card-0-eq* *assms* by $(\text{simp add: power2-eq-square})$
 then show *?thesis* using a2ss a2ge that by *blast*
qed

The following is an analogous version of the Balog–Szemerédi–Gowers Theorem for a sumset instead of a difference set. The proof is similar to that of the original version, again using *obtains-subsets-differenceset-card-bound*, however, instead of the Ruzsa triangle inequality we will use the alternative triangle inequality for sumsets *triangle-ineq-sumsets*.

theorem *Balog-Szemerédi-Gowers-sumset*: fixes $A::\text{a set}$ and $c::\text{real}$

assumes *afin*: finite A and $A \neq \{\}$ and $c > 0$ and additive-energy $A = 2 * c$
and *ass*: $A \subseteq G$
obtains A' where $A' \subseteq A$ and $\text{card } A' \geq c^2 * \text{card } A / 4$ and
 $\text{card } (\text{sumset } A' A') \leq 2^{30} * \text{card } A / c^{34}$

proof –

obtain B and A' where *bss*: $B \subseteq A$ and *bne*: $B \neq \{\}$ and *bge*: $\text{card } B \geq (c^4) * (\text{card } A) / 16$

and *a2ss*: $A' \subseteq A$ and *a2ne*: $A' \neq \{\}$ and *a2ge*: $\text{card } A' \geq (c^2) * (\text{card } A) / 4$

and *hcardle*: $\text{card } (\text{differenceset } A' B) \leq 2^{13} * \text{card } A / c^{15}$

using *assms obtains-subsets-differenceset-card-bound* **by** *metis*

have *finA'*: finite A' and *finB*: finite B **using** *afin a2ss bss* **using** *infinite-super*
by *auto*

have *bg0*: $(\text{card } B :: \text{real}) > 0$ **using** *bne afin bss infinite-super* **by** *fastforce*

have $\text{card } (\text{minusset } B) * \text{card } (\text{sumset } A' A') \leq$

$\text{card } (\text{sumset } (\text{minusset } B) A') * \text{card } (\text{sumset } (\text{minusset } B) A')$

using *finA' finB ass a2ss bss triangle-ineq-sumsets*

finite-minusset minusset-subset-carrier subset-trans **by** *metis*

then have $\text{card } B * \text{card } (\text{sumset } A' A') \leq (\text{card } (\text{differenceset } A' B))^2$

using *card-minusset bss ass power2-eq-square*

by *(metis card-minusset' subset-trans sumset-commute)*

then have $(\text{card } (\text{sumset } A' A')) \leq (\text{card } (\text{differenceset } A' B))^2 / \text{card } B$

using *bg0 nonzero-mult-div-cancel-left[of card B card(sumset A' A')]*

divide-right-mono **by** *(smt (verit) of-nat-0 of-nat-mono of-nat-div-le-of-nat)*

moreover have $(\text{card } (\text{differenceset } A' B))^2 \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2$

using *hcardle* **by** *simp*

ultimately have $(\text{card } (\text{sumset } A' A')) \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / (\text{card } B)$

using *pos-le-divide-eq[OF bg0]* **by** *simp*

moreover have $(c^4) * (\text{card } A) / 16 > 0$

using *assms card-0-eq* **by** *fastforce*

moreover have $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / (\text{card } B) =$

$((2^{13}) * (1/c^{15}) * (\text{card } A))^2 * (1/(\text{card } B))$ **by** *simp*

moreover have $((2^{13}) * (1/c^{15}) * (\text{card } A))^2 * (1/(\text{card } B)) \leq$

$((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / ((c^4) * (\text{card } A) / 16)$ **using** *bge frac-le less-eq-real-def*

zero-le-power2 calculation(2, 3) **by** *metis*

ultimately have $(\text{card } (\text{sumset } A' A')) \leq ((2^{13}) * (1/c^{15}) * (\text{card } A))^2 / ((c^4) * (\text{card } A) / 16)$

by *linarith*

then have $(\text{card } (\text{sumset } A' A')) \leq (2^{30}) * (\text{card } A) / (c^{34})$

using *card-0-eq assms* **by** *(simp add: power2-eq-square)*

then show *?thesis* **using** *a2ss a2ne a2ge* that **by** *blast*

qed

end

end

8 Supplementary results related to intermediate lemmas used in the proof of the Balog–Szemerédi–Gowers Theorem

```

theory Balog-Szemerédi-Gowers-Supplementary
  imports
    Balog-Szemerédi-Gowers-Main-Proof
begin

```

```

context additive-abelian-group

```

```

begin

```

Even though it is not applied anywhere in this development, for the sake of completeness we give the following analogous version of Lemma 2.17 (*popular-differences-card*) but for popular sums instead of popular differences. The proof is identical to that of Lemma 2.17, with the obvious modifications.

```

lemma popular-sums-card:

```

```

  fixes A::'a set and c::real

```

```

  assumes finite A and additive-energy A = 2 * c and A ⊆ G

```

```

  shows card (popular-sum-set c A) ≥ c * card A

```

```

proof(cases card A ≠ 0)

```

```

  assume hA: card A ≠ 0

```

```

  have hc: c ≥ 0 using assms additive-energy-def of-nat-0-le-iff

```

```

  by (smt (verit, best) assms(3) divide-nonneg-nonneg of-nat-0-le-iff)

```

```

  have (2 * c) * (card A)^3 = (∑ d ∈ (sumset A A). (f-sum d A)^2)

```

```

  using assms f-sum-card-quadruple-set-additive-energy by auto

```

```

  also have ... = ((∑ d ∈ (popular-sum-set c A). (f-sum d A)^2))

```

```

  + ((∑ d ∈ ((sumset A A) - (popular-sum-set c A)). (f-sum d A)^2))

```

```

  using popular-sum-set-def assms finite-sumset by (metis (no-types, lifting)

```

```

    add commute mem-Collect-eq subsetI sum.subset-diff)

```

```

  also have ... ≤ (card (popular-sum-set c A)) * (card A)^2

```

```

  + c * card A * ((∑ d ∈ (sumset A A - (popular-sum-set c A)) . (f-sum d A))

```

```

proof –

```

```

  have ∀ d ∈ ((sumset A A) - (popular-sum-set c A)) . (f-sum d A)^2 ≤

```

```

    (c * (card A)) * (f-sum d A)

```

```

proof

```

```

  fix d assume hd1: d ∈ sumset A A - popular-sum-set c A

```

```

  have hnonneg: f-sum d A ≥ 0 by auto

```

```

  have ¬ popular-sum d c A using hd1 popular-sum-set-def by blast

```

```

  from this have f-sum d A ≤ c * card A using popular-sum-def by auto

```

```

  thus real ((f-sum d A)^2) ≤ c * real (card A) * real (f-sum d A)

```

```

  using power2-eq-square hnonneg mult-right-mono of-nat-0 of-nat-le-iff of-nat-mult

```

```

by metis

```

```

qed

```

```

moreover have ∀ d ∈ (sumset A A) . f-sum d A ≤ (card A)^2

```

```

  using f-sum-def finite-sumset assms

```

by (metis f-sum-le-card le-antisym nat-le-linear power2-nat-le-imp-le)
 ultimately have $((\sum d \in ((\text{sumset } A \ A) - \text{popular-sum-set } c \ A) . (f\text{-sum } d \ A)^{\wedge 2})) \leq$
 $((\sum d \in ((\text{sumset } A \ A) - \text{popular-sum-set } c \ A) . (c * \text{card } A) * (f\text{-sum } d \ A)))$
 using *assms finite-sumset sum-distrib-left sum-mono* by *fastforce*
 then have $((\sum d \in ((\text{sumset } A \ A) - \text{popular-sum-set } c \ A) . (f\text{-sum } d \ A)^{\wedge 2}))$
 \leq
 $(c * \text{card } A) * ((\sum d \in ((\text{sumset } A \ A) - \text{popular-sum-set } c \ A) . (f\text{-sum } d \ A)))$
 by (metis (no-types) of-nat-sum sum-distrib-left)
 moreover have $(\sum d \in \text{popular-sum-set } c \ A . (f\text{-sum } d \ A)^{\wedge 2}) \leq$
 $(\sum d \in \text{popular-sum-set } c \ A . (\text{card } A)^{\wedge 2})$ using *f-sum-le-card assms sum-mono*
assms popular-sum-set-def
 by (metis (no-types, lifting) power2-nat-le-eq-le)
 moreover then have $(\sum d \in \text{popular-sum-set } c \ A . (f\text{-sum } d \ A)^{\wedge 2}) \leq$
 $(\text{card } (\text{popular-sum-set } c \ A)) * (\text{card } A)^{\wedge 2}$
 using *sum-distrib-right* by *simp*
 ultimately show *?thesis* by *linarith*
 qed
 also have ... $\leq ((\text{card } (\text{popular-sum-set } c \ A)) * (\text{card } A)^{\wedge 2}) + (c * \text{card } A) *$
 $(\text{card } A)^{\wedge 2}$
 proof -
 have $(\sum d \in (\text{sumset } A \ A - \text{popular-sum-set } c \ A) . (f\text{-sum } d \ A)) \leq$
 $(\sum d \in \text{sumset } A \ A . (f\text{-sum } d \ A))$ using *DiffD1 subsetI assms sum-mono2*
finite-sumset zero-le by *metis*
 then have $(c * \text{card } A) * ((\sum d \in (\text{sumset } A \ A - \text{popular-sum-set } c \ A) . (f\text{-sum}$
 $d \ A)))$
 $\leq (c * \text{card } A) * (\text{card } A)^{\wedge 2}$
 using *f-sum-card hc le0 mult-left-mono of-nat-0 of-nat-mono zero-le-mult-iff*
assms by *metis*
 then show *?thesis* by *linarith*
 qed
 finally have $(2 * c) * (\text{card } A)^{\wedge 3} \leq ((\text{card } (\text{popular-sum-set } c \ A)) * (\text{card } A)^{\wedge 2})$
 $+$
 $(c * \text{card } A) * (\text{card } A)^{\wedge 2}$ by *linarith*
 then have $(\text{card } (\text{popular-sum-set } c \ A)) \geq$
 $((2 * c) * (\text{card } A)^{\wedge 3} - (c * \text{card } A) * (\text{card } A)^{\wedge 2}) / ((\text{card } A)^{\wedge 2})$
 using *hA* by (*simp add: field-simps*)
 moreover have $((2 * c) * (\text{card } A)^{\wedge 3} - (c * \text{card } A) * (\text{card } A)^{\wedge 2}) / ((\text{card } A)^{\wedge 2})$
 $= 2 * c * \text{card } A - c * \text{card } A$
 using *hA* by (*simp add: power2-eq-square power3-eq-cube*)
 ultimately show *?thesis* by *linarith*
 next
 assume $\neg \text{card } A \neq 0$
 thus *?thesis* by *auto*
 qed

The following is an analogous version of lemma *obtains-subsets-differenceset-card-bound* (2.18 in Gowers's notes [3]) but for a sumset instead of a difference set. It is not used anywhere in this development but we provide it for the sake of com-

pleteness. The proof is identical to that of lemma *obtains-subsets-differenceset-card-bound* with *f-diff* changed to *f-sum*, *popular-diff* changed to *popular-sum*, \oplus interchanged with \ominus , and instead of lemma *popular-differences-card* we apply its analogous version for popular sums, that is lemma *popular-sums-card*.

lemma *obtains-subsets-sumset-card-bound*: **fixes** $A::'a$ set **and** $c::real$
assumes *finite* A **and** $c>0$ **and** $A \neq \{\}$ **and** $A \subseteq G$ **and** *additive-energy* $A = 2 * c$
obtains B **and** A' **where** $B \subseteq A$ **and** $B \neq \{\}$ **and** $\text{card } B \geq c^4 * \text{card } A / 16$
and $A' \subseteq A$ **and** $A' \neq \{\}$ **and** $\text{card } A' \geq c^2 * \text{card } A / 4$
and $\text{card } (\text{sumset } A' B) \leq 2^{13} * \text{card } A / c^{15}$

proof–

let $?X = A \times \{0::nat\}$
let $?Y = A \times \{1::nat\}$
let $?E = \text{mk-edge } \{(x, y) \mid x \in ?X \wedge y \in ?Y \wedge (\text{popular-sum } (\text{fst } y \oplus \text{fst } x) c A)\}$

interpret $H: \text{fin-bipartite-graph } ?X \cup ?Y ?E ?X ?Y$

proof (*unfold-locales, auto simp add: partition-on-def assms(3) assms(1) disjoint-def*)

show $\{\} = A \times \{0\} \implies \text{False}$ **using** *assms(3)* **by** *auto*

next

show $\{\} = A \times \{\text{Suc } 0\} \implies \text{False}$ **using** *assms(3)* **by** *auto*

next

show $A \times \{0\} = A \times \{\text{Suc } 0\} \implies \text{False}$ **using** *assms(3)* **by** *fastforce*

next

fix $x \ y$ **assume** $x \in A$ **and** $y \in A$ **and** *popular-sum* $(y \oplus x) c A$

thus $\{(x, 0), (y, \text{Suc } 0)\} \in \text{all-bi-edges } (A \times \{0\}) (A \times \{\text{Suc } 0\})$

using *all-bi-edges-def*[*of* $A \times \{0\}$ $A \times \{\text{Suc } 0\}$]

by (*simp add: in-mk-edge-img*)

qed

have *edges1*: $\forall a \in A. \forall b \in A. (\{(a, 0), (b, 1)\} \in ?E \iff \text{popular-sum } (b \oplus a) c A)$

by (*auto simp add: in-mk-uedge-img-iff*)

have *hXA*: $\text{card } A = \text{card } ?X$ **by** (*simp add: card-cartesian-product*)

have *hYA*: $\text{card } A = \text{card } ?Y$ **by** (*simp add: card-cartesian-product*)

have *hA*: $\text{card } A \neq 0$ **using** *assms card-0-eq* **by** *blast*

have *edge-density*: $H.\text{edge-density } ?X ?Y \geq c^2$

proof–

define $f:: 'a \Rightarrow ('a \times nat)$ *edge set* **where** $f \equiv (\lambda x. \{(a, 0), (b, 1)\} \mid a \ b. a \in A \wedge b \in A \wedge b \oplus a = x)$

have *f-disj*: *pairwise* $(\lambda s \ t. \text{disjnt } (f \ s) (f \ t))$ (*popular-sum-set* $c \ A$)

proof (*intro pairwiseI*)

fix $x \ y$ **assume** *hx*: $x \in \text{popular-sum-set } c \ A$ **and** *hy*: $y \in \text{popular-sum-set } c$

A

and *hxy*: $x \neq y$

show *disjnt* $(f \ x) (f \ y)$

proof–

have $\forall a. \neg (a \in f \ x \wedge a \in f \ y)$

proof (*intro allI notI*)

```

fix  $a$  assume  $a \in f\ x \wedge a \in f\ y$ 
then obtain  $z\ w$  where  $hazw: a = \{(z, 0), (w, 1)\}$  and  $hx: \{(z, 0), (w,$ 
1)\}  $\in f\ x$ 
and  $hy: \{(z, 0), (w, 1)\} \in f\ y$  using  $f\text{-def}$  by  $blast$ 
have  $w \oplus z = x$  using  $f\text{-def}$   $hx$  by ( $simp\ add: doubleton\text{-eq}\text{-iff}$ )
moreover have  $w \oplus z = y$  using  $f\text{-def}$   $hy$  by ( $simp\ add: doubleton\text{-eq}\text{-iff}$ )
ultimately show  $False$  using  $hxy$  by  $auto$ 
qed
thus  $?thesis$  using  $disjnt\text{-iff}$  by  $auto$ 
qed
have  $f\text{-sub}\text{-edges}: \forall d \in popular\text{-sum}\text{-set}\ c\ A. (f\ d) \subseteq ?E$ 
using  $popular\text{-sum}\text{-set}\text{-def}$   $f\text{-def}$   $edges1$  by  $auto$ 
have  $f\text{-union}\text{-sub}: (\bigcup d \in popular\text{-sum}\text{-set}\ c\ A. (f\ d)) \subseteq ?E$  using  $popu-$ 
 $lar\text{-sum}\text{-set}\text{-def}$ 
 $f\text{-def}$   $edges1$  by  $auto$ 
have  $f\text{-disj}2: disjoint\ (f\ ' (popular\text{-sum}\text{-set}\ c\ A))$  using  $f\text{-disj}$ 
 $pairwise\text{-image}$ [ $of\ disjnt\ f\ popular\text{-sum}\text{-set}\ c\ A$ ] by ( $simp\ add: pairwise\text{-def}$ )
have  $f\text{-finite}: \bigwedge B. B \in f\ ' popular\text{-sum}\text{-set}\ c\ A \implies finite\ B$ 
using  $finite\text{-subset}$   $f\text{-sub}\text{-edges}$   $H.fin\text{-edges}$  by  $auto$ 
have  $card\text{-eq}\text{-f}\text{-diff}: \forall d \in popular\text{-sum}\text{-set}\ c\ A. card\ (f\ d) = f\text{-sum}\ d\ A$ 
proof
fix  $d$  assume  $d \in popular\text{-sum}\text{-set}\ c\ A$ 
define  $g: ('a \times 'a) \Rightarrow ('a \times nat)\ edge$  where  $g = (\lambda (a, b). \{(b, 0), (a, 1)\})$ 
have  $g\text{-inj}: inj\text{-on}\ g\ \{(a, b) \mid a\ b. a \in A \wedge b \in A \wedge a \oplus b = d\}$ 
proof ( $intro\ inj\text{-on}I$ )
fix  $x\ y$  assume  $x \in \{(a, b) \mid a\ b. a \in A \wedge b \in A \wedge a \oplus b = d\}$  and
 $y \in \{(a, b) \mid a\ b. a \in A \wedge b \in A \wedge a \oplus b = d\}$  and  $hg: g\ x = g\ y$ 
then obtain  $a1\ a2\ b1\ b2$  where  $hx: x = (a1, a2)$  and  $hy: y = (b1, b2)$ 
by  $blast$ 
thus  $x = y$  using  $g\text{-def}$   $hg$   $hx$   $hy$  by ( $simp\ add: doubleton\text{-eq}\text{-iff}$ )
qed
have  $g\text{-image}: g\ ' \{(a, b) \mid a\ b. a \in A \wedge b \in A \wedge a \oplus b = d\} = f\ d$  using
 $f\text{-def}$   $g\text{-def}$  by  $auto$ 
show  $card\ (f\ d) = f\text{-sum}\ d\ A$  using  $card\text{-image}$   $g\text{-inj}$   $g\text{-image}$   $f\text{-sum}\text{-def}$  by
 $fastforce$ 
qed
have  $c^{\wedge} 2 * (card\ A)^{\wedge} 2 = c * (card\ A) * (c * (card\ A))$  using  $power2\text{-eq}\text{-square}$ 
by ( $metis\ of\text{-nat}\text{-power}\ power\text{-mult}\text{-distrib}$ )
also have  $\dots \leq (card\ (popular\text{-sum}\text{-set}\ c\ A)) * (c * (card\ A))$ 
using  $assms$   $popular\text{-sums}\text{-card}$   $hA$  by  $force$ 
also have  $\dots \leq (\sum d \in popular\text{-sum}\text{-set}\ c\ A. f\text{-sum}\ d\ A)$  using  $sum\text{-mono}$ 
 $popular\text{-sum}\text{-set}\text{-def}$ 
 $popular\text{-sum}\text{-def}$  by ( $smt\ (verit, ccfv\text{-SIG})\ mem\text{-Collect}\text{-eq}\ of\text{-nat}\text{-sum}\ of\text{-real}\ of\text{-nat}\text{-eq}$ 
 $sum\text{-constant}$ )
also have  $\dots = (\sum d \in popular\text{-sum}\text{-set}\ c\ A. card\ (f\ d))$ 
using  $card\text{-eq}\text{-f}\text{-diff}$   $sum.cong$  by  $auto$ 
also have  $\dots = sum\ card\ (f\ ' (popular\text{-sum}\text{-set}\ c\ A))$ 

```

using *f-disj sum-card-image*[of popular-sum-set $c A$ f] *popular-sum-set-def finite-sumset assms(1) finite-subset* **by** *auto*
also have $\dots = \text{card} (\bigcup d \in \text{popular-sum-set } c A. (f d))$
using *card-Union-disjoint*[of $f^{-1} (\text{popular-sum-set } c A)$] *f-disj2 f-finite* **by** *auto*
also have $\dots \leq \text{card } ?E$ **using** *card-mono f-union-sub H.fin-edges* **by** *auto*
finally have $c^2 * (\text{card } A)^2 \leq \text{card } ?E$ **by** *linarith*
then have $c^2 * (\text{card } A)^2 \leq \text{card} (H.\text{all-edges-between } ?X ?Y)$
using *H.card-edges-between-set* **by** *auto*
moreover have $H.\text{edge-density } ?X ?Y = \text{card} (H.\text{all-edges-between } ?X ?Y) / (\text{card } A)^2$
using *H.edge-density-def power2-eq-square hXA hYA*
by *(smt (verit, best))*
ultimately have $(c^2 * (\text{card } A)^2) / (\text{card } A)^2 \leq H.\text{edge-density } ?X ?Y$ **using** *hA*
divide-le-cancel **by** *(smt (verit, del-insts) H.edge-density-ge0 <c^2 * real ((card A)^2) = c * real (card A) * (c * real (card A))> divide-divide-eq-right zero-le-divide-iff)*
thus *?thesis* **using** *hA assms(2)* **by** *auto*
qed
obtain X' and Y' **where** $X'_{\text{sub}}: X' \subseteq ?X$ **and** $Y'_{\text{sub}}: Y' \subseteq ?Y$ **and**
 $hX': \text{card } X' \geq (H.\text{edge-density } ?X ?Y)^2 * (\text{card } ?X) / 16$ **and**
 $hY': \text{card } Y' \geq (H.\text{edge-density } ?X ?Y) * (\text{card } ?Y) / 4$ **and**
 $hwalks: \forall x \in X'. \forall y \in Y'. \text{card} (\{p. H.\text{connecting-walk } x y p \wedge H.\text{walk-length } p = 3\}) \geq$
 $(H.\text{edge-density } ?X ?Y)^6 * \text{card } ?X * \text{card } ?Y / 2^{13}$
using *H.walks-of-length-3-subsets-bipartite <c>0>* **by** *auto*
have $((c^2)^2) * (\text{card } A) \leq (H.\text{edge-density } ?X ?Y)^2 * (\text{card } A)$
using *edge-density assms(2) hA power-mono zero-le-power2 mult-le-cancel-right*
by *(smt (verit) of-nat-less-of-nat-power-cancel-iff of-nat-zero-less-power-iff power2-less-eq-zero-iff power-0-left)*
then have $\text{card } X' \geq (c^4) * (\text{card } A) / 16$ **using** *hX'* *divide-le-cancel hXA* **by** *fastforce*
have $c^2 * (\text{card } A) / 4 \leq (H.\text{edge-density } ?X ?Y) * \text{card } ?Y / 4$ **using** *hYA hA edge-density*
mult-le-cancel-right **by** *simp*
then have $\text{card } Y' \geq c^2 * (\text{card } A) / 4$ **using** *hY'* **by** *linarith*
have $(H.\text{edge-density } ?X ?Y)^6 * (\text{card } ?X * \text{card } ?Y) / 2^{13} \geq (c^2)^6 * ((\text{card } A)^2) / 2^{13}$ **using**
hXA hYA power2-eq-square edge-density divide-le-cancel mult-le-cancel-right hA
by *(smt (verit, ccfv-SIG) of-nat-power power2-less-0 power-less-imp-less-base zero-le-power)*
then have $\text{card-walks}: \forall x \in X'. \forall y \in Y'. \text{card} (\{p. H.\text{connecting-walk } x y p \wedge H.\text{walk-length } p = 3\}) \geq (c^{12}) * ((\text{card } A)^2) / 2^{13}$
using *hwalks* **by** *fastforce*

let $?B = (\lambda (a, b). a) \text{ ' } X'$
let $?C = (\lambda (a, b). a) \text{ ' } Y'$
have $hBA: ?B \subseteq A$ **and** $hCA: ?C \subseteq A$ **using** $Y'_{\text{sub}} X'_{\text{sub}}$ **by** *auto*

have *inj-on-X'*: *inj-on* ($\lambda (a, b). a$) *X'* **using** *X'sub* **by** (*intro inj-onI*) (*auto*)
have *inj-on-Y'*: *inj-on* ($\lambda (a, b). a$) *Y'* **using** *Y'sub* **by** (*intro inj-onI*) (*auto*)
have *hBX'*: *card ?B = card X'* **and** *hCY'*: *card ?C = card Y'*
using *card-image inj-on-X' inj-on-Y'* **by** *auto*
then have *cardB*: *card ?B $\geq (c^4) * (card A)/16$* **and** *cardC*: *card ?C $\geq c^2 * (card A)/4$*
using *cardX' cardY'* **by** *auto*
have *card-ineq1*: $\bigwedge x y. x \in ?B \implies y \in ?C \implies \text{card} (\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}) \geq (c^2) * ((card A)^2) / 2^3$
proof–
fix *x y* **assume** *hx*: $x \in ?B$ **and** *hy*: $y \in ?C$
have *hxA*: $x \in A$ **and** *hyA*: $y \in A$ **using** *hx hy hBA hCA* **by** *auto*
define *f*:: $'a \times 'a \Rightarrow ('a \times nat)$ *list*
where *f* $\equiv (\lambda (z, w). [(x, 0), (z, 1), (w, 0), (y, 1)])$
have *f-inj-on*: *inj-on f* $\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$ **using** *f-def* **by** (*intro inj-onI*) (*auto*)
have *f-image*: $f^{-1} \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\} = \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$
proof
show $f^{-1} \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\} \subseteq \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$
proof
fix *p* **assume** *hp*: $p \in f^{-1} \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$
then obtain *z w* **where** *hz*: $z \in A$ **and** *hw*: $w \in A$ **and** *hzx*: *popular-sum* ($z \oplus x$) *c A* **and** *hzw*: *popular-sum* ($z \oplus w$) *c A* **and** *hyw*: *popular-sum* ($y \oplus w$) *c A* **and** *hp*: $p = [(x, 0), (z, 1), (w, 0), (y, 1)]$ **using** *f-def hp* **by** *fast*
then have *hcon*: *H.connecting-walk* ($x, 0$) ($y, 1$) *p*
unfolding *H.connecting-walk-def H.is-walk-def*
using *hxA hyA H.vert-adj-def H.vert-adj-sym edges1* **by** *simp*
thus $p \in \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$
using *hp H.walk-length-conv* **by** *auto*
qed
next
show $\{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\} \subseteq f^{-1} \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$
proof(*intro subsetI*)
fix *p* **assume** *hp*: $p \in \{p. H.\text{connecting-walk } (x, 0) (y, 1) p \wedge H.\text{walk-length } p = 3\}$

```

p = 3}
  then have len: length p = 4 using H.walk-length-conv by auto
  have hpsub: set p ⊆ A × {0} ∪ A × {1} using hp H.connecting-walk-def
H.is-walk-def
  by auto
  then have fst-sub: fst ' set p ⊆ A by auto
  have h1A: fst (p!1) ∈ A and h2A: fst (p!2) ∈ A using fst-sub len by auto
  have hpnum: p = [p!0, p!1, p!2, p!3]
  proof (auto simp add: list-eq-iff-nth-eq len)
    fix k assume k < (4::nat)
    then have k = 0 ∨ k = 1 ∨ k = 2 ∨ k = 3 by auto
    thus p ! k = [p ! 0, p ! Suc 0, p ! 2, p ! 3] ! k by fastforce
  qed
  then have set (H.walk-edges p) = {{p!0, p!1} , {p!1, p!2}, {p!2, p!3}}
using
  comp-sgraph.walk-edges.simps(2) comp-sgraph.walk-edges.simps(3)
  by (metis empty-set list.simps(15))
  then have h1: {p!0, p!1} ∈ ?E and h2: {p!2, p!1} ∈ ?E and h3: {p!2,
p!3} ∈ ?E
    using hp H.connecting-walk-def H.is-walk-def len by auto
  have hxp: p!0 = (x, 0) using hp len hd-conv-nth H.connecting-walk-def
H.is-walk-def
  by fastforce
  have hyp: p!3 = (y, 1) using hp len last-conv-nth H.connecting-walk-def
H.is-walk-def
  by fastforce
  have h1p: p!1 = (fst (p!1), 1)
  proof -
  have p!1 ∈ A × {0} ∪ A × {1} using hpnum hpsub
    by (metis (no-types, lifting) insertCI list.simps(15) subsetD)
  then have hsplite: snd (p!1) = 0 ∨ snd (p!1) = 1 by auto
  then have snd (p!1) = 1
  proof (cases snd (p!1) = 0)
    case True
  then have 1: {(x, 0), (fst (p!1), 0)} ∈ ?E using h1 hxp doubleton-eq-iff
    by (smt (verit, del-Insts) surjective-pairing)
    have hY: (fst (p!1), 0) ∉ ?Y and hX: (x, 0) ∈ ?X using hxA by
auto
  then have 2: {(x, 0), (fst (p!1), 0)} ∉ ?E using H.X-vert-adj-Y
H.vert-adj-def by meson
  then show ?thesis using 1 2 by blast
  next
  case False
  then show ?thesis using hsplite by auto
  qed
  thus (p ! 1) = (fst (p ! 1), 1)
    by (metis (full-types) split-pairs)
  qed
  have h2p: p!2 = (fst (p!2), 0)

```

proof–
have $p!2 \in A \times \{0\} \cup A \times \{1\}$ **using** $hpnum\ hpsub$
by (*metis* (*no-types*, *lifting*) *insertCI list.simps(15) subsetD*)
then have $hsplit: snd\ (p!2) = 0 \vee snd\ (p!2) = 1$ **by** *auto*
then have $snd\ (p!2) = 0$
proof(*cases snd (p!2) = 1*)
case *True*
then have $1: \{(fst\ (p!2), 1), (y, 1)\} \in ?E$ **using** $h3\ hyp\ doubleton-eq-iff$
by (*smt* (*verit*, *del-insts*) *surjective-pairing*)
have $hY: (y, 1) \notin ?X$ **and** $hX: (fst\ (p!2), 1) \in ?Y$ **using** $hyA\ h2A$
by *auto*
then have $2: \{(fst\ (p!2), 1), (y, 1)\} \notin ?E$ **using** $H.Y-vert-adj-X$
H.vert-adj-def
by *meson*
then show *?thesis* **using** $1\ 2$ **by** *blast*
next
case *False*
then show *?thesis* **using** $hsplit$ **by** *auto*
qed
thus $(p!2) = (fst\ (p!2), 0)$
by (*metis* (*full-types*) *split-pairs*)
qed
have $hpop1: popular-sum\ ((fst\ (p!1)) \oplus x)\ c\ A$ **using** $edges1\ h1\ hxp\ h1p$
hxA\ h1A
by (*smt* ($z3$))
have $hpop2: popular-sum((fst\ (p!1)) \oplus (fst\ (p!2)))\ c\ A$ **using** $edges1\ h2$
h1p\ h2p\ h1A\ h2A
by (*smt* ($z3$))
have $hpop3: popular-sum\ (y \oplus (fst\ (p!2)))\ c\ A$ **using** $edges1\ h3\ h2p\ hyp$
hyA\ h2A
by (*smt* ($z3$))
thus $p \in f^{-1}\ \{(z, w) \mid z, w. z \in A \wedge w \in A \wedge popular-sum\ (z \oplus x)\ c\ A \wedge$
 $popular-sum\ (z \oplus w)\ c\ A \wedge popular-sum\ (y \oplus w)\ c\ A\}$ **using** $f-def\ hpnum$
hxp\ h1p\ h2p\ hyp
 $h1A\ h2A\ hpop1\ hpop2\ hpop3$ **by** *force*
qed
qed
have $hx1: (x, 0) \in X'$ **and** $hy2: (y, 1) \in Y'$ **using** $hx\ X'sub\ hy\ Y'sub$ **by**
auto
have $card\ \{(z, w) \mid z, w. z \in A \wedge w \in A \wedge popular-sum\ (z \oplus x)\ c\ A \wedge$
 $popular-sum\ (z \oplus w)\ c\ A \wedge popular-sum\ (y \oplus w)\ c\ A\} =$
 $card\ \{p. H.connecting-walk\ (x, 0)\ (y, 1)\ p \wedge H.walk-length\ p = 3\}$
using *card-image f-inj-on f-image* **by** *fastforce*
thus $card\ \{(z, w) \mid z, w. z \in A \wedge w \in A \wedge popular-sum\ (z \oplus x)\ c\ A \wedge$
 $popular-sum\ (z \oplus w)\ c\ A \wedge popular-sum\ (y \oplus w)\ c\ A\} \geq c^{12} * ((card$
 $A)^2) / 2^{13}$
using $hx1\ hy2\ card-walks$ **by** *auto*
qed
have $\bigwedge x\ x2\ y\ y2\ z\ w. (x, x2) \in X' \implies (y, y2) \in Y' \implies z \in A \implies w \in A$

$\implies \text{popular-sum } (z \oplus x) \text{ c } A \implies \text{popular-sum } (z \oplus w) \text{ c } A \implies \text{popular-sum } (y \oplus w) \text{ c } A \implies$
 $c \wedge 3 * \text{real } (\text{card } A) \wedge 3 \leq$
 $(\text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge$
 $p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})$
proof –
fix $x \ x2 \ y \ y2 \ z \ w$ **assume** $(x, x2) \in X'$ **and** $(y, y2) \in Y'$ **and** $z \in A$ **and** $w \in A$ **and**
1: **popular-sum** $(z \oplus x) \text{ c } A$ **and** 2: **popular-sum** $(z \oplus w) \text{ c } A$ **and**
3: **popular-sum** $(y \oplus w) \text{ c } A$
define $f :: 'a \times 'a \times 'a \times 'a \times 'a \times 'a \Rightarrow ('a \times 'a) \times ('a \times 'a) \times ('a \times 'a)$
where
 $f \equiv (\lambda (p, q, r, s, t, u). ((p, q), (r, s), (t, u)))$

have $f\text{-inj}$: $\text{inj-on } f \ \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}$ **using**
 $f\text{-def}$
by (intro inj-onI) (auto)
have $f\text{-image}$: $f \ ' \ \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\} =$
 $\{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\} \times$
 $\{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\} \times$
 $\{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = y \oplus w\}$ **using** $f\text{-def}$ **by** force

have 4: $\text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\} =$
 $\text{card } (\{(p, q). p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\} \times$
 $\{(p, q). p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\} \times \{(p, q). p \in A \wedge q \in A \wedge p \oplus$
 $q = y \oplus w\})$
using $\text{card-image } f\text{-inj } f\text{-image}$ **by** fastforce
moreover **have** 5: $\text{card } (\{(p, q). p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\} \times$
 $\{(p, q). p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\} \times \{(p, q). p \in A \wedge q \in A \wedge p \oplus$
 $q = y \oplus w\}) =$
 $\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\} *$
 $\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\} *$
 $\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = y \oplus w\}$
using $\text{card-cartesian-product3}$ **by** auto
have $c * \text{card } A \leq \text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\}$
using 1 $\text{popular-sum-def } f\text{-sum-def}$ **by** auto
then **have** $(c * \text{card } A) * (c * \text{card } A) \leq \text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge$
 $p \oplus q = z \oplus x\} *$
 $\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\}$
using 2 $\text{popular-sum-def } f\text{-sum-def}$ $\text{mult-mono assms(2)}$ $\text{mult-nonneg-nonneg}$
 $\text{of-nat-0-le-iff of-nat-mult}$ **by** fastforce
then **have** 6: $(c * \text{card } A) * (c * \text{card } A) * (c * \text{card } A) \leq \text{card } \{(p, q) \mid p \ q.$
 $p \in A \wedge q \in A \wedge p \oplus q = z \oplus x\} *$

$\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = z \oplus w\} * \\
\text{card } \{(p, q) \mid p \ q. p \in A \wedge q \in A \wedge p \oplus q = y \oplus w\} \\
\text{using } 3 \text{ popular-sum-def f-sum-def mult-mono assms}(2) \text{ mult-nonneg-nonneg} \\
\text{of-nat-0-le-iff} \\
\text{of-nat-mult by fastforce} \\
\text{have } \gamma: c \wedge 3 * \text{card } A \wedge 3 = (c * \text{card } A) * ((c * \text{card } A) * (c * \text{card } A)) \\
\text{by (simp add: power3-eq-cube algebra-simps)} \\
\text{show } c \wedge 3 * \text{real } (\text{card } A) \wedge 3 \leq \\
(\text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \\
\wedge u \in A \wedge \\
p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}) \text{ using } 4 \ 5 \ 6 \ 7 \text{ by auto} \\
\text{qed} \\
\text{then have card-ineq2: } \bigwedge x \ y \ z \ w. x \in ?B \implies y \in ?C \implies (z, w) \in \{(z, w) \mid z \\
w. z \in A \wedge w \in A \wedge \\
\text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus w) \\
c \ A\} \implies \\
\text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge \\
u \in A \wedge \\
p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\} \geq c \wedge 3 * \text{card } A \wedge 3 \\
\text{by auto} \\
\text{have card-ineq3: } \bigwedge x \ y. x \in ?B \implies y \in ?C \implies \text{card } (\bigcup (z, w) \in \{(z, w) \mid z \\
w. z \in A \wedge w \in A \wedge \\
\text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus w) \\
c \ A\}. \\
\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge \\
t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}) \geq \\
c \wedge 15 * ((\text{card } A) \wedge 5) / 2 \wedge 13 \\
\text{proof-} \\
\text{fix } x \ y \text{ assume } hx: x \in ?B \text{ and } hy: y \in ?C \\
\text{have } hxG: x \in G \text{ and } hyG: y \in G \text{ using } hx \ hy \ hBA \ hCA \text{ assms}(4) \text{ by auto} \\
\text{let } ?f = (\lambda (z, w). \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge \\
r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u \\
= y \oplus w\}) \\
\text{have } hpair-disj: \text{pairwise } (\lambda a \ b. \text{disjnt } (?f \ a) \ (?f \ b)) \\
\{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus \\
w) \ c \ A \wedge \\
\text{popular-sum } (y \oplus w) \ c \ A\} \\
\text{proof (intro pairwiseI)} \\
\text{fix } a \ b \text{ assume } a \in \{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \ c \ A \\
\wedge \\
\text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus w) \ c \ A\} \ b \in \{(z, w) \mid z \ w. z \in \\
A \wedge w \in A \wedge \\
\text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus \\
w) \ c \ A\} \text{ and} \\
a \neq b \\
\text{then obtain } a1 \ a2 \ b1 \ b2 \text{ where } ha: a = (a1, a2) \text{ and } hb: b = (b1, b2) \text{ and} \\
ha1: a1 \in G \text{ and} \\
ha2: a2 \in G \text{ and } hb1: b1 \in G \text{ and } hb2: b2 \in G \text{ and } hne: (a1, a2) \neq (b1, \\
b2)$

using *assms(4)* **by** *blast*
have $(\forall x. \neg (x \in (?f a) \wedge x \in (?f b)))$
proof(*intro allI notI*)
fix *d* **assume** $d \in (?f a) \wedge d \in (?f b)$
then obtain $p q r s t u$ **where** $d = (p, q, r, s, t, u)$ **and** $hpq1: p \oplus q = a1 \oplus x$ **and**
 $htu1: t \oplus u = y \oplus a2$ **and** $hpq2: p \oplus q = b1 \oplus x$ **and** $htu2: t \oplus u = y \oplus b2$
using *ha hb* **by** *auto*
then have $y \oplus a2 = y \oplus b2$ **using** *htu1 htu2* **by** *auto*
then have $2: a2 = b2$ **using** *ha2 hb2 hyG* **by** (*metis invertible invertible-left-cancel*)
have $1: a1 = b1$ **using** *hpq1 hpq2 ha1 hb1 hxG* **by** *simp*
show *False* **using** $1\ 2\ hne$ **by** *auto*
qed
thus *disjnt* $(?f a) (?f b)$ **using** *disjnt-iff[of (?f a) (?f b)]* **by** *auto*
qed
have *hfinite-walks*: $\bigwedge B. B \in ((\lambda (z, w). \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}) ' \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}) \implies \text{finite } B$
proof-
fix *B* **assume** $B \in ((\lambda (z, w). \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}) ' \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\})$
then have $B \subseteq A \times A \times A \times A \times A$ **by** *auto*
thus *finite* *B* **using** *assms(1)*
by (*auto simp add: finite-subset*)
qed
have *hdisj*: *disjoint* $((\lambda (z, w). \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}) ' \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\})$ **using** *hpair-disj pairwise-image[of disjnt* $(\lambda (z, w). \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})$
 $\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}]$ **by** (*simp add: pairwise-def*)
have $\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge \text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$

$\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A \} \subseteq A \times A$ **by auto**
then have *hwalks-finite*: $\text{finite } \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c } A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}$ **using** *finite-subset*
assms(1)
by fastforce
have *f-ineq*: $\forall a \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c } A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}. c^{\wedge 3} * (\text{card } A)^{\wedge 3}$
 \leq
 $\text{card } (?f a)$ **using** *card-ineq2 hx hy by auto*
have $c^{\wedge 15} * ((\text{card } A)^{\wedge 5}) / 2^{\wedge 13} = (c^{\wedge 12} * (\text{card } A)^{\wedge 2} / 2^{\wedge 13}) * (c^{\wedge 3} * \text{card } A^{\wedge 3})$
by (*simp add: algebra-simps*)
also have $\dots \leq \text{card } \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c}$
 $A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\} * (c^{\wedge 3} * (\text{card } A)^{\wedge 3})$
using *card-ineq1[of x y] hx hy mult-le-cancel-right hA by (smt (verit, best)*
assms(2)
mult-pos-pos of-nat-0-less-iff of-nat-le-0-iff zero-less-power)
also have $\dots = (\sum a \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c}$
 $A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}. (c^{\wedge 3} * (\text{card } A)^{\wedge 3}))$ **by auto**
also have $\dots \leq (\sum a \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c}$
 $A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}. \text{card } (?f a))$
using *sum-mono f-ineq by (smt (verit, del-insts) of-nat-sum)*
also have $\dots = \text{sum card } (?f ' \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c}$
 $A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\})$
using *sum-card-image[of \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c } A \wedge
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\} ?f] hpair-disj$*
hwalks-finite by auto
also have $\dots = \text{card } (\bigcup (z, w) \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c } A \wedge$
 $\text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}. \{(p, q, r, s, t, u) \mid p \text{ q}$
 $r \text{ s } t \text{ u}.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s$
 $= z \oplus w \wedge$
 $t \oplus u = y \oplus w\})$ **using** *card-Union-disjoint hdisj hfinite-walks by (metis*
(no-types, lifting))
finally show $c^{\wedge 15} * \text{real } (\text{card } A^{\wedge 5}) / 2^{\wedge 13} \leq \text{real } (\text{card } (\bigcup (z, w) \in \{(z, w) \mid z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \text{ c } A \wedge \text{popular-sum } (z \oplus w) \text{ c } A \wedge \text{popular-sum } (y \oplus w) \text{ c } A\}. \{(p, q, r, s, t, u) \mid p \text{ q } r \text{ s } t \text{ u}. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus$

$w\})\})$ by *simp*
qed
have $pos: 0 < c \wedge 15 * real (card A \wedge 5) / 2 \wedge 13$ **using** $hA \langle c > 0 \rangle$ **by** *auto*
have $(5:: nat) \leq 6$ **by** *auto*
then have $(card A \wedge 6 / card A \wedge 5) = (card A) \wedge (6 - 5)$
using hA *power-diff* **by** (*metis of-nat-eq-0-iff of-nat-power*)
then have $cardApow: (card A \wedge 6 / card A \wedge 5) = card A$ **using** *power-one-right*
by *simp*
have $hdsb: \forall d \in \text{sumset } ?C ?B. \exists y \in ?C. \exists x \in ?B.$
 $(\bigcup (z, w) \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge$
 $\text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}.$
 $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\})$
 $\subseteq \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
proof
fix d **assume** $d \in \text{sumset } ?C ?B$
then obtain $y x$ **where** $hy: y \in ?C$ **and** $hx: x \in ?B$ **and** $hxy: d = y \oplus x$
using *sumset-def minusset-def hBA hCA assms(4) subset-trans*
by (*smt (verit, best) minusset.simps sumset.cases*)
have $hxG: x \in G$ **and** $hyG: y \in G$ **using** $hx hy hBA hCA assms(4)$ **by** *auto*
have $(\bigcup (z, w) \in \{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge$
 $\text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}. \{(p, q, r, s, t, u) \mid p q$
 $r s t u.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s$
 $= z \oplus w \wedge t \oplus u = y \oplus w\})$
 $\subseteq \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge$
 $u \in A \wedge$
 $d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
proof (*rule Union-least*)
fix X **assume** $X \in (\lambda(z, w). \{(p, q, r, s, t, u) \mid p q r s t u.$
 $p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s$
 $= z \oplus w \wedge$
 $t \oplus u = y \oplus w\})$ ‘ $\{(z, w) \mid z w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A$
 \wedge
 $\text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$
then obtain $z w$ **where** $hX: X = \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q$
 $\in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus$
 $w\}$
and $hz: z \in A$ **and** $hw: w \in A$ **by** *auto*
have $hzG: z \in G$ **and** $hwG: w \in G$ **using** $hz hw assms(4)$ **by** *auto*
have $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus$
 $w\} \subseteq$
 $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u$
 $\in A \wedge$
 $d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
proof

fix e **assume** $e \in \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}$
then obtain $p \ q \ r \ s \ t \ u$ **where** $p \oplus q = z \oplus x$ **and** $r \oplus s = z \oplus w$ **and** $t \oplus u = y \oplus w$
and $hp: p \in A$ **and** $hq: q \in A$ **and** $hr: r \in A$ **and** $hs: s \in A$ **and** $ht: t \in A$
and $hu: u \in A$ **and** $he: e = (p, q, r, s, t, u)$ **by** *blast*
then have $p \oplus q \ominus (r \oplus s) \oplus t \oplus u = z \oplus x \ominus (z \oplus w) \oplus y \oplus w$
by (*smt (verit, ccfv-threshold) assms(4) associative composition-closed hwG hxG hyG hzG inverse-closed subset-eq*)
also have $\dots = y \oplus x$ **using** *hxG hyG hzG hwG*
by (*smt (verit) associative commutative composition-closed inverse-closed invertible invertible-right-inverse2*)
finally have $d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u$ **using** *hxy* **by** *simp*
thus $e \in \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$ **using** $he \ hp \ hq \ hr \ hs \ ht \ hu$ **by** *auto*
qed
thus $X \subseteq \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
using *hX* **by** *auto*
qed
thus $\exists y \in (\lambda(a, b). a) \text{ ' } Y'$. $\exists x \in (\lambda(a, b). a) \text{ ' } X'$. $(\bigcup (z, w) \in \{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus w) \ c \ A\}).$
 $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\} \subseteq \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
using *hx hy* **by** *meson*
qed
moreover have $\forall d \in \text{sumset } ?C \ ?B. \text{card } \{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\} \geq c \wedge 15 * (\text{card } A) \wedge 5 / 2 \wedge 13$
proof
fix d **assume** $d \in \text{sumset } ((\lambda(a, b). a) \text{ ' } Y') ((\lambda(a, b). a) \text{ ' } X')$
then obtain $x \ y$ **where** $hy: y \in ?C$ **and** $hx: x \in ?B$ **and** $hsub: (\bigcup (z, w) \in \{(z, w) \mid z \ w. z \in A \wedge w \in A \wedge \text{popular-sum } (z \oplus x) \ c \ A \wedge \text{popular-sum } (z \oplus w) \ c \ A \wedge \text{popular-sum } (y \oplus w) \ c \ A\}).$
 $\{(p, q, r, s, t, u) \mid p \ q \ r \ s \ t \ u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$

$t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w$)
 $\subseteq \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$ **using** *hdsb* **by**
meson
have $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\} \subseteq A \times A \times A \times A$
 $\times A \times A$ **by** *auto*
then have *fin*: *finite* $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
using *finite-subset* *assms(1)* *finite-cartesian-product* **by** *fastforce*
have $c^{15} * (\text{card } A)^5 / 2^{13} \leq \text{card } (\bigcup (z, w) \in \{(z, w) \mid z w. z \in A$
 $\wedge w \in A \wedge \text{popular-sum } (z \oplus x) c A \wedge$
 $\text{popular-sum } (z \oplus w) c A \wedge \text{popular-sum } (y \oplus w) c A\}$.
 $\{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge s \in A \wedge$
 $t \in A \wedge u \in A \wedge p \oplus q = z \oplus x \wedge r \oplus s = z \oplus w \wedge t \oplus u = y \oplus w\}$)
using *card-ineq3* *hx hy* **by** *auto*
also have $\dots \leq \text{card } \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$
using *hsub* *card-mono* *fin* **by** *auto*
finally show $c^{15} * (\text{card } A)^5 / 2^{13} \leq \text{card } \{(p, q, r, s, t, u) \mid p q r s$
 $t u. p \in A \wedge q \in A \wedge r \in A \wedge$
 $s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}$ **by** *linarith*
qed
moreover have *pairwise* $(\lambda s t. \text{disjnt } ((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in$
 $A \wedge q \in A \wedge$
 $r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}) s)$
 $((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in A \wedge q \in A \wedge$
 $r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}) t))$
(sumset ?C ?B)
unfolding *disjnt-def* **by** *(intro pairwiseI)* *(auto)*
moreover have $\forall d \in \text{sumset } ?C ?B. ((\lambda d. \{(p, q, r, s, t, u) \mid p q r s t u. p \in$
 $A \wedge q \in A \wedge$
 $r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge (d = p \oplus q \oplus r \oplus s \oplus t \oplus u\})) d) \subseteq$
 $A \times A \times A \times A \times A \times A \times A$
by *blast*
ultimately have $\text{card } (\text{sumset } ?C ?B) \leq ((\text{card } A)^6) / (c^{15} * (\text{card } A)^5$
 $/ 2^{13})$
using *assms(1)* *hA* *finite-cartesian-product* *card-cartesian-product-6* *[of A]*
 $\text{pos } \text{card-le-image-div}[\text{of } A \times A \times A \times A \times A \times A \times A (\lambda d. \{(p, q, r, s, t, u) \mid p q$
 $r s t u. p \in A \wedge q \in A \wedge$
 $r \in A \wedge s \in A \wedge t \in A \wedge u \in A \wedge d = p \oplus q \ominus (r \oplus s) \oplus t \oplus u\}) \text{sumset } ?C$
 $?B$
 $(c^{15} * (\text{card } A)^5 / 2^{13})]$ **by** *auto*
also have $\dots = (\text{card } A^6 / \text{card } A^5) / (c^{15} / 2^{13})$
using *hA* *assms(3)* *field-simps* **by** *simp*
also have $\dots = (\text{card } A) / (c^{15} / 2^{13})$
using *cardApow* **by** *metis*
finally have *final*: $\text{card } (\text{sumset } ?C ?B) \leq 2^{13} * (1 / c^{15}) * \text{real } (\text{card } A)$
by *argo*

```

have  $0 < c^4 * \text{real}(\text{card } A) / 16$  and  $0 < c^2 * \text{real}(\text{card } A) / 4$  using
  assms(2) hA by auto
then have  $?B \neq \{\}$  and  $?C \neq \{\}$  using cardB cardC by auto
then show ?thesis using hCA hBA cardC cardB final that by auto
qed

end
end

```

References

- [1] C. Edmonds. Undirected graph theory. *Archive of Formal Proofs*, September 2022. https://isa-afp.org/entries/Undirected_Graph_Theory.html, Formal proof development.
- [2] W. T. Gowers. A new proof of Szemerédi's theorem. *Geometric & Functional Analysis GAFA*, 11(3):465–588, 2001.
- [3] W. T. Gowers. Introduction to additive combinatorics, 2022. Lecture notes for Part III of the Mathematical Tripos taught at the University of Cambridge, available at <https://drive.google.com/file/d/1ut0mUqSyPMweoxoDTfhXverEONyFgcuO/view>.
- [4] Y. Zhao. Graph theory and additive combinatorics. Online at <https://yufeizhao.com/gtacbook/>, 2022. book draft.