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

March 17, 2025

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[1] and Yufei Zhao[2]. Similar material is found in many textbooks [1].

Contents

Acknowledgements

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

[1] https://www.dpmms.cam.ac.uk/~par31/notes/tic.pdf
[2] 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*] ⇒ *bool*
where *finite-graph-partition V P n* ≡ *partition-on V P* ∧ *finite P* ∧ *card P* = *n*

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

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

lemma *finite-graph-partition-equals*:
finite-graph-partition V P n ⟹ (⋃*P*) = *V*
by (*meson finite-graph-partition-def partition-on-def*)

lemma *finite-graph-partition-subset*:
⟦*finite-graph-partition V P n*; *X* ∈ *P*⟧ ⟹ *X* ⊆ *V*
using *finite-graph-partition-equals* **by** *blast*

lemma *trivial-graph-partition-exists*:
assumes *V* ≠ {}
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* ∈ *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 *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*. *card X*) = *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* :: *nat set* $\Rightarrow$ *nat set* $\Rightarrow$ *nat set* $\times$ *nat set set* $\Rightarrow$ (*nat* $\times$ *nat*) *set*
where *all-edges-between X Y G* $\equiv$ {(*x*,*y*). *x*$\in$*X* $\wedge$ *y*$\in$*Y* $\wedge$ {*x*,*y*} $\in$ *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 *card* (*all-edges-between X Y G*) $\leq$ *card X* $*$ *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* = *all-edges-between X Z G* $\cup$ *all-edges-between Y Z G*

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

lemma *all-edges-between-Un2*:
all-edges-between X ($Y \cup Z$) *G* = *all-edges-between X Y G* $\cup$ *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* $\wedge$ *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*)

ϵ-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* ⇒ *uvert set* ⇒ *uvert set* ⇒ *ugraph* ⇒ *bool*
(‹-−*regular′-pair*› [*999*]*1000*)
where *ε−regular-pair X Y G* ≡
∀ *A B*. *A* ⊆ *X* ∧ *B* ⊆ *Y* ∧ (*card A* ≥ *ε* ∗ *card X*) ∧ (*card B* ≥ *ε* ∗ *card Y*) ⟶
|*edge-density A B G* − *edge-density X Y G*| ≤ *ε* **for** *ε*::*real*

lemma *regular-pair-commute*: *ε−regular-pair X Y G* ⟷ *ε−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* ∪ *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* = (∑ *X*∈*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.prems 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* (⋃ *P*)

by (*simp add*: *finite-graph-partition-equals insert*)
then have *sumXP*: *card U* = *card X* + *card* ($\bigcup$ *P*)
by (*metis UX card-finite-graph-partition finite-graph-partition-equals insert.hyps insert.prems sum.insert*)
have *FUX*: *finite* (*U* − *X*)
by (*simp add*: *insert.prems*)
have *XUP*: *X* ∪ ($\bigcup$ *P*) = *U*
using *finite-graph-partition-equals insert.prems*(*2*) **by** *auto*
then have *edge-density U W G* = *edge-density* (*X* ∪ $\bigcup$ *P*) *W G*
by *auto*
also have ... = (*edge-density X W G* ∗ *card X* + *edge-density* ($\bigcup$ *P*) *W G* ∗ *card* ($\bigcup$ *P*))
/ (*card X* + *card* ($\bigcup$ *P*))
proof (*rule edge-density-Un*)
show *disjnt X* ($\bigcup$ *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* ($\bigcup$ *P*))
/ *card U*
using *UX card-finite-graph-partition finite-graph-partition-equals insert.prems*(*1*) *insert.prems*(*2*) *sumXP* **by** *auto*
also have ... = ($\sum$ *Y* ∈ *insert X P*. *edge-density Y W G* ∗ *card Y*) / *card U*
using *UX insert.prems insert.hyps*
apply (*simp add*: *insert.IH* [*OF FUX UX*] *divide-simps algebra-simps finite-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*] ⇒ (*uvert set* × *uvert set*) *set*
(‹-−*irregular*′*-set*› [*999*]*1000*)
where *ε*−*irregular-set* ≡ λ*G P*. {(*R*,*S*)|*R S*. *R*∈*P* ∧ *S*∈*P* ∧ ¬ *ε*−*regular-pair R S G*}
for *ε*::*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*] ⇒ *bool*

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

lemma *irregular-set-subset*: *ε–irregular-set G P* $\subseteq$ *P* $\times$ *P*
by (*auto simp*: *irregular-set-def*)

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

lemma *finite-irregular-set* [*simp*]: *finite P* $\Longrightarrow$ *finite* (*ε–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*:: [*uvert set*, *uvert set*, *ugraph*] $\Rightarrow$ *real* **where**
energy-graph-subsets U W G $\equiv$
card U $*$ *card W* $*$ (*edge-density U W G*)^2 / (*card* (*uverts G*))^2

Definition for partitions

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

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

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

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

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

lemma *energy-graph-partitions-commute*:
energy-graph-partitions G W U = *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* ≡ *energy-graph-partitions G P P*

lemma *mean-square-density*:
  mean-square-density G U ≡
    $(\sum R{\in}U.\ \sum S{\in}U.\ card\ R * card\ S * (edge\text{-}density\ R\ S\ G)^2)\ /\ (card\ (uverts\ G))^2$
  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 $(\sum R{\in}P.\ \sum S{\in}P.\ real\ (card\ R * card\ S)) \le (real(card\ V))^2$
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** $Y \in insert\ X\ P$ **for** *Y*
      by (*meson finite-graph-partition-finite insert.prems that*)
    have *C*: $card\ Y \le card\ V$ **if** $Y \in insert\ X\ P$ **for** *Y*
      by (*meson card-mono finite-graph-partition-subset insert.prems that*)
    have *D* [*simp*]: $(\sum Y{\in}P.\ real\ (card\ Y)) = real\ (card\ V) - real\ (card\ X)$
      by (*smt* (*verit*) *card-finite-graph-partition insert.hyps insert.prems of-nat-sum sum.cong sum.insert*)
    have *disjnt X* $(\bigcup P)$
      using *insert.prems insert.hyps*
      by (*auto simp add*: *finite-graph-partition-def disjnt-iff pairwise-insert partition-on-def*)
    with *insert* **have** ∗: $(\sum R{\in}P.\ \sum S{\in}P.\ real\ (card\ R * card\ S)) \le (real\ (card\ (V - X)))^2$
      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.prems* **by** *blast*
    have $(\sum R \in insert\ X\ P.\ \sum S \in insert\ X\ P.\ real\ (card\ R * card\ S))$
      $= real\ (card\ X * card\ X) + 2 * (card\ V - card\ X) * card\ X$
        $+ (\sum R{\in}P.\ \sum S{\in}P.\ real\ (card\ R * card\ S))$
      using ‹$X \notin P$› ‹*finite P*›
      by (*simp add*: *C of-nat-diff sum.distrib algebra-simps flip*: *sum-distrib-right*)
    also have $\ldots \le real\ (card\ X * card\ X) + 2 * (card\ V - card\ X) * card\ X + (real\ (card\ (V - X)))^2$
      using ∗ **by** *linarith*
    also have $\ldots \le (real\ (card\ V))^2$

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* $\leq$ *1*
proof−
have $(\sum R{\in}P.\ \sum S{\in}P.\ real\ (card\ R * card\ S) * (edge\text{-}density\ R\ S\ G)^2)$
$\leq (\sum R{\in}P.\ \sum S{\in}P.\ real\ (card\ R * card\ S))$
by (*intro sum-mono mult-right-le-one-le*) (*auto simp*: *abs-square-le-1 edge-density-ge0 edge-density-le1*)
also have $\ldots \leq (real(card\ (uverts\ G)))^2$
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 $a :: 'a \Rightarrow real$
assumes $\bigwedge i.\ i \in I \Longrightarrow a\ i \geq 0$
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** $?L \leq ?R$)
proof −
have $?L = (\sum i{\in}I.\ sqrt\ (a\ i) * (sqrt\ (a\ i) * b\ i))^2$
by (*smt* (*verit*, *ccfv-SIG*) *assms mult.assoc real-sqrt-mult-self sum.cong*)
also have ... $\leq (\sum i{\in}I.\ (sqrt\ (a\ i))^2) * (\sum i{\in}I.\ (sqrt\ (a\ i) * b\ i)^2)$
by (*rule Cauchy-Schwarz-ineq-sum*)
also have ... $= ?R$
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*: *finite-graph-partition U P n*
shows $card\ U * (edge\text{-}density\ U\ W\ G)^2 \leq (\sum R{\in}P.\ card\ R * (edge\text{-}density\ R\ W\ G)^2)$
proof (*cases finite U*)
case *True*
have §: $(\sum R{\in}P.\ card\ R * edge\text{-}density\ R\ W\ G)^2$
$\leq (sum\ card\ P) * (\sum R{\in}P.\ card\ R * (edge\text{-}density\ R\ W\ G)^2)$
by (*simp add*: *sum-products-le*)
have $card\ U * (edge\text{-}density\ U\ W\ G)^2 = (\sum R{\in}P.\ card\ R * (edge\text{-}density\ U\ W$

$G)^2$)
  by (*metis* ‹*finite U*› *P sum-distrib-right card-finite-graph-partition of-nat-sum*)
 also have ... = *edge-density U W G* ∗ ($\sum$ *R*∈*P*. *edge-density U W G* ∗ *card R*)
  by (*simp add*: *sum-distrib-left power2-eq-square mult-ac*)
 also have ... = ($\sum$ *R*∈*P*. *edge-density R W G* ∗ *real* (*card R*)) ∗ *edge-density U W G*
 proof −
  have *edge-density U W G* ∗ ($\sum$ *R*∈*P*. *edge-density R W G* ∗ *card R*)
    = *edge-density U W G* ∗ (*edge-density U W G* ∗ ($\sum$ *R*∈*P*. *card R*))
   using ‹*finite U*› *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 ... = ($\sum$ *R*∈*P*. *card R* ∗ *edge-density R W G*) ∗ *edge-density U W G*
  by (*simp add*: *sum-distrib-left mult-ac*)
 also have ... = ($\sum$ *R*∈*P*. *card R* ∗ *edge-density R W G*)^2 / *card U*
  using *assms* **by** (*simp add*: *edge-density-partition* [*OF P*] *mult-ac flip*: *power2-eq-square*)
 also have ... ≤ ($\sum$ *R*∈*P*. *card R* ∗ (*edge-density R W G*)^2)
  using § *P card-finite-graph-partition* ‹*finite U*›
  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* ≥ *energy-graph-subsets U W G*
proof −
 have (*card U* ∗ *card W*) ∗ (*edge-density U W G*)^2 = *card W* ∗ (*card U* ∗ (*edge-density U W G*)^2)
  by (*simp add*: *mult-ac*)
 also have ... ≤ *card W* ∗ ($\sum$ *R*∈*P*. *card R* ∗ (*edge-density R W G*)^2)
  by (*intro mult-left-mono energy-graph-partition-half*) (*use assms* **in** *auto*)
 also have ... = ($\sum$ *R*∈*P*. *card R* ∗ (*card W* ∗ (*edge-density W R G*)^2))
  by (*simp add*: *sum-distrib-left edge-density-commute mult-ac*)
 also have ... ≤ ($\sum$ *R*∈*P*. *card R* ∗ ($\sum$ *S*∈*Q*. *card S* ∗ (*edge-density S R G*)^2))
  by (*intro mult-left-mono energy-graph-partition-half sum-mono*) (*use assms* **in** *auto*)
 also have ... ≤ ($\sum$ *R*∈*P*. $\sum$ *S*∈*Q*. (*card R* ∗ *card S*) ∗ (*edge-density R S G*)^2)
  by (*simp add*: *sum-distrib-left edge-density-commute mult-ac*)
 finally
 have (*card U* ∗ *card W*) ∗ (*edge-density U W G*)^2
  ≤ ($\sum$ *R*∈*P*. $\sum$ *S*∈*Q*. (*card R* ∗ *card S*) ∗ (*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: $\bigwedge R.\ R \in P \Longrightarrow$ *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 ‹finite V› 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* $(F\ `\ P)$
    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 :: *nat set* $\Rightarrow$ *real*
  proof –
    have $Q = (\bigcup R \in P.\ F\ R)$
      using *ref* **by** (*force simp add*: *refines-def dest*: *F*)
    then have $(\sum S{\in}Q.\ f\ S) =$ *sum f* $(\bigcup R \in P.\ F\ R)$
      by *blast*
    also have $\ldots =$ (*sum* $\circ$ *sum*) f $(F\ `\ P)$
     by (*smt* (*verit*, *best*) *dFP disjnt-def finite-F image-iff pairwiseD sum.Union-disjoint*)
    also have $\ldots = (\sum R \in P.\ \sum U{\in}F\ R.\ f\ U)$
      unfolding *comp-apply* **by** (*metis injF sum.reindex-cong*)
    finally show *?thesis*
      by *simp*
  qed
  have *?egP* $= (\sum R \in P.\ \sum T{\in}U.$ *energy-graph-subsets R T G*)
    by (*simp add*: *energy-graph-partitions-def*)
  also have $\ldots \leq (\sum R{\in}P.\ \sum T{\in}U.$ *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 ... = ($\sum R \in P. \sum D \in F\ R. \sum T{\in}U.$ *energy-graph-subsets D T G*)
by (*simp add*: *energy-graph-partitions-def sum.swap* [**where** $B = U$])
also have ... = *?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* $G\ Q\ Q' \geq$ *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* $G\ Q \geq$ *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 $\mathcal{U} \subseteq$ *uverts* G and $\mathcal{W} \subseteq$ *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*: *card* (*part2* $X\ Y$) ≤ 2
by (*simp add*: *part2-def card-insert-if*)

lemma *sum-part2*: $[\![X \subseteq Y; f\{\} = 0]\!] \Longrightarrow$ *sum* f (*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 ε::*real* **and** $U\ W\ G$
defines *alpha* $\equiv$ *edge-density* $U\ W\ G$
defines $u \equiv \lambda X\ Y.$ *edge-density* $X\ Y\ G$ − *alpha*

assumes *finite U finite W*
and $U' \subseteq U$ $W' \subseteq W$ $\varepsilon > 0$
and U': *card* $U' \geq \varepsilon *$ *card U* **and** W': *card* $W' \geq \varepsilon *$ *card W*
and *gt*: $|u\ U'\ W'| > \varepsilon$
shows ($\sum A \in$ *part2* U' U. $\sum B \in$ *part2* W' W. *energy-graph-subsets A B G*)
$\geq$ *energy-graph-subsets U W G* + ε^*4* $*$ (*card U* $*$ *card W*) / (*card* (*uverts* *G*))^2
(**is** *?lhs* $\geq$ *?rhs*)
proof −
define *UF* **where** $UF \equiv$ *part2* U' U
define *WF* **where** $WF \equiv$ *part2* W' W
obtain [*simp*]: *finite U finite W*
using *assms* **by** (*meson finite-subset*)
obtain *card'*: *card* $U' > 0$ *card* $W' > 0$
using *gt* ‹$\varepsilon > 0$› U' W'
by (*force simp*: *u-def alpha-def edge-density-def mult-le-0-iff zero-less-mult-iff*)
then obtain *card*: *card* $U > 0$ *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*: *card* $x \neq 0$ **if** $x \in UF$ **for** x
using *card' assms that* **by** (*auto simp*: *UF-def part2-def split*: *if-split-asm*)
have *WF-ne*: *card* $x \neq 0$ **if** $x \in WF$ **for** x
using *card' assms that* **by** (*auto simp*: *WF-def part2-def split*: *if-split-asm*)
have *cardUW*: *card* $U =$ *card* U' + *card*($U - U'$) *card* $W =$ *card* W' + *card*($W - W'$)
using *card card'* ‹$U' \subseteq U$› ‹$W' \subseteq W$›
by (*metis card-eq-0-iff card-Diff-subset card-mono le-add-diff-inverse less-le*)+
have $U = (U - U') \cup U'$ *disjnt* $(U - U')$ U'
using ‹$U' \subseteq U$› **by** (*force simp*: *disjnt-iff*)+
then have *CU*: *card* (*all-edges-between U Z G*)
= *card* (*all-edges-between* $(U - U')$ *Z G*) + *card* (*all-edges-between* U' *Z G*)
if *finite Z* **for** *Z*
by (*metis* ‹*finite* U'› *all-edges-between-Un1 all-edges-between-disjnt1* ‹*finite U*›
card-Un-disjnt finite-Diff finite-all-edges-between that)

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

have $*$: $(\sum X\in UF.\ \sum Y\in WF.\ real\ (card\ (all\text{-}edges\text{-}between\ X\ Y\ G)))$
$= card\ (all\text{-}edges\text{-}between\ U\ W\ G)$
by $(simp\ add{:}\ UF\text{-}def\ WF\text{-}def\ cardUW\ CU\ CW\ sum\text{-}part2\ \langle U' \subseteq U\rangle\ \langle W' \subseteq W\rangle)$
have $**$: $real\ (card\ U) * real\ (card\ W) = (\sum X\in UF.\ \sum Y\in WF.\ card\ X * card\ Y)$
by $(simp\ add{:}\ UF\text{-}def\ WF\text{-}def\ cardUW\ sum\text{-}part2\ \langle U' \subseteq U\rangle\ \langle W' \subseteq W\rangle\ algebra\text{-}simps)$

let $?S = \sum X\in UF.\ \sum Y\in WF.\ (card\ X * card\ Y)\ /\ (card\ U * card\ W) * (edge\text{-}density\ X\ Y\ G)^2$
define T **where** $T \equiv (\sum X\in UF.\ \sum Y\in WF.\ (card\ X * card\ Y)\ /\ (card\ U * card\ W) * (edge\text{-}density\ X\ Y\ G))$
have §: $2 * T = alpha + alpha * (\sum X\in UF.\ \sum Y\in WF.\ (card\ X * card\ Y)\ /\ (card\ U * card\ W))$
unfolding $alpha\text{-}def\ T\text{-}def$
by $(simp\ add{:}\ *\ **\ edge\text{-}density\text{-}def\ divide\text{-}simps\ sum\text{-}part2\ \langle U' \subseteq U\rangle\ \langle W' \subseteq W\rangle\ UF\text{-}ne\ WF\text{-}ne\ flip{:}\ sum\text{-}divide\text{-}distrib)$
have $\varepsilon * \varepsilon \le u\ U'\ W' * u\ U'\ W'$
by $(metis\ abs\text{-}ge\text{-}zero\ abs\text{-}mult\text{-}self\text{-}eq\ \langle \varepsilon > 0\rangle\ gt\ less\text{-}le\ mult\text{-}mono)$
then have $(\varepsilon*\varepsilon)*(\varepsilon*\varepsilon) \le (card\ U' * card\ W')\ /\ (card\ U * card\ W) * (u\ U'\ W')^2$
using $card\ mult\text{-}mono\ [OF\ U'\ W']\ \ \langle \varepsilon > 0\rangle$
apply $(simp\ add{:}\ divide\text{-}simps\ eval\text{-}nat\text{-}numeral)$
by $(smt\ (verit,\ del\text{-}insts)\ mult.assoc\ mult.commute\ mult\text{-}mono'\ of\text{-}nat\text{-}0\text{-}le\text{-}iff\ zero\text{-}le\text{-}mult\text{-}iff)$
also have $\ldots \le (\sum X\in UF.\ \sum Y\in WF.\ \ (card\ X * card\ Y)\ /\ (card\ U * card\ W) * (u\ X\ Y)^2)$
by $(simp\ add{:}\ UF\text{-}def\ WF\text{-}def\ sum\text{-}part2\ \langle U' \subseteq U\rangle\ \langle W' \subseteq W\rangle)$
also have $\ldots = ?S - 2 * T * alpha$
$+\ alpha^2 * (\sum X\in UF.\ \sum Y\in WF.\ (card\ X * card\ Y)\ /\ (card\ U * card\ W))$
by $(simp\ add{:}\ u\text{-}def\ T\text{-}def\ power2\text{-}diff\ mult\text{-}ac\ ring\text{-}distribs\ divide\text{-}simps$
$sum\text{-}distrib\text{-}left\ sum\text{-}distrib\text{-}right\ sum\text{-}subtractf\ sum.distrib\ flip{:}\ sum\text{-}divide\text{-}distrib)$
also have $\ldots = ?S - alpha^2$
using § **by** $(simp\ add{:}\ power2\text{-}eq\text{-}square\ algebra\text{-}simps)$
finally have 12: $alpha^2 + \varepsilon\hat{\ }4 \le ?S$
by $(simp\ add{:}\ eval\text{-}nat\text{-}numeral)$
have $?rhs = (alpha^2 + \varepsilon\hat{\ }4) * (card\ U * card\ W\ /\ (card\ (uverts\ G))^2)$
unfolding $alpha\text{-}def\ energy\text{-}graph\text{-}subsets\text{-}def$
by $(simp\ add{:}\ ring\text{-}distribs\ divide\text{-}simps\ power2\text{-}eq\text{-}square)$
also have $\ldots \le ?S * (card\ U * card\ W\ /\ (card\ (uverts\ G))^2)$
by $(rule\ mult\text{-}right\text{-}mono\ [OF\ 12])\ auto$
also have $\ldots = ?lhs$
using $card$ **unfolding** $energy\text{-}graph\text{-}subsets\text{-}def\ UF\text{-}def\ WF\text{-}def$
by $(auto\ simp\ add{:}\ algebra\text{-}simps\ sum\text{-}part2\ \langle U' \subseteq U\rangle\ \langle W' \subseteq W\rangle\ )$
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*: $k * (2 \ \hat{} \ Suc\ k) \le 2\hat{}(2\hat{}k)$
proof (*induction k rule*: *less-induct*)
  case (*less k*)
  show *?case*
  proof (*cases* $k \le Suc\ (Suc\ 0)$)
    case *False*
    define *j* **where** $j = k - Suc\ 0$
    have *kj*: $k = Suc\ j$
      using *False j-def* **by** *force*
    with *False* **have** §: $(2\hat{}j + 3) \le (2::nat) \ \hat{} \ k$
      by (*simp add*: *Suc-leI le-less-trans not-less-eq-eq numeral-3-eq-3*)
    have $k * (2 \ \hat{} \ Suc\ k) \le 6 * j * 2\hat{}j$
      using *False* **by** (*simp add*: *kj*)
    also have $\ldots \le 6 * 2\hat{}(2\hat{}j)$
      using *kj less.IH* **by** *force*
    also have $\ldots < 2\hat{}(2\hat{}j + 3)$
      by (*simp add*: *power-add*)
    also have $\ldots \le 2\hat{}2\hat{}k$
      by (*simp add*: §)
    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 *fgp*: *finite-graph-partition* (*uverts G*) *P k* **and** *finite* (*uverts G*)
    and *irreg*: $\neg$ ε−*regular-partition G P* **and** $\varepsilon > 0$
  obtains *Q* **where** *refines* (*uverts G*) *Q P*
      mean-square-density G Q $\ge$ *mean-square-density G P* + $\varepsilon\hat{}5$
      $\bigwedge R.\ R \in P \Longrightarrow card\ \{S \in Q.\ S \subseteq R\} \le 2 \ \hat{} \ Suc\ k$
      $card\ Q \le k * 2 \ \hat{} \ Suc\ k$
proof −
  define *sum-pp* **where** *sum-pp* $\equiv$ ($\sum$(*R*,*S*) $\in$ ε−*irregular-set G P*. *card R* $*$ *card S*)
  have *cardP*: *card P* = *k*
    using *fgp finite-graph-partition-def* **by** *force*
  then have $k \ne 0$
   using *assms* **unfolding** *regular-partition-def irregular-set-def finite-graph-partition-def*

by *fastforce*
with *assms* **have** *G-nonempty*: $0 < card\ (uverts\ G)$
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*: $sum\text{-}pp > \varepsilon * (card\ (uverts\ G))^2$
by (*metis irreg not-le part-GP regular-partition-def sum-pp-def*)
then have *sum-irreg-pos*: $sum\text{-}pp > 0$
using ‹$\varepsilon > 0$› *G-nonempty less-asym* **by** *fastforce*
have $\exists X{\subseteq}R.\ \exists Y{\subseteq}S.\ \varepsilon * card\ R \leq card\ X \wedge \varepsilon * card\ S \leq card\ Y \wedge$
$|edge\text{-}density\ X\ Y\ G - edge\text{-}density\ R\ S\ G| > \varepsilon$
if $(R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P$ **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*: $\bigwedge R\ S.\ [\![(R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P]\!] \Longrightarrow X0\ R\ S \subseteq R \wedge Y0\ R\ S \subseteq S$
and *XY0-eps*:
$\bigwedge R\ S.\ (R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P$
$\Longrightarrow \varepsilon * card\ R \leq card\ (X0\ R\ S) \wedge \varepsilon * card\ S \leq card\ (Y0\ R\ S) \wedge$
$|edge\text{-}density\ (X0\ R\ S)\ (Y0\ R\ S)\ G - edge\text{-}density\ R\ S\ G| > \varepsilon$
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** $X \equiv \lambda R\ S.\ if\ iP\ R < iP\ S\ then\ Y0\ S\ R\ else\ X0\ R\ S$
define *Y* **where** $Y \equiv \lambda R\ S.\ if\ iP\ R < iP\ S\ then\ X0\ S\ R\ else\ Y0\ R\ S$
have *XY-psub-P*: $\bigwedge R\ S.\ [\![(R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P]\!] \Longrightarrow X\ R\ S \subseteq R \wedge Y\ R\ S \subseteq S$
using *XY0-psub-P* **by** (*force simp*: *X-def Y-def irregular-set-swap*)
have *XY-eps*:
$\bigwedge R\ S.\ (R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P$
$\Longrightarrow \varepsilon * card\ R \leq card\ (X\ R\ S) \wedge \varepsilon * card\ S \leq card\ (Y\ R\ S) \wedge$
$|edge\text{-}density\ (X\ R\ S)\ (Y\ R\ S)\ G - edge\text{-}density\ R\ S\ G| > \varepsilon$
using *XY0-eps* **by** (*force simp*: *X-def Y-def edge-density-commute irregular-set-swap*)
have *card-elem-P*: $card\ R > 0$ **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** $(R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P$ **for** *R S*
using *XY-eps* [*OF that*] *that* ‹$\varepsilon > 0$› *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** $XP\ R \equiv ((\lambda S.\ part2\ (X\ R\ S)\ R)$ ‘ $\{S.\ (R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P\})$ **for** *R*

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

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

define *PP* **where** $PP \equiv \lambda R.\ insert\ \{R\}\ (XP\ R \cup YP\ R)$
define *QS* **where** $QS\ R \equiv common\text{-}refinement\ (PP\ R)$ **for** *R*
define *r* **where** $r\ R \equiv card\ (QS\ R)$ **for** *R*
have *finite P*
  using *fgp 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*: $P \in PP\ R \Longrightarrow finite\ P$ **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** $R \in P$ **for** *R*
  unfolding *QS-def*
proof (*intro partition-on-common-refinement partition-onI*)
  show $\bigwedge\mathcal{A}.\ \mathcal{A} \in PP\ R \Longrightarrow \{\} \notin \mathcal{A}$
    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** $R{\in}P$ **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** $R{\in}P$ **for** *R*
  using *QS-def finite-QS* **by** *force*
have *QS-ne*: $\{\} \notin QS\ R$ **if** $R{\in}P$ **for** *R*
  using *QS-def part-QS partition-onD3 that* **by** *blast*
have *QS-subset-P*: $q \in QS\ R \Longrightarrow q \subseteq R$ **if** $R{\in}P$ **for** *R q*
  by (*meson finite-graph-partition-subset part-P-QS that*)
then have *QS-inject*: $R = R'$
  if $R{\in}P\ R'{\in}P\ q \in QS\ R\ q \in QS\ R'$ **for** $R\ R'\ q$
  by (*metis UnionI disjnt-iff equals0I 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 $\bigcup Q = uverts\ G$
proof
show $\bigcup Q \subseteq uverts\ G$
using *QS-subset-P Q-def fgp finite-graph-partition-equals* **by** *fastforce*
show $uverts\ G \subseteq \bigcup Q$
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* ‹$R \in P$› ∗ *pairwiseD partition-on-def* ‹$p \neq q$›)
next
case *False*
with ∗ **show** *?thesis*
by (*metis QS-subset-P* ‹$R \in P$› ‹$S \in P$› *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 $?REG = ?PP - \varepsilon{-}irregular\text{-}set\ G\ P$
define *sum-eps* **where** $sum\text{-}eps \equiv (\sum (R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P.\ \varepsilon\hat{\ }4 * (card\ R * card\ S) / (card\ (uverts\ G))^2)$
have *A*: $energy\text{-}graph\text{-}subsets\ R\ S\ G + \varepsilon\hat{\ }4 * (card\ R * card\ S) / (card\ (uverts\ G))^2$
$\leq energy\text{-}graph\text{-}partitions\ G\ (part2\ (X\ R\ S)\ R)\ (part2\ (Y\ R\ S)\ S)$
(**is** $?L \leq ?R$)
if ∗: $(R,S) \in \varepsilon{-}irregular\text{-}set\ G\ P$ **for** *R S*
proof −
have $R{\in}P$ $S{\in}P$
using ∗ **by** (*auto simp*: *irregular-set-def*)
have $?L \leq (\sum A \in part2\ (X\ R\ S)\ R.\ \sum B \in part2\ (Y\ R\ S)\ S.\ energy\text{-}graph\text{-}subsets\ A\ B\ G)$

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 ... = ($\sum$ (*i*,*j*) ∈ *?REG*. *energy-graph-subsets i j G*)

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

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

have *card XS* $\leq$ *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 ... $\leq$ *card* (iP −‘ $\{..iP\ i\} \cap P$)
using *1* **by** (*simp add*: ‹*finite P*› *card-mono*)
also have ... $\leq$ *Suc* (*iP i*)
by (*metis card-vimage-inj-on-le bij-betw-def card-atMost finite-atMost iP*)
finally have *cXS*: *card XS* $\leq$ *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 *card YS* $\leq$ *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 ... $\leq$ *card* (iP −‘ $\{iP\ i..<k\} \cap P$)
using *2* **by** (*simp add*: ‹*finite P*› *card-mono*)
also have ... $\leq$ *card* $\{iP\ i..<k\}$
by (*meson bij-betw-def card-vimage-inj-on-le finite-atLeastLessThan iP*)
finally have *card YS* $\leq k - iP\ i$
by *simp*
with *less-cardP cXS* **have** k': *card XS* + *card YS* $\leq$ *Suc k*
by *linarith*
have *finXYS*: *finite* (*XS* ∪ *YS*)
unfolding *XS-def YS-def* **using** ‹*finite P*› **by** (*auto intro*: *finite-vimageI*)

have *XP i* ∪ *YP i* ⊆ *XS* ∪ *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 *card* (*XP i* ∪ *YP i*) $\leq$ *card XS* + *card YS*
by (*meson card-Un-le card-mono finXYS order-trans*)
then show *card* (*XP i* ∪ *YP i*) $\leq$ *Suc k*
using k' *le-trans* **by** *blast*
fix *x*
assume $x \in$ *XP i* ∪ *YP i*
then show $0 \leq$ *card x* ∧ *card x* ≤ 2
using *XP-def YP-def card-part2* **by** *force*
qed *auto*
finally show *?thesis* **.**
qed
have $i' = i$ **if** $q \subseteq i$ $i' \in P$ $q \in QS\ i'$ **for** i' q
by (*metis QS-ne QS-subset-P* ‹$i \in P$› *disjnt-iff equals0I pairwiseD part-GP partition-on-def subset-eq that*)
then have *QinP i* ⊆ *QS i*
by (*auto simp*: *QinP-def Q-def*)
then have *card* (*QinP i*) $\leq$ *card* (*QS i*)
by (*simp add*: *card-mono that*)
also have ... ≤ 2 ^ *Suc k*
using *QS-def card-cr* **by** *presburger*
finally show *?thesis* **.**
qed
have *card Q* $\leq$ *card* ($\bigcup i \in P.\ QinP\ i$)

unfolding *Q-def*
proof (*rule card-mono*)
show ($\bigcup$ (*QS* ‘ *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* ‹*finite P*›)
qed
also have ... $\leq$ ($\sum i{\in}P.\ 2\ \hat{}\ Suc\ k$)
by (*smt* (*verit*) ‹*finite P*› *card-QP card-UN-le order-trans sum-mono*)
finally show *card Q* $\leq$ $k * 2\ \hat{}\ Suc\ k$
by (*simp add*: *cardP*)
qed
qed

1.7 The Regularity Proof Itself

We start with a trivial partition (one part). If it is already ϵ-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 ϵ^{-5} steps, resulting in an ϵ-regular partition.

theorem *Szemeredi-Regularity-Lemma*:
assumes $\varepsilon > 0$
obtains *M* **where** $\bigwedge G.$ *card* (*uverts G*) $> 0 \Longrightarrow \exists P.$ ε*−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 Φ **where** $\Phi \equiv \lambda Q\ P.$ *refines* (*uverts G*) *Q P* $\wedge$
mean-square-density G Q $\geq$ *mean-square-density G P* +
$\varepsilon\hat{}5$ $\wedge$
card Q $\leq$ *card P* $* 2\ \hat{}\ Suc$ (*card P*)
define *nxt* **where** *nxt* $\equiv \lambda P.$ *if* ε*−regular-partition G P then P else SOME Q.*
Φ *Q P*
define *iter* **where** *iter* $\equiv \lambda i.$ (*nxt* $\hat{}\hat{}$ *i*) {*uverts G*}
define *last* **where** *last* $\equiv$ *Suc* (*nat*$\lceil 1\ /\ \varepsilon\ \hat{}\ 5\rceil$)
have *iter-Suc* [*simp*]: *iter* (*Suc i*) = *nxt* (*iter i*) **for** *i*
by (*simp add*: *iter-def*)
have Φ: Φ (*nxt P*) *P*
if *Pk*: *partition-on* (*uverts G*) *P* **and** *irreg*: $\neg$ ε*−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 Φ-def nxt-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 Φ show ?case
      by (metis Φ-def iter-Suc nxt-def refines-def)
  qed
  have False if irreg: ⋀i. i≤last ⟹ ¬ ε−regular-partition G (iter i)
  proof −
    have Φ-loop: Φ (nxt (iter i)) (iter i) if i≤last for i
      using Φ irreg partition-on that by blast
    have iter-grow: mean-square-density G (iter i) ≥ i ∗ ε^5 if i≤last for i
      using that
    proof (induction i)
      case (Suc i)
      then show ?case
        by (clarsimp simp: algebra-simps) (smt (verit, best) Suc-leD Φ-def Φ-loop)
    qed (auto simp: iter-def)
    have last ∗ ε^5 ≤ mean-square-density G (iter last)
      by (simp add: iter-grow)
    also have . . . ≤ 1
     by (meson finG finite-elements finite-graph-partition-def mean-square-density-bounded
partition-on)
    finally have real last ∗ ε ^ 5 ≤ 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 ≤ last and ε−regular-partition G (iter i)
    by force
  then have reglar: ε−regular-partition G (iter (i + d)) for d
    by (induction d) (auto simp add: nxt-def)
  define tower where tower ≡ λk. (power(2::nat) ^^ k) 2
  have [simp]: tower (Suc k) = 2 ^ tower k for k
    by (simp add: tower-def)
  have iter-tower: card (iter i) ≤ tower (2∗i) for i
  proof (induction i)
    case (Suc i)
    then have Qm: card (iter i) ≤ tower (2 ∗ i)
      by simp
    then have ∗: card (nxt (iter i)) ≤ card (iter i) ∗ 2 ^ Suc (card (iter i))
      using Φ by (simp add: Φ-def nxt-def partition-on)
    also have . . . ≤ 2 ^ 2 ^ 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 $\exists P.\ \varepsilon{-}regular\text{-}partition\ G\ P \wedge card\ P \leq tower(2 * last)$
  by (*metis* ‹$i \leq last$› *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