

# Szemerédi's Regularity Lemma

Chelsea Edmonds, Angeliki Koutsoukou-Argyraki and Lawrence C. Paulson  
Computer Laboratory, University of Cambridge CB3 0FD  
{cle47,ak2110,lp15}@cam.ac.uk

February 6, 2026

## Abstract

Szemerédi's regularity lemma [2] is a key result in the study of large graphs. It asserts the existence an upper bound on the number of parts the vertices of a graph need to be partitioned into such that the edges between the parts are random in a certain sense. This bound depends only on the desired precision and not on the graph itself, in the spirit of Ramsey's theorem. The formalisation follows online course notes by Tim Gowers<sup>1</sup> and Yufei Zhao<sup>2</sup>. Similar material is found in many textbooks [1].

## Contents

<b>1 Szemerédi's Regularity Lemma</b>	<b>3</b>
1.1 Partitions	3
1.1.1 Partitions indexed by integers	3
1.1.2 Tools to combine the refinements of the partition $P_i$ for each $i$	4
1.2 Edges	4
1.3 Edge Density and Regular Pairs	5
1.4 Energy of a Graph	8
1.5 Partitioning and Energy	10
1.6 Energy boost for partitions	16
1.7 The Regularity Proof Itself	23

## Acknowledgements

The authors were supported by the ERC Advanced Grant ALEXANDRIA (Project 742178) funded by the European Research Council.

<sup>1</sup><https://www.dpmms.cam.ac.uk/~par31/notes/tic.pdf>

<sup>2</sup><https://yufeizhao.com/gtacbook/> and <https://yufeizhao.com/gtac/gtac.pdf> are drafts of a textbook in preparation.

## References

- [1] R. Diestel. *Graph Theory*. Springer, 2017.
- [2] E. Szemerédi. Regular partitions of graphs. Technical Report STAN-CS-75-489, Stanford University Computer Science Department, Apr. 1975.

# 1 Szemerédi’s Regularity Lemma

**theory** *Szemeredi*  
**imports** *HOL–Library.Disjoint-Sets Girth-Chromatic.Ugraphs HOL–Analysis.Convex*  
**begin**

We formalise Szemerédi’s Regularity Lemma, which is a major result in the study of large graphs (extremal graph theory). We follow Yufei Zhao’s notes “Graph Theory and Additive Combinatorics” (MIT), latest version here: <https://yufeizhao.com/gtacbook/> and W.T. Gowers’s notes “Topics in Combinatorics” (University of Cambridge, Lent 2004, Chapter 3) <https://www.dpmms.cam.ac.uk/~par31/notes/tic.pdf>. We also used an earlier version of Zhao’s book: <https://yufeizhao.com/gtac/gtac.pdf>.

## 1.1 Partitions

### 1.1.1 Partitions indexed by integers

**definition** *finite-graph-partition* :: [*uvert set, uvert set set, nat*]  $\Rightarrow$  *bool*  
**where** *finite-graph-partition*  $V P n \equiv$  *partition-on*  $V P \wedge$  *finite*  $P \wedge$  *card*  $P = n$

**lemma** *finite-graph-partition-0* [*iff*]:  
*finite-graph-partition*  $V P 0 \iff V = \{\}$   $\wedge$   $P = \{\}$   
**by** (*auto simp: finite-graph-partition-def partition-on-def*)

**lemma** *finite-graph-partition-empty* [*iff*]:  
*finite-graph-partition*  $\{\} P n \iff P = \{\}$   $\wedge$   $n = 0$   
**by** (*auto simp: finite-graph-partition-def partition-on-def*)

**lemma** *finite-graph-partition-equals*:  
*finite-graph-partition*  $V P n \implies (\bigcup P) = V$   
**by** (*meson finite-graph-partition-def partition-on-def*)

**lemma** *finite-graph-partition-subset*:  
[[*finite-graph-partition*  $V P n$ ;  $X \in P$ ]]  $\implies X \subseteq V$   
**using** *finite-graph-partition-equals* **by** *blast*

**lemma** *trivial-graph-partition-exists*:  
**assumes**  $V \neq \{\}$   
**shows** *finite-graph-partition*  $V \{V\}$  (*Suc 0*)  
**by** (*simp add: assms finite-graph-partition-def partition-on-space*)

**lemma** *finite-graph-partition-finite*:  
**assumes** *finite-graph-partition*  $V P k$  *finite*  $V X \in P$   
**shows** *finite*  $X$   
**by** (*meson assms finite-graph-partition-subset infinite-super*)

**lemma** *finite-graph-partition-gt0*:

**assumes** *finite-graph-partition*  $V P k$  *finite*  $V X \in P$   
**shows**  $\text{card } X > 0$   
**by** (*metis assms card-0-eq finite-graph-partition-def finite-graph-partition-finite-gr-zeroI partition-on-def*)

**lemma** *card-finite-graph-partition*:  
**assumes** *finite-graph-partition*  $V P k$  *finite*  $V$   
**shows**  $(\sum_{X \in P} \text{card } X) = \text{card } V$   
**by** (*metis assms finite-graph-partition-def finite-graph-partition-finite product-partition*)

### 1.1.2 Tools to combine the refinements of the partition $P$ $i$ for each $i$

These are needed to retain the “intuitive” idea of partitions as indexed by integers.

## 1.2 Edges

All edges between two sets of vertices,  $X$  and  $Y$ , in a graph,  $G$

**definition** *all-edges-between*  $:: \text{nat set} \Rightarrow \text{nat set} \Rightarrow \text{nat set} \times \text{nat set} \text{ set} \Rightarrow (\text{nat} \times \text{nat}) \text{ set}$   
**where** *all-edges-between*  $X Y G \equiv \{(x,y). x \in X \wedge y \in Y \wedge \{x,y\} \in \text{uedges } G\}$

**lemma** *all-edges-between-subset*: *all-edges-between*  $X Y G \subseteq X \times Y$   
**by** (*auto simp: all-edges-between-def*)

**lemma** *max-all-edges-between*:  
**assumes** *finite*  $X$  *finite*  $Y$   
**shows**  $\text{card } (\text{all-edges-between } X Y G) \leq \text{card } X * \text{card } Y$   
**by** (*metis assms card-mono finite-SigmaI all-edges-between-subset card-cartesian-product*)

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

**lemma** *all-edges-between-disjnt1*:  
**assumes** *disjnt*  $X Y$   
**shows** *disjnt* (*all-edges-between*  $X Z G$ ) (*all-edges-between*  $Y Z G$ )  
**using** *assms* **by** (*auto simp: all-edges-between-def disjnt-iff*)

**lemma** *all-edges-between-disjnt2*:  
**assumes** *disjnt*  $Y Z$   
**shows** *disjnt* (*all-edges-between*  $X Y G$ ) (*all-edges-between*  $X Z G$ )  
**using** *assms* **by** (*auto simp: all-edges-between-def disjnt-iff*)

**lemma** *all-edges-between-Un1*:  
*all-edges-between*  $(X \cup Y) Z G = \text{all-edges-between } X Z G \cup \text{all-edges-between } Y Z G$

**by** (*auto simp: all-edges-between-def*)

**lemma** *all-edges-between-Un2*:

*all-edges-between X (Y ∪ Z) G = all-edges-between X Y G ∪ all-edges-between X Z G*

**by** (*auto simp: all-edges-between-def*)

**lemma** *finite-all-edges-between*:

**assumes** *finite X finite Y*

**shows** *finite (all-edges-between X Y G)*

**by** (*meson all-edges-between-subset assms finite-cartesian-product finite-subset*)

### 1.3 Edge Density and Regular Pairs

The edge density between two sets of vertices,  $X$  and  $Y$ , in  $G$ . Authors disagree on whether the sets are assumed to be disjoint!. Quite a few authors assume disjointness, e.g. Malliaris and Shelah <https://www.jstor.org/stable/23813167>.

**definition** *edge-density X Y G*  $\equiv$  *card(all-edges-between X Y G) / (card X \* card Y)*

**lemma** *edge-density-ge0*: *edge-density X Y G*  $\geq 0$

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

**lemma** *edge-density-le1*: *edge-density K Y G*  $\leq 1$

**proof** (*cases finite K ∧ finite Y*)

**case** *True*

**then show** *?thesis*

**using** *of-nat-mono [OF max-all-edges-between, of K Y]*

**by** (*fastforce simp add: edge-density-def divide-simps*)

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

**lemma** *all-edges-between-swap*:

*all-edges-between X Y G = (λ(x,y). (y,x)) ‘ (all-edges-between Y X G)*

**unfolding** *all-edges-between-def*

**by** (*auto simp add: insert-commute image-iff split: prod.split*)

**lemma** *card-all-edges-between-commute*:

*card (all-edges-between X Y G) = card (all-edges-between Y X G)*

**proof** –

**have** *inj-on (λ(x, y). (y, x)) A for A :: (nat\*nat)set*

**by** (*auto simp: inj-on-def*)

**then show** *?thesis*

**by** (*simp add: all-edges-between-swap [of X Y] card-image*)

**qed**

**lemma** *edge-density-commute*: *edge-density X Y G = edge-density Y X G*

**by** (*simp add: edge-density-def card-all-edges-between-commute mult.commute*)

$\epsilon$ -regular pairs, for two sets of vertices. Again, authors disagree on whether the sets need to be disjoint, though it seems that overlapping sets cause double-counting. Authors also disagree about whether or not to use the strict subset relation here. The proofs below are easier if it is strict but later proofs require the non-strict version. The two definitions can be proved to be equivalent under fairly mild conditions, but even those conditions turn out to be onerous.

**definition** *regular-pair*::  $real \Rightarrow uvert\ set \Rightarrow uvert\ set \Rightarrow ugraph \Rightarrow bool$   
 ( $\langle \text{--regular'-pair} \rangle$  [999]1000)  
**where**  $\epsilon$ -*regular-pair*  $X\ Y\ G \equiv$   
 $\forall A\ B. A \subseteq X \wedge B \subseteq Y \wedge (card\ A \geq \epsilon * card\ X) \wedge (card\ B \geq \epsilon * card\ Y) \longrightarrow$   
 $|edge-density\ A\ B\ G - edge-density\ X\ Y\ G| \leq \epsilon$  **for**  $\epsilon::real$

**lemma** *regular-pair-commute*:  $\epsilon$ -*regular-pair*  $X\ Y\ G \longleftrightarrow \epsilon$ -*regular-pair*  $Y\ X\ G$   
**by** (*metis edge-density-commute regular-pair-def*)

**lemma** *edge-density-Un*:

**assumes** *disjnt*  $X1\ X2$  *finite*  $X1$  *finite*  $X2$   
**shows**  $edge-density\ (X1 \cup X2)\ Y\ G = (edge-density\ X1\ Y\ G * card\ X1 +$   
 $edge-density\ X2\ Y\ G * card\ X2) / (card\ X1 + card\ X2)$   
**proof** (*cases finite Y*)  
**case** *True*  
**with** *assms show ?thesis*  
**by** (*simp add: edge-density-def all-edges-between-disjnt1 all-edges-between-Un1*  
*finite-all-edges-between card-Un-disjnt card-ge-0-finite divide-simps*)  
**qed** (*simp add: edge-density-def*)

**lemma** *edge-density-partition*:

**assumes** *finite-graph-partition*  $U\ P\ n$   
**shows**  $edge-density\ U\ W\ G = (\sum X \in P. edge-density\ X\ W\ G * card\ X) / card\ U$   
**proof** (*cases finite U*)  
**case** *True*  
**have** *finite P*  
**using** *assms finite-graph-partition-def* **by** *blast*  
**then show** *?thesis*  
**using** *True assms*  
**proof** (*induction P arbitrary: n U*)  
**case** *empty*  
**then show** *?case*  
**by** (*simp add: edge-density-def finite-graph-partition-def partition-on-def*)  
**next**  
**case** (*insert X P*)  
**then have**  $n > 0$   
**by** (*metis finite-graph-partition-0 gr-zeroI insert-not-empty*)  
**with** *insert.premis insert.hyps*  
**have**  $UX$ : *finite-graph-partition*  $(U-X)\ P\ (n-1)$   
**by** (*auto simp: finite-graph-partition-def partition-on-def disjnt-iff pairwise-insert*)  
**then have**  $finU$ : *finite*  $(\bigcup P)$

```

    by (simp add: finite-graph-partition-equals insert)
  then have sumXP: card U = card X + card (⋃ P)
  by (metis UX card-finite-graph-partition finite-graph-partition-equals insert.hyps
insert.prem1 sum.insert)
  have FUX: finite (U - X)
  by (simp add: insert.prem1)
  have XUP: X ∪ (⋃ P) = U
  using finite-graph-partition-equals insert.prem1(2) by auto
  then have edge-density U W G = edge-density (X ∪ ⋃ P) W G
  by auto
  also have ... = (edge-density X W G * card X + edge-density (⋃ P) W G *
card (⋃ P))
    / (card X + card (⋃ P))
  proof (rule edge-density-Un)
  show disjnt X (⋃ P)
  using UX disjnt-iff finite-graph-partition-equals by auto
  show finite X
  using XUP ⟨finite U⟩ by blast
  qed (use finU in auto)
  also have ... = (edge-density X W G * card X + edge-density (U - X) W G *
card (⋃ P))
    / card U
  using UX card-finite-graph-partition finite-graph-partition-equals insert.prem1(1)
insert.prem1(2) sumXP by auto
  also have ... = (∑ Y ∈ insert X P. edge-density Y W G * card Y) / card U
  using UX insert.prem1 insert.hyps
  apply (simp add: insert.IH [OF FUX UX] divide-simps algebra-simps fi-
nite-graph-partition-equals)
  by (metis (no-types, lifting) Diff-eq-empty-iff finite-graph-partition-empty
sum.empty)
  finally show ?case .
  qed
qed (simp add: edge-density-def)

```

Let  $P, Q$  be partitions of a set of vertices  $V$ . Then  $P$  refines  $Q$  if for all  $A \in P$  there is  $B \in Q$  such that  $A \subseteq B$ .

For the sake of generality, and following Zhao's Online Lecture <https://www.youtube.com/watch?v=vcsxCFSLyP8&t=16s> we do not impose disjointness: we do not include  $i \neq j$  below.

**definition** *irregular-set*::  $[real, ugraph, uvert set set] \Rightarrow (uvert set \times uvert set) set$   
 ( $\langle \text{--irregular}^l\text{-set} \rangle [999]1000$ )  
 where  $\varepsilon\text{-irregular-set} \equiv \lambda G P. \{(R,S) \mid R S. R \in P \wedge S \in P \wedge \neg \varepsilon\text{-regular-pair } R S G\}$   
 for  $\varepsilon::real$

A regular partition may contain a few irregular pairs as long as their total size is bounded as follows.

**definition** *regular-partition*::  $[real, ugraph, uvert set set] \Rightarrow bool$

( $\langle \text{--regular}'\text{-partition} \rangle$  [999]1000)  
**where**  
 $\varepsilon\text{-regular-partition} \equiv \lambda G P .$   
 $\text{partition-on } (\text{uverts } G) P \wedge$   
 $(\sum (R,S) \in \text{irregular-set } \varepsilon G P . \text{card } R * \text{card } S) \leq \varepsilon * (\text{card } (\text{uverts } G))^2$  **for**  
 $\varepsilon :: \text{real}$

**lemma** *irregular-set-subset*:  $\varepsilon\text{-irregular-set } G P \subseteq P \times P$   
**by** (*auto simp: irregular-set-def*)

**lemma** *irregular-set-swap*:  $(i,j) \in \varepsilon\text{-irregular-set } G P \longleftrightarrow (j,i) \in \varepsilon\text{-irregular-set } G P$   
**by** (*auto simp add: irregular-set-def regular-pair-commute*)

**lemma** *finite-irregular-set [simp]*:  $\text{finite } P \implies \text{finite } (\varepsilon\text{-irregular-set } G P)$   
**by** (*metis finite-SigmaI finite-subset irregular-set-subset*)

## 1.4 Energy of a Graph

Definition 3.7 (Energy), written  $q(U, W)$

**definition** *energy-graph-subsets*::  $[\text{uvert set}, \text{uvert set}, \text{ugraph}] \Rightarrow \text{real}$  **where**  
 $\text{energy-graph-subsets } U W G \equiv$   
 $\text{card } U * \text{card } W * (\text{edge-density } U W G)^2 / (\text{card } (\text{uverts } G))^2$

Definition for partitions

**definition** *energy-graph-partitions* ::  $[\text{ugraph}, \text{uvert set set}, \text{uvert set set}] \Rightarrow \text{real}$   
**where**  $\text{energy-graph-partitions } G P Q \equiv \sum R \in P . \sum S \in Q . \text{energy-graph-subsets } R S G$

**lemma** *energy-graph-subsets-0 [simp]*:  
 $\text{energy-graph-subsets } \{ \} B G = 0$   $\text{energy-graph-subsets } A \{ \} G = 0$   
**by** (*auto simp: energy-graph-subsets-def*)

**lemma** *energy-graph-subsets-ge0 [simp]*:  
 $\text{energy-graph-subsets } U W G \geq 0$   
**by** (*auto simp: energy-graph-subsets-def*)

**lemma** *energy-graph-partitions-ge0 [simp]*:  
 $\text{energy-graph-partitions } G U W \geq 0$   
**by** (*auto simp: sum-nonneg energy-graph-partitions-def*)

**lemma** *energy-graph-subsets-commute*:  
 $\text{energy-graph-subsets } U W G = \text{energy-graph-subsets } W U G$   
**by** (*simp add: energy-graph-subsets-def edge-density-commute*)

**lemma** *energy-graph-partitions-commute*:  
 $\text{energy-graph-partitions } G W U = \text{energy-graph-partitions } G U W$   
**by** (*simp add: energy-graph-partitions-def energy-graph-subsets-commute sum.swap*)  
**[where  $A=W$ ]**

Definition 3.7 (Energy of a Partition), or following Gowers, mean square density: a version of energy for a single partition of the vertex set.

**abbreviation** *mean-square-density* :: [ugraph, uvert set set] ⇒ real  
**where** *mean-square-density*  $G P \equiv \text{energy-graph-partitions } G P P$

**lemma** *mean-square-density*:

*mean-square-density*  $G U \equiv$   
 $(\sum R \in U. \sum S \in U. \text{card } R * \text{card } S * (\text{edge-density } R S G)^2) / (\text{card } (\text{uverts } G))^2$   
**by** (*simp add: energy-graph-partitions-def energy-graph-subsets-def sum-divide-distrib*)

Observation: the energy is between 0 and 1 because the edge density is bounded above by 1.

**lemma** *sum-partition-le*:

**assumes** *finite-graph-partition*  $V P k$  *finite*  $V$

**shows**  $(\sum R \in P. \sum S \in P. \text{real } (\text{card } R * \text{card } S)) \leq (\text{real } (\text{card } V))^2$

**proof** –

**have** *finite*  $P$

**using** *assms finite-graph-partition-def* **by** *blast*

**then show** *?thesis*

**using** *assms*

**proof** (*induction*  $P$  *arbitrary: V k*)

**case** (*insert*  $X P$ )

**have** [*simp*]: *finite*  $Y$  **if**  $Y \in \text{insert } X P$  **for**  $Y$

**by** (*meson finite-graph-partition-finite insert.prem* *that*)

**have**  $C$ :  $\text{card } Y \leq \text{card } V$  **if**  $Y \in \text{insert } X P$  **for**  $Y$

**by** (*meson card-mono finite-graph-partition-subset insert.prem* *that*)

**have**  $D$  [*simp*]:  $(\sum Y \in P. \text{real } (\text{card } Y)) = \text{real } (\text{card } V) - \text{real } (\text{card } X)$

**by** (*smt (verit) card-finite-graph-partition insert.hyps insert.prem of-nat-sum sum.cong sum.insert*)

**have** *disjnt*  $X (\bigcup P)$

**using** *insert.prem insert.hyps*

**by** (*auto simp add: finite-graph-partition-def disjnt-iff pairwise-insert partition-on-def*)

**with** *insert* **have**  $*$ :  $(\sum R \in P. \sum S \in P. \text{real } (\text{card } R * \text{card } S)) \leq (\text{real } (\text{card } (V - X)))^2$

**unfolding** *finite-graph-partition-def*

**by** (*simp add: lessThan-Suc partition-on-insert disjoint-family-on-insert sum.distrib*)

**have** [*simp*]:  $V \cap X = X$

**using** *finite-graph-partition-equals insert.prem* **by** *blast*

**have**  $(\sum R \in \text{insert } X P. \sum S \in \text{insert } X P. \text{real } (\text{card } R * \text{card } S))$

$= \text{real } (\text{card } X * \text{card } X) + 2 * (\text{card } V - \text{card } X) * \text{card } X$

$+ (\sum R \in P. \sum S \in P. \text{real } (\text{card } R * \text{card } S))$

**using**  $\langle X \notin P \rangle$   $\langle \text{finite } P \rangle$

**by** (*simp add: C of-nat-diff sum.distrib algebra-simps flip: sum-distrib-right*)

**also have**  $\dots \leq \text{real } (\text{card } X * \text{card } X) + 2 * (\text{card } V - \text{card } X) * \text{card } X + (\text{real } (\text{card } (V - X)))^2$

**using**  $*$  **by** *linarith*

**also have**  $\dots \leq (\text{real } (\text{card } V))^2$

by (*simp add: of-nat-diff C card-Diff-subset-Int algebra-simps power2-eq-square*)  
 finally show ?case .  
 qed auto  
 qed

**lemma** *mean-square-density-bounded:*

assumes *finite-graph-partition (uverts G) P k finite (uverts G)*

shows *mean-square-density G P ≤ 1*

**proof** –

have  $(\sum R \in P. \sum S \in P. \text{real} (\text{card } R * \text{card } S) * (\text{edge-density } R \ S \ G)^2)$   
 $\leq (\sum R \in P. \sum S \in P. \text{real} (\text{card } R * \text{card } S))$

by (*intro sum-mono mult-right-le-one-le*) (*auto simp: abs-square-le-1 edge-density-ge0 edge-density-le1*)

also have  $\dots \leq (\text{real}(\text{card } (\text{uverts } G)))^2$

using *sum-partition-le assms by blast*

finally show ?thesis

by (*simp add: mean-square-density divide-simps*)

qed

## 1.5 Partitioning and Energy

See Gowers’s remark after Lemma 11. Further partitioning of subsets of the vertex set cannot make the energy decrease. We follow Gowers’s proof, which avoids the use of probability.

**lemma** *sum-products-le:*

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

assumes  $\bigwedge i. i \in I \implies a \ i \geq 0$

shows  $(\sum i \in I. a \ i * b \ i)^2 \leq (\sum i \in I. a \ i) * (\sum i \in I. a \ i * (b \ i)^2)$  (*is ?L ≤ ?R*)

**proof** –

have  $?L = (\sum i \in I. \text{sqrt } (a \ i) * (\text{sqrt } (a \ i) * b \ i))^2$

by (*smt (verit, ccfv-SIG) assms mult.assoc real-sqrt-mult-self sum.cong*)

also have  $\dots \leq (\sum i \in I. (\text{sqrt } (a \ i))^2) * (\sum i \in I. (\text{sqrt } (a \ i) * b \ i)^2)$

by (*rule Cauchy-Schwarz-ineq-sum*)

also have  $\dots = ?R$

by (*smt (verit) assms mult.assoc mult.commute power2-eq-square real-sqrt-pow2 sum.cong*)

finally show ?thesis .

qed

**lemma** *energy-graph-partition-half:*

assumes  $P: \text{finite-graph-partition } U \ P \ n$

shows  $\text{card } U * (\text{edge-density } U \ W \ G)^2 \leq (\sum R \in P. \text{card } R * (\text{edge-density } R \ W \ G)^2)$

**proof** (*cases finite U*)

case *True*

have  $\S: (\sum R \in P. \text{card } R * \text{edge-density } R \ W \ G)^2$

$\leq (\text{sum card } P) * (\sum R \in P. \text{card } R * (\text{edge-density } R \ W \ G)^2)$

by (*simp add: sum-products-le*)

have  $\text{card } U * (\text{edge-density } U \ W \ G)^2 = (\sum R \in P. \text{card } R * (\text{edge-density } U \ W \ G)^2)$

$G)^2$ )  
**by** (*metis*  $\langle$ finite  $U \rangle$   $P$  *sum-distrib-right card-finite-graph-partition of-nat-sum*)  
**also have**  $\dots = \text{edge-density } U \ W \ G * (\sum R \in P. \text{edge-density } U \ W \ G * \text{card } R)$   
**by** (*simp add: sum-distrib-left power2-eq-square mult-ac*)  
**also have**  $\dots = (\sum R \in P. \text{edge-density } R \ W \ G * \text{real } (\text{card } R)) * \text{edge-density } U \ W \ G$   
**proof** –  
**have**  $\text{edge-density } U \ W \ G * (\sum R \in P. \text{edge-density } R \ W \ G * \text{card } R)$   
 $= \text{edge-density } U \ W \ G * (\text{edge-density } U \ W \ G * (\sum R \in P. \text{card } R))$   
**using**  $\langle$ finite  $U \rangle$  *assms card-finite-graph-partition* **by** (*auto simp: edge-density-partition [OF P]*)  
**then show** *?thesis*  
**by** (*simp add: mult.commute sum-distrib-left*)  
**qed**  
**also have**  $\dots = (\sum R \in P. \text{card } R * \text{edge-density } R \ W \ G) * \text{edge-density } U \ W \ G$   
**by** (*simp add: sum-distrib-left mult-ac*)  
**also have**  $\dots = (\sum R \in P. \text{card } R * \text{edge-density } R \ W \ G)^2 / \text{card } U$   
**using** *assms* **by** (*simp add: edge-density-partition [OF P] mult-ac flip: power2-eq-square*)  
**also have**  $\dots \leq (\sum R \in P. \text{card } R * (\text{edge-density } R \ W \ G)^2)$   
**using**  $\S P$  *card-finite-graph-partition*  $\langle$ finite  $U \rangle$   
**by** (*force simp add: mult-ac divide-simps simp flip: of-nat-sum*)  
**finally show** *?thesis* .  
**qed** (*simp add: sum-nonneg*)

**proposition** *energy-graph-partition-increase:*

**assumes**  $P$ : *finite-graph-partition*  $U \ P \ k$  **and**  $V$ : *finite-graph-partition*  $W \ Q \ l$   
**shows** *energy-graph-partitions*  $G \ P \ Q \geq$  *energy-graph-subsets*  $U \ W \ G$

**proof** –

**have**  $(\text{card } U * \text{card } W) * (\text{edge-density } U \ W \ G)^2 = \text{card } W * (\text{card } U * (\text{edge-density } U \ W \ G)^2)$

**by** (*simp add: mult-ac*)

**also have**  $\dots \leq \text{card } W * (\sum R \in P. \text{card } R * (\text{edge-density } R \ W \ G)^2)$

**by** (*intro mult-left-mono energy-graph-partition-half*) (*use assms in auto*)

**also have**  $\dots = (\sum R \in P. \text{card } R * (\text{card } W * (\text{edge-density } W \ R \ G)^2))$

**by** (*simp add: sum-distrib-left edge-density-commute mult-ac*)

**also have**  $\dots \leq (\sum R \in P. \text{card } R * (\sum S \in Q. \text{card } S * (\text{edge-density } S \ R \ G)^2))$

**by** (*intro mult-left-mono energy-graph-partition-half sum-mono*) (*use assms in auto*)

**also have**  $\dots \leq (\sum R \in P. \sum S \in Q. (\text{card } R * \text{card } S) * (\text{edge-density } R \ S \ G)^2)$

**by** (*simp add: sum-distrib-left edge-density-commute mult-ac*)

**finally**

**have**  $(\text{card } U * \text{card } W) * (\text{edge-density } U \ W \ G)^2$

$\leq (\sum R \in P. \sum S \in Q. (\text{card } R * \text{card } S) * (\text{edge-density } R \ S \ G)^2)$  .

**then show** *?thesis*

**unfolding** *energy-graph-partitions-def energy-graph-subsets-def*

**by** (*simp add: divide-simps flip: sum-divide-distrib*)

**qed**

The following is the fully general version of Gowers’s Lemma 11 Further partitioning of subsets of the vertex set cannot make the energy decrease.

Note that  $V$  should be *uverts*  $G$  even though this more general version holds.

**lemma** *energy-graph-partitions-increase-half*:

**assumes** *ref*: *refines*  $V Q P$  **and** *finite*  $V$  **and** *part-VP*: *partition-on*  $V P$   
**and**  $U: \{\} \notin U$

**shows** *energy-graph-partitions*  $G Q U \geq$  *energy-graph-partitions*  $G P U$   
**(is**  $?egQ \geq ?egP$ **)**

**proof** –

**have**  $\exists F$ . *partition-on*  $R F \wedge F = \{S \in Q. S \subseteq R\}$  **if**  $R \in P$  **for**  $R$

**using** *ref refines-obtains-subset that* **by** *blast*

**then obtain**  $F$  **where**  $F: \bigwedge R. R \in P \implies$  *partition-on*  $R (F R) \wedge F R = \{S \in Q. S \subseteq R\}$

**by** *fastforce*

**have** *injF*: *inj-on*  $F P$

**by** (*metis*  $F$  *inj-on-inverseI* *partition-on-def*)

**have** *finite-P*: *finite*  $R$  **if**  $R \in P$  **for**  $R$

**by** (*metis* *Union-upper*  $\langle$ *finite*  $V$  $\rangle$  *part-VP* *finite-subset* *partition-on-def that*)

**then have** *finite-F*: *finite*  $(F R)$  **if**  $R \in P$  **for**  $R$

**using** *that* **by** (*simp* *add*:  $F$ )

**have** *dFP*: *disjoint*  $(F \text{ ' } P)$

**using** *part-VP*

**by** (*smt* (*verit*, *best*)  $F$  *Union-upper* *disjnt-iff* *disjointD* *le-inf-iff* *pairwise-imageI* *partition-on-def* *subset-empty*)

**have** *F-ne*:  $F R \neq \{\}$  **if**  $R \in P$  **for**  $R$

**by** (*metis*  $F$  *Sup-empty* *part-VP* *partition-on-def that*)

**have** *F-sums-Q*:  $(\sum R \in P. \sum U \in F R. f U) = (\sum S \in Q. f S)$  **for**  $f :: \text{nat set} \Rightarrow \text{real}$

**proof** –

**have**  $Q = (\bigcup R \in P. F R)$

**using** *ref* **by** (*force* *simp* *add*: *refines-def* *dest*:  $F$ )

**then have**  $(\sum S \in Q. f S) = \text{sum } f (\bigcup R \in P. F R)$

**by** *blast*

**also have**  $\dots = (\text{sum} \circ \text{sum}) f (F \text{ ' } P)$

**by** (*smt* (*verit*, *best*) *dFP* *disjnt-def* *finite-F* *image-iff* *pairwiseD* *sum.Union-disjoint*)

**also have**  $\dots = (\sum R \in P. \sum U \in F R. f U)$

**unfolding** *comp-apply* **by** (*metis* *injF* *sum.reindex-cong*)

**finally show** *?thesis*

**by** *simp*

**qed**

**have**  $?egP = (\sum R \in P. \sum T \in U. \text{energy-graph-subsets } R T G)$

**by** (*simp* *add*: *energy-graph-partitions-def*)

**also have**  $\dots \leq (\sum R \in P. \sum T \in U. \text{energy-graph-partitions } G (F R) \{T\})$

**proof** –

**have** *finite-graph-partition*  $R (F R)$  (*card*  $(F R)$ )

**if**  $R \in P$  **for**  $R$

**by** (*meson*  $F$  *finite-F* *finite-graph-partition-def that*)

**moreover have** *finite-graph-partition*  $T \{T\}$  (*Suc*  $0$ )

**if**  $T \in U$  **for**  $T$

**using**  $U$  **by** (*metis* *that* *trivial-graph-partition-exists*)

**ultimately show** *?thesis*

**using** *finite-P* **by** (*intro sum-mono energy-graph-partition-increase*) *auto*  
**qed**  
**also have**  $\dots = (\sum R \in P. \sum D \in F R. \sum T \in U. \text{energy-graph-subsets } D T G)$   
**by** (*simp add: energy-graph-partitions-def sum.swap [where B = U]*)  
**also have**  $\dots = ?egQ$   
**by** (*simp add: energy-graph-partitions-def F-sums-Q*)  
**finally show** *?thesis* .  
**qed**

**proposition** *energy-graph-partitions-increase:*

**assumes** *refines V Q P refines V' Q' P'*  
**and** *finite V finite V'*  
**shows** *energy-graph-partitions G Q Q' ≥ energy-graph-partitions G P P'*  
**proof** –  
**obtain**  $\{\} \notin P' \{\} \notin Q$   
**using** *assms unfolding refines-def partition-on-def by presburger*  
**then show** *?thesis*  
**using** *assms unfolding refines-def*  
**by** (*smt (verit, ccfv-SIG) assms energy-graph-partitions-commute energy-graph-partitions-increase-half*)  
**qed**

The original version of Gowers’s Lemma 11 (also in Zhao) is not general enough to be used for anything.

**corollary** *mean-square-density-increase:*

**assumes** *refines V Q P finite V*  
**shows** *mean-square-density G Q ≥ mean-square-density G P*  
**using** *assms energy-graph-partitions-increase by presburger*

The Energy Boost Lemma says that an irregular partition increases the energy substantially. We assume that  $\mathcal{U} \subseteq \text{uverts } G$  and  $\mathcal{W} \subseteq \text{uverts } G$  are not irregular, as witnessed by their subsets  $U1 \subseteq \mathcal{U}$  and  $W1 \subseteq \mathcal{W}$ . The proof follows Lemma 12 of Gowers.

**definition** *part2 X Y*  $\equiv$  *if*  $X \subset Y$  *then*  $\{X, Y-X\}$  *else*  $\{Y\}$

**lemma** *card-part2:*  $\text{card } (\text{part2 } X Y) \leq 2$   
**by** (*simp add: part2-def card-insert-if*)

**lemma** *sum-part2:*  $\llbracket X \subseteq Y; f\{\} = 0 \rrbracket \implies \text{sum } f (\text{part2 } X Y) = f X + f (Y-X)$   
**by** (*force simp add: part2-def sum.insert-if*)

**lemma** *partition-part2:*

**assumes**  $A \subseteq B$   $A \neq \{\}$   
**shows** *partition-on B (part2 A B)*  
**using** *assms by (auto simp add: partition-on-def part2-def disjnt-iff pairwise-insert)*

**proposition** *energy-boost:*

**fixes**  $\varepsilon::\text{real}$  **and**  $U W G$   
**defines**  $\alpha \equiv \text{edge-density } U W G$   
**defines**  $u \equiv \lambda X Y. \text{edge-density } X Y G - \alpha$

**assumes** *finite U finite W*  
**and**  $U' \subseteq U \ W' \subseteq W \ \varepsilon > 0$   
**and**  $U': \text{card } U' \geq \varepsilon * \text{card } U$  **and**  $W': \text{card } W' \geq \varepsilon * \text{card } W$   
**and**  $gt: |u \ U' \ W'| > \varepsilon$   
**shows**  $(\sum A \in \text{part2 } U' \ U. \sum B \in \text{part2 } W' \ W. \text{energy-graph-subsets } A \ B \ G)$   
 $\geq \text{energy-graph-subsets } U \ W \ G + \varepsilon^4 * (\text{card } U * \text{card } W) / (\text{card } (\text{uverts } G))^2$   
*(is ?lhs  $\geq$  ?rhs)*

**proof** –  
**define** *UF* **where**  $UF \equiv \text{part2 } U' \ U$   
**define** *WF* **where**  $WF \equiv \text{part2 } W' \ W$   
**obtain** [*simp*]: *finite U finite W*  
**using** *assms* **by** (*meson finite-subset*)  
**obtain**  $\text{card}'$ :  $\text{card } U' > 0 \ \text{card } W' > 0$   
**using** *gt*  $\langle \varepsilon > 0 \rangle \ U' \ W'$   
**by** (*force simp: u-def alpha-def edge-density-def mult-le-0-iff zero-less-mult-iff*)  
**then obtain** *card*:  $\text{card } U > 0 \ \text{card } W > 0$   
**using** *assms* **by** *fastforce*  
**then obtain** [*simp*]: *finite U' finite W'*  
**by** (*meson card' card-ge-0-finite*)  
**obtain** [*simp*]:  $W' \neq W - W' \ U' \neq U - U'$   
**by** (*metis DiffD2 card' all-not-in-conv card.empty less-irrefl*)  
**have** *UF-ne*:  $\text{card } x \neq 0$  **if**  $x \in UF$  **for**  $x$   
**using**  $\text{card}'$  *assms* **that** **by** (*auto simp: UF-def part2-def split: if-split-asm*)  
**have** *WF-ne*:  $\text{card } x \neq 0$  **if**  $x \in WF$  **for**  $x$   
**using**  $\text{card}'$  *assms* **that** **by** (*auto simp: WF-def part2-def split: if-split-asm*)  
**have** *cardUW*:  $\text{card } U = \text{card } U' + \text{card}(U - U') \ \text{card } W = \text{card } W' + \text{card}(W - W')$   
**using** *card card'*  $\langle U' \subseteq U \rangle \ \langle W' \subseteq W \rangle$   
**by** (*metis card-eq-0-iff card-Diff-subset card-mono le-add-diff-inverse less-le*)  
**have**  $U = (U - U') \cup U' \ \text{disjnt } (U - U') \ U'$   
**using**  $\langle U' \subseteq U \rangle$  **by** (*force simp: disjnt-iff*)  
**then have** *CU*:  $\text{card } (\text{all-edges-between } U \ Z \ G)$   
 $= \text{card } (\text{all-edges-between } (U - U') \ Z \ G) + \text{card } (\text{all-edges-between } U' \ Z \ G)$   
**if** *finite Z* **for**  $Z$   
**by** (*metis*  $\langle \text{finite } U' \rangle \ \text{all-edges-between-Un1} \ \text{all-edges-between-disjnt1} \ \langle \text{finite } U \rangle$   
 $\text{card-Un-disjnt} \ \text{finite-Diff} \ \text{finite-all-edges-between} \ \text{that}$ )

**have**  $W = (W - W') \cup W' \ \text{disjnt } (W - W') \ W'$   
**using**  $\langle W' \subseteq W \rangle$  **by** (*force simp: disjnt-iff*)  
**then have** *CW*:  $\text{card } (\text{all-edges-between } Z \ W \ G)$   
 $= \text{card } (\text{all-edges-between } Z \ (W - W') \ G) + \text{card } (\text{all-edges-between } Z \ W' \ G)$   
**if** *finite Z* **for**  $Z$   
**by** (*metis*  $\langle \text{finite } W' \rangle \ \text{all-edges-between-Un2} \ \text{all-edges-between-disjnt2} \ \langle \text{finite } W \rangle$   
 $\text{card-Un-disjnt} \ \text{finite-Diff2} \ \text{finite-all-edges-between} \ \text{that}$ )

**have** \*:  $(\sum X \in UF. \sum Y \in WF. \text{real}(\text{card}(\text{all-edges-between } X \ Y \ G)))$   
 $= \text{card}(\text{all-edges-between } U \ W \ G)$   
**by** (*simp add: UF-def WF-def cardUW CU CW sum-part2*  $\langle U' \subseteq U \rangle \langle W' \subseteq W \rangle$ )  
**have** \*\*:  $\text{real}(\text{card } U) * \text{real}(\text{card } W) = (\sum X \in UF. \sum Y \in WF. \text{card } X * \text{card } Y)$   
**by** (*simp add: UF-def WF-def cardUW sum-part2*  $\langle U' \subseteq U \rangle \langle W' \subseteq W \rangle$  *algebra-simps*)  
  
**let** ?S =  $\sum X \in UF. \sum Y \in WF. (\text{card } X * \text{card } Y) / (\text{card } U * \text{card } W) * (\text{edge-density } X \ Y \ G)^2$   
**define** T **where**  $T \equiv (\sum X \in UF. \sum Y \in WF. (\text{card } X * \text{card } Y) / (\text{card } U * \text{card } W) * (\text{edge-density } X \ Y \ G))$   
**have** §:  $2 * T = \text{alpha} + \text{alpha} * (\sum X \in UF. \sum Y \in WF. (\text{card } X * \text{card } Y) / (\text{card } U * \text{card } W))$   
**unfolding** *alpha-def T-def*  
**by** (*simp add: \* \*\* edge-density-def divide-simps sum-part2*  $\langle U' \subseteq U \rangle \langle W' \subseteq W \rangle$  *UF-ne WF-ne flip: sum-divide-distrib*)  
**have**  $\varepsilon * \varepsilon \leq u \ U' \ W' * u \ U' \ W'$   
**by** (*metis abs-ge-zero abs-mult-self-eq*  $\langle \varepsilon > 0 \rangle$  *gt less-le mult-mono*)  
**then have**  $(\varepsilon * \varepsilon) * (\varepsilon * \varepsilon) \leq (\text{card } U' * \text{card } W') / (\text{card } U * \text{card } W) * (u \ U' \ W')^2$   
**using** *card mult-mono [OF U' W']*  $\langle \varepsilon > 0 \rangle$   
**apply** (*simp add: divide-simps eval-nat-numeral*)  
**by** (*smt (verit, del-insts) mult.assoc mult.commute mult-mono' of-nat-0-le-iff zero-le-mult-iff*)  
**also have**  $\dots \leq (\sum X \in UF. \sum Y \in WF. (\text{card } X * \text{card } Y) / (\text{card } U * \text{card } W) * (u \ X \ Y)^2)$   
**by** (*simp add: UF-def WF-def sum-part2*  $\langle U' \subseteq U \rangle \langle W' \subseteq W \rangle$ )  
**also have**  $\dots = ?S - 2 * T * \text{alpha} + \text{alpha}^2 * (\sum X \in UF. \sum Y \in WF. (\text{card } X * \text{card } Y) / (\text{card } U * \text{card } W))$   
**by** (*simp add: u-def T-def power2-diff mult-ac ring-distrib divide-simps sum-distrib-left sum-distrib-right sum-subtractf sum.distrib flip: sum-divide-distrib*)  
**also have**  $\dots = ?S - \text{alpha}^2$   
**using** § **by** (*simp add: power2-eq-square algebra-simps*)  
**finally have** 12:  $\text{alpha}^2 + \varepsilon^4 \leq ?S$   
**by** (*simp add: eval-nat-numeral*)  
**have** ?rhs =  $(\text{alpha}^2 + \varepsilon^4) * (\text{card } U * \text{card } W / (\text{card}(\text{uverts } G))^2)$   
**unfolding** *alpha-def energy-graph-subsets-def*  
**by** (*simp add: ring-distrib divide-simps power2-eq-square*)  
**also have**  $\dots \leq ?S * (\text{card } U * \text{card } W / (\text{card}(\text{uverts } G))^2)$   
**by** (*rule mult-right-mono [OF 12]*) *auto*  
**also have**  $\dots = ?lhs$   
**using** *card unfolding energy-graph-subsets-def UF-def WF-def*  
**by** (*auto simp add: algebra-simps sum-part2*  $\langle U' \subseteq U \rangle \langle W' \subseteq W \rangle$ )  
**finally show** ?thesis .  
**qed**

## 1.6 Energy boost for partitions

We can always find a refinement that increases the energy by a certain amount.

A necessary lemma for the tower of exponentials in the result. Angeliki's proof

```

lemma le-tower-2:  $k * (2 \wedge \text{Suc } k) \leq 2 \wedge (2 \wedge k)$ 
proof (induction k rule: less-induct)
  case (less k)
  show ?case
  proof (cases k ≤ Suc (Suc 0))
    case False
    define j where  $j = k - \text{Suc } 0$ 
    have kj:  $k = \text{Suc } j$ 
    using False j-def by force
    with False have  $\S$ :  $(2 \wedge j + 3) \leq (2::\text{nat}) \wedge k$ 
    by (simp add: Suc-leI le-less-trans not-less-eq-eq numeral-3-eq-3)
    have  $k * (2 \wedge \text{Suc } k) \leq 6 * j * 2 \wedge j$ 
    using False by (simp add: kj)
    also have  $\dots \leq 6 * 2 \wedge (2 \wedge j)$ 
    using kj less.IH by force
    also have  $\dots < 2 \wedge (2 \wedge j + 3)$ 
    by (simp add: power-add)
    also have  $\dots \leq 2 \wedge 2 \wedge k$ 
    by (simp add: \S)
    finally show ?thesis
    by simp
  qed (auto simp: le-Suc-eq)
qed

```

The bound  $2^{k+1}$  comes from a different source by Zhao: “Graph Theory and Additive Combinatorics”, <https://yufeizhao.com/gtacbook/>. It's needed because our *regular-partition* includes the diagonal; otherwise,  $k2^k$  would work. Gowers' version has a flatly incorrect bound.

**proposition** *exists-refinement*:

```

assumes fpp: finite-graph-partition (uverts G) P k and finite (uverts G)
and irreg:  $\neg \varepsilon$ -regular-partition G P and  $\varepsilon > 0$ 
obtains Q where refines (uverts G) Q P
   $\text{mean-square-density } G \ Q \geq \text{mean-square-density } G \ P + \varepsilon^5$ 
   $\bigwedge R. R \in P \implies \text{card } \{S \in Q. S \subseteq R\} \leq 2 \wedge \text{Suc } k$ 
   $\text{card } Q \leq k * 2 \wedge \text{Suc } k$ 

```

**proof** –

```

define sum-pp where  $\text{sum-pp} \equiv (\sum (R,S) \in \varepsilon$ -irregular-set G P.  $\text{card } R * \text{card } S$ )
have cardP:  $\text{card } P = k$ 
using fpp finite-graph-partition-def by force
then have  $k \neq 0$ 
using assms unfolding regular-partition-def irregular-set-def finite-graph-partition-def

```

**by** *fastforce*  
**with** *assms* **have** *G-nonempty*:  $0 < \text{card } (\text{uverts } G)$   
**by** (*metis card-gt-0-iff finite-graph-partition-empty*)  
**have** *part-GP*: *partition-on* (*uverts* *G*) *P*  
**using** *fgp finite-graph-partition-def* **by** *blast*  
**then** **have** *finP*: *finite* *R*  $R \neq \{\}$  **if**  $R \in P$  **for** *R*  
**using** *assms that partition-onD3 finite-graph-partition-finite* **by** *blast+*  
**have** *spp*:  $\text{sum-pp} > \varepsilon * (\text{card } (\text{uverts } G))^2$   
**by** (*metis irreg not-le part-GP regular-partition-def sum-pp-def*)  
**then** **have** *sum-irreg-pos*:  $\text{sum-pp} > 0$   
**using**  $\langle \varepsilon > 0 \rangle$  *G-nonempty less-asm* **by** *fastforce*  
**have**  $\exists X \subseteq R. \exists Y \subseteq S. \varepsilon * \text{card } R \leq \text{card } X \wedge \varepsilon * \text{card } S \leq \text{card } Y \wedge$   
 $|\text{edge-density } X Y G - \text{edge-density } R S G| > \varepsilon$   
**if**  $(R, S) \in \varepsilon\text{-irregular-set } G P$  **for** *R S*  
**using** *that fgp finite-graph-partition-subset* **by** (*simp add: irregular-set-def regular-pair-def not-le*)  
**then** **obtain** *X0 Y0*  
**where** *XY0-psub-P*:  $\bigwedge R S. [(R, S) \in \varepsilon\text{-irregular-set } G P] \implies X0 R S \subseteq R \wedge$   
 $Y0 R S \subseteq S$   
**and** *XY0-eps*:  
 $\bigwedge R S. (R, S) \in \varepsilon\text{-irregular-set } G P$   
 $\implies \varepsilon * \text{card } R \leq \text{card } (X0 R S) \wedge \varepsilon * \text{card } S \leq \text{card } (Y0 R S) \wedge$   
 $|\text{edge-density } (X0 R S) (Y0 R S) G - \text{edge-density } R S G| > \varepsilon$   
**by** *metis*  
**obtain** *iP* **where** *iP*: *bij-betw* *iP* *P*  $\{..<k\}$   
**by** (*metis fgp finite-graph-partition-def to-nat-on-finite cardP*)  
**define** *X* **where**  $X \equiv \lambda R S. \text{if } iP R < iP S \text{ then } Y0 S R \text{ else } X0 R S$   
**define** *Y* **where**  $Y \equiv \lambda R S. \text{if } iP R < iP S \text{ then } X0 S R \text{ else } Y0 R S$   
**have** *XY-psub-P*:  $\bigwedge R S. [(R, S) \in \varepsilon\text{-irregular-set } G P] \implies X R S \subseteq R \wedge Y R$   
 $S \subseteq S$   
**using** *XY0-psub-P* **by** (*force simp: X-def Y-def irregular-set-swap*)  
**have** *XY-eps*:  
 $\bigwedge R S. (R, S) \in \varepsilon\text{-irregular-set } G P$   
 $\implies \varepsilon * \text{card } R \leq \text{card } (X R S) \wedge \varepsilon * \text{card } S \leq \text{card } (Y R S) \wedge$   
 $|\text{edge-density } (X R S) (Y R S) G - \text{edge-density } R S G| > \varepsilon$   
**using** *XY0-eps* **by** (*force simp: X-def Y-def edge-density-commute irregular-set-swap*)  
**have** *card-elem-P*:  $\text{card } R > 0$  **if**  $R \in P$  **for** *R*  
**by** (*metis card-eq-0-iff finP neq0-conv that*)  
**have** *XY-nonempty*:  $X R S \neq \{\}$   $Y R S \neq \{\}$  **if**  $(R, S) \in \varepsilon\text{-irregular-set } G P$   
**for** *R S*  
**using** *XY-eps [OF that]* *that*  $\langle \varepsilon > 0 \rangle$  *card-elem-P [of R]* *card-elem-P [of S]*  
**by** (*auto simp: irregular-set-def mult-le-0-iff*)

By the assumption that our partition is irregular, there are many irregular pairs. For each irregular pair, find pairs of subsets that witness irregularity.

**define** *XP* **where**  $XP R \equiv ((\lambda S. \text{part2 } (X R S) R) ' \{S. (R, S) \in \varepsilon\text{-irregular-set } G P\})$  **for** *R*

**define** *YP* **where**  $YP\ S \equiv ((\lambda R. \text{part2 } (Y\ R\ S)\ S) \text{ ‘ } \{R. (R,S) \in \varepsilon\text{-irregular-set } G\ P\})$  **for** *S*

include degenerate partition to ensure it works whether or not there’s an irregular pair

**define** *PP* **where**  $PP \equiv \lambda R. \text{insert } \{R\} (XP\ R \cup YP\ R)$   
**define** *QS* **where**  $QS\ R \equiv \text{common-refinement } (PP\ R)$  **for** *R*  
**define** *r* **where**  $r\ R \equiv \text{card } (QS\ R)$  **for** *R*  
**have** *finite P*  
**using** *fpp finite-graph-partition-def* **by** *blast*  
**then have** *finPP: finite (PP R)* **for** *R*  
**by** (*simp add: PP-def XP-def YP-def irregular-set-def*)  
**have** *inPP-fin: P ∈ PP R ⇒ finite P* **for** *P R*  
**by** (*auto simp: PP-def XP-def YP-def part2-def*)  
**have** *finite-QS: finite (QS R)* **for** *R*  
**by** (*simp add: QS-def finPP finite-common-refinement inPP-fin*)

**have** *part-QS: partition-on R (QS R) if R ∈ P* **for** *R*  
**unfolding** *QS-def*  
**proof** (*intro partition-on-common-refinement partition-onI*)  
**show**  $\bigwedge \mathcal{A}. \mathcal{A} \in PP\ R \implies \{\} \notin \mathcal{A}$   
**using** *that XY-nonempty XY-psub-P finP*  
**by** (*fastforce simp add: PP-def XP-def YP-def part2-def*)  
**qed** (*auto simp: disjnt-iff PP-def XP-def YP-def part2-def dest: XY-psub-P*)

**have** *part-P-QS: finite-graph-partition R (QS R) (r R) if R ∈ P* **for** *R*  
**by** (*simp add: finite-QS finite-graph-partition-def part-QS r-def that*)  
**then have** *fin-SQ [simp]: finite (QS R) if R ∈ P* **for** *R*  
**using** *QS-def finite-QS* **by** *force*  
**have** *QS-ne: {} ∉ QS R if R ∈ P* **for** *R*  
**using** *QS-def part-QS partition-onD3* **that** **by** *blast*  
**have** *QS-subset-P: q ∈ QS R ⇒ q ⊆ R* **if** *R ∈ P* **for** *R q*  
**by** (*meson finite-graph-partition-subset part-P-QS that*)  
**then have** *QS-inject: R = R'*  
**if** *R ∈ P R' ∈ P q ∈ QS R q ∈ QS R'* **for** *R R' q*  
**by** (*metis UnionI disjnt-iff equalsOI pairwiseD part-GP part-QS partition-on-def that*)

**define** *Q* **where**  $Q \equiv (\bigcup R \in P. QS\ R)$   
**define** *m* **where**  $m \equiv \sum R \in P. r\ R$   
**show** *thesis*

**proof**  
**show** *ref-QP: refines (uverts G) Q P*  
**unfolding** *refines-def*  
**proof** (*intro conjI strip part-GP*)  
**fix** *X*  
**assume**  $X \in Q$   
**then show**  $\exists Y \in P. X \subseteq Y$   
**by** (*metis QS-subset-P Q-def UN-iff*)  
**next**

```

show partition-on (uverts  $G$ )  $Q$ 
proof (intro conjI partition-onI)
  show  $\bigcup Q = \text{uverts } G$ 
  proof
    show  $\bigcup Q \subseteq \text{uverts } G$ 
    using QS-subset-P Q-def fgp finite-graph-partition-equals by fastforce
    show  $\text{uverts } G \subseteq \bigcup Q$ 
    by (metis Q-def Sup-least UN-upper Union-mono part-GP part-QS
partition-onD1)
  qed
show disjnt  $p$   $q$  if  $p \in Q$  and  $q \in Q$  and  $p \neq q$  for  $p$   $q$ 
proof –
  from that
  obtain  $R$   $S$  where  $R \in P$   $S \in P$ 
  and  $*$ :  $p \in QS$   $R$   $q \in QS$   $S$ 
  by (auto simp: Q-def QS-def)
  show ?thesis
  proof (cases R=S)
    case True
    then show ?thesis
    using part-QS [of R]
    by (metis  $\langle R \in P \rangle$   $*$  pairwiseD partition-on-def  $\langle p \neq q \rangle$ )
  next
  case False
  with  $*$  show ?thesis
  by (metis QS-subset-P  $\langle R \in P \rangle$   $\langle S \in P \rangle$  disjnt-iff pairwiseD part-GP
partition-on-def subsetD)
  qed
qed
show  $\{\}$   $\notin Q$ 
  using QS-ne Q-def by blast
qed
qed
have disj-QSP: disjoint-family-on QS P
  unfolding disjoint-family-on-def by (metis Int-emptyI QS-inject)
let  $?PP = P \times P$ 
let  $?REG = ?PP - \varepsilon\text{-irregular-set } G$   $P$ 
define sum-eps where sum-eps  $\equiv (\sum (R,S) \in \varepsilon\text{-irregular-set } G$   $P. \varepsilon^4 * (\text{card } R * \text{card } S) / (\text{card } (\text{uverts } G))^2)$ 
have  $A$ : energy-graph-subsets  $R$   $S$   $G + \varepsilon^4 * (\text{card } R * \text{card } S) / (\text{card } (\text{uverts } G))^2$ 
   $\leq \text{energy-graph-partitions } G$  (part2 ( $X$   $R$   $S$ )  $R$ ) (part2 ( $Y$   $R$   $S$ )  $S$ )
  (is  $?L \leq ?R$ )
  if  $*$ :  $(R,S) \in \varepsilon\text{-irregular-set } G$   $P$  for  $R$   $S$ 
proof –
  have  $R \in P$   $S \in P$ 
  using  $*$  by (auto simp: irregular-set-def)
  have  $?L \leq (\sum A \in \text{part2 } (X$   $R$   $S)$   $R. \sum B \in \text{part2 } (Y$   $R$   $S)$   $S. \text{energy-graph-subsets } A$   $B$   $G)$ 

```

```

    using XY-psub-P [OF *] XY-eps [OF *] assms
    by (intro energy-boost ⟨R ∈ P⟩ ⟨S ∈ P⟩ finP ⟨ε>0⟩) auto
  also have ... ≤ ?R
    by (simp add: energy-graph-partitions-def)
  finally show ?thesis .
qed
have B: energy-graph-partitions G (part2 (X R S) R) (part2 (Y R S) S)
  ≤ energy-graph-partitions G (QS R) (QS S)
  if (R,S) ∈ ε-irregular-set G P for R S
proof -
  have R∈P S∈P using that by (auto simp: irregular-set-def)
  have [simp]: ¬ X R S ⊂ R ↔ X R S = R ¬ Y R S ⊂ S ↔ Y R S = S
    using XY-psub-P that by blast+
  have XPX: part2 (X R S) R ∈ PP R
    using that by (simp add: PP-def XP-def)
  have I: partition-on R (QS R)
    using QS-def ⟨R ∈ P⟩ part-QS by force
  moreover have ∀ q ∈ QS R. ∃ b ∈ part2 (X R S) R. q ⊆ b
    using common-refinement-exists [OF - XPX] by (simp add: QS-def)
  ultimately have ref-XP: refines R (QS R) (part2 (X R S) R)
    by (simp add: refines-def XY-nonempty XY-psub-P that partition-part2)
  have YPY: part2 (Y R S) S ∈ PP S
    using that by (simp add: PP-def YP-def)
  have J: partition-on S (QS S)
    using QS-def ⟨S ∈ P⟩ part-QS by force
  moreover have ∀ q ∈ QS S. ∃ b ∈ part2 (Y R S) S. q ⊆ b
    using common-refinement-exists [OF - YPY] by (simp add: QS-def)
  ultimately have ref-YP: refines S (QS S) (part2 (Y R S) S)
    by (simp add: XY-nonempty XY-psub-P that partition-part2 refines-def)
  show ?thesis
    using ⟨R ∈ P⟩ ⟨S ∈ P⟩
    by (simp add: finP energy-graph-partitions-increase [OF ref-XP ref-YP])
qed
have mean-square-density G P + ε^5 ≤ mean-square-density G P + sum-eps
proof -
  have ε^5 = (ε * (card (uverts G))^2) * (ε^4 / (card (uverts G))^2)
    using G-nonempty by (simp add: field-simps eval-nat-numeral)
  also have ... ≤ sum-pp * (sum-eps / sum-pp)
  proof (rule mult-mono)
    show ε^4 / real ((card (uverts G))^2) ≤ sum-eps / sum-pp
      using sum-irreg-pos sum-eps-def sum-pp-def
      by (auto simp add: case-prod-unfold sum.neutral simp flip: sum-distrib-left
sum-divide-distrib of-nat-sum of-nat-mult)
    qed (use spp sum-nonneg in auto)
  also have ... ≤ sum-eps
    by (simp add: sum-irreg-pos)
  finally show ?thesis by simp
qed
also have ... = (∑ (i,j) ∈ ?REG. energy-graph-subsets i j G)

```

$+$   $(\sum (i,j) \in \varepsilon\text{-irregular-set } G P. \text{energy-graph-subsets } i j G) +$   
*sum-eps*  
**by** (*simp add: <finite P> energy-graph-partitions-def sum.cartesian-product irregular-set-subset sum.subset-diff*)  
**also have**  $\dots \leq (\sum (i,j) \in ?REG. \text{energy-graph-subsets } i j G)$   
 $+$   $(\sum (i,j) \in \varepsilon\text{-irregular-set } G P. \text{energy-graph-partitions } G (\text{part2 } (X i j) i) (\text{part2 } (Y i j) j))$   
**using** *A unfolding sum-eps-def case-prod-unfold*  
**by** (*force intro: sum-mono simp flip: sum.distrib*)  
**also have**  $\dots \leq (\sum (i,j) \in ?REG. \text{energy-graph-partitions } G (QS i) (QS j))$   
 $+$   $(\sum (i,j) \in \varepsilon\text{-irregular-set } G P. \text{energy-graph-partitions } G (\text{part2 } (X i j) i) (\text{part2 } (Y i j) j))$   
**by** (*auto intro!: part-P-QS sum-mono energy-graph-partition-increase*)  
**also have**  $\dots \leq (\sum (i,j) \in ?REG. \text{energy-graph-partitions } G (QS i) (QS j))$   
 $+$   $(\sum (i,j) \in \varepsilon\text{-irregular-set } G P. \text{energy-graph-partitions } G (QS i) (QS j))$   
**using** *B*  
**proof** (*intro sum-mono add-mono ordered-comm-monoid-add-class.sum-mono2*)  
**qed** (*auto split: prod.split*)  
**also have**  $\dots = (\sum (i,j) \in ?PP. \text{energy-graph-partitions } G (QS i) (QS j))$   
**by** (*metis (no-types, lifting) <finite P> finite-SigmaI irregular-set-subset sum.subset-diff*)  
**also have**  $\dots = (\sum i \in P. \sum j \in P. \text{energy-graph-partitions } G (QS i) (QS j))$   
**by** (*simp flip: sum.cartesian-product*)  
**also have**  $\dots = (\sum A \in Q. \sum B \in Q. \text{energy-graph-subsets } A B G)$   
**unfolding** *energy-graph-partitions-def Q-def*  
**by** (*simp add: disj-QSP <finite P> sum.UNION-disjoint-family sum.swap [of - P QS -]*)  
**also have**  $\dots = \text{mean-square-density } G Q$   
**by** (*simp add: mean-square-density energy-graph-subsets-def sum-divide-distrib*)  
**finally show**  $\text{mean-square-density } G P + \varepsilon \hat{=} 5 \leq \text{mean-square-density } G Q .$

**define** *QinP* **where**  $QinP \equiv \lambda i. \{j \in Q. j \subseteq i\}$   
**show** *card-QP*:  $\text{card } (QinP i) \leq 2 \hat{=} Suc k$   
**if**  $i \in P$  **for**  $i$   
**proof** –  
**have** *less-cardP*:  $iP i < k$   
**using** *iP bij-betwE* **that** **by** *blast*  
**have** *card-cr*:  $\text{card } (QS i) \leq 2 \hat{=} Suc k$   
**proof** –  
**have**  $\text{card } (QS i) \leq \text{prod card } (PP i)$   
**by** (*simp add: QS-def card-common-refinement finPP inPP-fin*)  
**also have**  $\dots = \text{prod card } (XP i \cup YP i)$   
**using** *finPP* **by** (*simp add: PP-def prod.insert-if*)  
**also have**  $\dots \leq 2 \hat{=} Suc k$   
**proof** (*rule prod-le-power*)  
**define** *XS* **where**  $XS \equiv (\bigcup R \in \{R \in P. iP R \leq iP i\}. \{\text{part2 } (X0 i R) i\})$   
**define** *YS* **where**  $YS \equiv (\bigcup R \in \{R \in P. iP R \geq iP i\}. \{\text{part2 } (Y0 R i) i\})$   
**have** *1*:  $\{R \in P. iP R \leq iP i\} \subseteq iP - \{..iP i\} \cap P$   
**by** *auto*

**have**  $\text{card } XS \leq \text{card } \{R \in P. iP R \leq iP i\}$   
**by** (*force simp add: XS-def ⟨finite P⟩ intro: order-trans [OF card-UN-le]*)  
**also have**  $\dots \leq \text{card } (iP - \{..iP i\} \cap P)$   
**using 1 by** (*simp add: ⟨finite P⟩ card-mono*)  
**also have**  $\dots \leq \text{Suc } (iP i)$   
**by** (*metis card-vimage-inj-on-le bij-betw-def card-atMost finite-atMost iP*)  
**finally have**  $cXS: \text{card } XS \leq \text{Suc } (iP i)$  .  
**have 2:**  $\{R \in P. iP R \geq iP i\} \subseteq iP - \{iP i..<k\} \cap P$   
**by** (*clarsimp (meson bij-betw-apply iP lessThan-iff nat-less-le)*)  
**have**  $\text{card } YS \leq \text{card } \{R \in P. iP R \geq iP i\}$   
**by** (*force simp add: YS-def ⟨finite P⟩ intro: order-trans [OF card-UN-le]*)  
**also have**  $\dots \leq \text{card } (iP - \{iP i..<k\} \cap P)$   
**using 2 by** (*simp add: ⟨finite P⟩ card-mono*)  
**also have**  $\dots \leq \text{card } \{iP i..<k\}$   
**by** (*meson bij-betw-def card-vimage-inj-on-le finite-atLeastLessThan iP*)  
**finally have**  $\text{card } YS \leq k - iP i$   
**by** *simp*  
**with less-cardP cXS have**  $k': \text{card } XS + \text{card } YS \leq \text{Suc } k$   
**by** *linarith*  
**have**  $\text{finXYS}: \text{finite } (XS \cup YS)$   
**unfolding XS-def YS-def using**  $\langle \text{finite } P \rangle$  **by** (*auto intro: finite-vimageI*)

**have**  $XP i \cup YP i \subseteq XS \cup YS$   
**apply** (*simp add: XP-def X-def YP-def Y-def XS-def YS-def irregular-set-def image-def subset-iff*)  
**by** (*metis insert-iff linear not-le*)  
**then have**  $\text{card } (XP i \cup YP i) \leq \text{card } XS + \text{card } YS$   
**by** (*meson card-Un-le card-mono finXYS order-trans*)  
**then show**  $\text{card } (XP i \cup YP i) \leq \text{Suc } k$   
**using**  $k'$  *le-trans* **by** *blast*  
**fix**  $x$   
**assume**  $x \in XP i \cup YP i$   
**then show**  $0 \leq \text{card } x \wedge \text{card } x \leq 2$   
**using**  $XP\text{-def } YP\text{-def card-part2}$  **by** *force*  
**qed** *auto*  
**finally show** *?thesis* .  
**qed**

**have**  $i' = i$  **if**  $q \subseteq i$  **if**  $i' \in P$   $q \in QS$   $i'$  **for**  $i' q$   
**by** (*metis QS-ne QS-subset-P ⟨i ∈ P⟩ disjnt-iff equalsOI pairwiseD part-GP partition-on-def subset-eq that*)  
**then have**  $QinP i \subseteq QS i$   
**by** (*auto simp: QinP-def Q-def*)  
**then have**  $\text{card } (QinP i) \leq \text{card } (QS i)$   
**by** (*simp add: card-mono that*)  
**also have**  $\dots \leq 2 \wedge \text{Suc } k$   
**using**  $QS\text{-def card-cr}$  **by** *presburger*  
**finally show** *?thesis* .  
**qed**

**have**  $\text{card } Q \leq \text{card } (\bigcup i \in P. QinP i)$

```

unfolding  $Q\text{-def}$ 
proof ( $rule\ card\ mono$ )
  show  $(\bigcup (QS \text{ ' } P)) \subseteq (\bigcup_{i \in P}. QinP\ i)$ 
    using  $ref\ QP\ QS\ subset\ P\ Q\ def\ QinP\ def$  by  $blast$ 
  show  $finite\ (\bigcup_{i \in P}. QinP\ i)$ 
    by ( $simp\ add: Q\ def\ QinP\ def\ \langle finite\ P \rangle$ )
qed
also have  $\dots \leq (\sum_{i \in P}. 2 \wedge Suc\ k)$ 
  by ( $smt\ (verit)\ \langle finite\ P \rangle\ card\ QP\ card\ UN\ le\ order\ trans\ sum\ mono$ )
finally show  $card\ Q \leq k * 2 \wedge Suc\ k$ 
  by ( $simp\ add: cardP$ )
qed
qed

```

## 1.7 The Regularity Proof Itself

We start with a trivial partition (one part). If it is already  $\epsilon$ -regular, we are done. If not, we refine it by applying lemma *exists-refinement* above, which increases the energy. We can repeat this step, but it cannot increase forever: by *mean-square-density-bounded* it cannot exceed 1. This defines an algorithm that must stop after at most  $\epsilon^{-5}$  steps, resulting in an  $\epsilon$ -regular partition.

**theorem** *Szemerédi-Regularity-Lemma*:

```

assumes  $\epsilon > 0$ 
obtains  $M$  where  $\bigwedge G. card\ (uverts\ G) > 0 \implies \exists P. \epsilon\text{-regular-partition}\ G\ P$ 
 $\wedge card\ P \leq M$ 
proof
  fix  $G$ 
  assume  $card\ (uverts\ G) > 0$ 
  then obtain  $finG: finite\ (uverts\ G)$  and  $nonempty: uverts\ G \neq \{\}$ 
    by ( $simp\ add: card\ gt\ 0\ iff$ )
  define  $\Phi$  where  $\Phi \equiv \lambda Q. refines\ (uverts\ G)\ Q\ P \wedge$ 
     $mean\ square\ density\ G\ Q \geq mean\ square\ density\ G\ P +$ 
 $\epsilon \wedge 5$ 
     $card\ Q \leq card\ P * 2 \wedge Suc\ (card\ P)$ 
  define  $next$  where  $next \equiv \lambda P. if\ \epsilon\text{-regular-partition}\ G\ P\ then\ P\ else\ SOME\ Q.$ 
 $\Phi\ Q\ P$ 
  define  $iter$  where  $iter \equiv \lambda i. (next \wedge i)\ \{uverts\ G\}$ 
  define  $last$  where  $last \equiv Suc\ (nat[1 / \epsilon \wedge 5])$ 
  have  $iter\ Suc\ [simp]: iter\ (Suc\ i) = next\ (iter\ i)$  for  $i$ 
    by ( $simp\ add: iter\ def$ )
  have  $\Phi: \Phi\ (next\ P)\ P$ 
  if  $Pk: partition\ on\ (uverts\ G)\ P$  and  $irreg: \neg \epsilon\text{-regular-partition}\ G\ P$  for  $P$ 
proof –
  have  $finite\ graph\ partition\ (uverts\ G)\ P\ (card\ P)$ 
    by ( $meson\ Pk\ finG\ finite\ elements\ finite\ graph\ partition\ def$ )
  then show ?thesis
    using  $that\ exists\ refinement\ [OF\ -\ finG\ irreg\ assms]\ irreg\ Pk$ 

```

```

    unfolding  $\Phi$ -def next-def by (smt (verit) someI)
qed
have partition-on: partition-on (uverts G) (iter i) for i
proof (induction i)
  case 0
  then show ?case
  by (simp add: iter-def nonempty trivial-graph-partition-exists partition-on-space)
next
  case (Suc i)
  with  $\Phi$  show ?case
  by (metis  $\Phi$ -def iter-Suc next-def refines-def)
qed
have False if irreg:  $\bigwedge i. i \leq \text{last} \implies \neg \varepsilon\text{-regular-partition } G \text{ (iter } i)$ 
proof -
  have  $\Phi$ -loop:  $\Phi$  (next (iter i)) (iter i) if  $i \leq \text{last}$  for i
  using  $\Phi$  irreg partition-on that by blast
  have iter-grow: mean-square-density G (iter i)  $\geq i * \varepsilon^5$  if  $i \leq \text{last}$  for i
  using that
  proof (induction i)
    case (Suc i)
    then show ?case
    by (clarsimp simp: algebra-simps) (smt (verit, best) Suc-leD  $\Phi$ -def  $\Phi$ -loop)
  qed (auto simp: iter-def)
  have last *  $\varepsilon^5 \leq \text{mean-square-density } G \text{ (iter last)}$ 
  by (simp add: iter-grow)
  also have ...  $\leq 1$ 
  by (meson finG finite-elements finite-graph-partition-def mean-square-density-bounded
partition-on)
  finally have real last *  $\varepsilon^5 \leq 1$  .
  with assms show False
  unfolding last-def by (meson lessI natceiling-lessD not-less pos-divide-less-eq
zero-less-power)
qed
then obtain i where  $i \leq \text{last}$  and  $\varepsilon\text{-regular-partition } G \text{ (iter } i)$ 
by force
then have reglar:  $\varepsilon\text{-regular-partition } G \text{ (iter } (i + d))$  for d
by (induction d) (auto simp add: next-def)
define tower where tower  $\equiv \lambda k. (\text{power}(2::\text{nat}) \text{ } k) 2$ 
have [simp]: tower (Suc k) =  $2 \wedge \text{tower } k$  for k
by (simp add: tower-def)
have iter-tower: card (iter i)  $\leq \text{tower } (2*i)$  for i
proof (induction i)
  case (Suc i)
  then have Qm: card (iter i)  $\leq \text{tower } (2 * i)$ 
  by simp
  then have *: card (next (iter i))  $\leq \text{card } (iter i) * 2 \wedge \text{Suc } (\text{card } (iter i))$ 
  using  $\Phi$  by (simp add:  $\Phi$ -def next-def partition-on)
  also have ...  $\leq 2 \wedge 2 \wedge \text{tower } (2 * i)$ 
  by (metis One-nat-def Suc.IH le-tower-2 lessI numeral-2-eq-2 order.trans

```

```

power-increasing-iff)
  finally show ?case
    by (simp add: Qm)
  qed (auto simp: iter-def tower-def)
  then show  $\exists P. \varepsilon\text{-regular-partition } G P \wedge \text{card } P \leq \text{tower}(2 * \text{last})$ 
    by (metis  $\langle i \leq \text{last} \rangle$  nat-le-iff-add reglar)
qed

```

The actual value of the bound is visible above: a tower of exponentials of height  $2(1 + \epsilon^{-5})$ .

end