Abstract
Random graphs are graphs with a fixed number of vertices, where each edge is present with a fixed probability. We are interested in the probability that a random graph contains a certain pattern, for example a cycle or a clique. A very high edge probability gives rise to perhaps too many edges (which degrades performance for many algorithms), whereas a low edge probability might result in a disconnected graph. We prove a theorem about a threshold probability such that a higher edge probability will asymptotically almost surely produce a random graph with the desired subgraph.

Contents
1 Introduction 2
2 Miscellaneous and contributed lemmas 2
3 Lemmas about probabilities 11
  3.1 Indicator variables and valid probability values . . . . . . . . 11
  3.2 Expectation and variance . . . . . . . . . . . . . . . . . . . . . 12
  3.3 Sets of indicator variables . . . . . . . . . . . . . . . . . . . . . 14
4 Lemmas about undirected graphs 18
  4.1 Subgraphs . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
  4.2 Induced subgraphs . . . . . . . . . . . . . . . . . . . . . . . . . 21
  4.3 Graph isomorphism . . . . . . . . . . . . . . . . . . . . . . . . 22
  4.4 Isomorphic subgraphs . . . . . . . . . . . . . . . . . . . . . . 26
  4.5 Density . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 28
  4.6 Fixed selectors . . . . . . . . . . . . . . . . . . . . . . . . . . . 31
5 Classes and properties of graphs 32
6 The subgraph threshold theorem 33
1 Introduction

Random graphs have been introduced by Erdős and Rényi in [?]. They describe a probability space where, for a fixed number of vertices, each possible edge is present with a certain probability independent from other edges, but with the same probability for each edge. They study what properties emerge when increasing the number of vertices, or as they call it, “the evolution of such a random graph”. The theorem which we will prove here is a slightly different version from that in the first section of that paper.

Here, we are interested in the probability that a random graph contains a certain pattern, for example a cycle or a clique. A very high edge probability gives rise to perhaps too many edges, which is usually undesired since it degrades the performance of many algorithms, whereas a low edge probability might result in a disconnected graph. The central theorem determines a threshold probability such that a higher edge probability will asymptotically almost surely produce a random graph with the desired subgraph.

The proof is outlined in [?, § 11.4] and [?, § 3]. The work is based on the comprehensive formalization of probability theory in Isabelle/HOL and on a previous definition of graphs in a work by Noschinski [?]. There, Noschinski formalized the proof that graphs with arbitrarily large girth and chromatic number exist. While the proof in this paper uses a different approach, the definition of a probability space on edges turned out to be quite useful.

2 Miscellaneous and contributed lemmas

theory Ugraph-Misc
imports Limits
~~/src/HOL/Probability/Probability
../Girth-Chromatic/Girth-Chromatic-Misc
begin

lemma setsum-square:
  fixes a :: 'i ⇒ 'a :: {monoid-mult, semiring-0}
  shows ((∑ i ∈ I. a i) ^ 2) = (∑ i ∈ I. ∑ j ∈ I. a i * a j)
  by (simp only: setsum-product power2-eq-square)

lemma setsum-split:
  assumes finite I
  shows (∑ i | i ∈ I ∧ f i else g i) = (∑ i | i ∈ I ∧ p i. f i) + (∑ i | i ∈ I ∧ ¬p i. g i)
  by (simp add: setsum.If-cases Int-def)

lemma setsum-split2:
  assumes finite I
  shows (∑ i | i ∈ I ∧ P i. if Q i then f i else g i) = (∑ i | i ∈ I ∧ P i ∧ Q i. f
proof (subst setsum.If-cases)
  show finite \{i \in I. P i\} using assms by simp

  have \{i \in I. P i\} \cap Collect Q = \{i \in I. P i \land Q i\} \{i \in I. P i\} \cap - Collect Q = \{i \in I. P i \land \neg Q i\}
    by auto
  thus setsum f (\{i \in I. P i\} \cap Collect Q) + setsum g (\{i \in I. P i\} \cap - Collect Q)
    = setsum f \{i \in I. P i \land Q i\} + setsum g \{i \in I. P i \land \neg Q i\}
    by presburger

qed

lemma setsum-upper:
  fixes f :: 'i => 'a :: \{ordered-cancel-ab-semigroup-add, comm-monoid-add\}
  assumes finite I \land i. i \in I \implies 0 \leq f i
  shows (\sum i \mid i \in I \land P i, f i) \leq setsum f I
proof -
  have setsum f I = (\sum i \in I. if P i then f i else f i)
    by simp
  hence setsum f I = (\sum i \mid i \in I \land P i, f i) + (\sum i \mid i \in I \land \neg P i, f i)
    by (simp only: setsum-split[OF (finite I)])
  moreover have 0 \leq (\sum i \mid i \in I \land \neg P i, f i)
    by (rule setsum-nonneg) (simp add: assms)
  ultimately show ?thesis
    by (metis (full-types) add.comm_neutral add-left_mono)
qed

lemma setsum-lower:
  fixes f :: 'i => 'a :: \{ordered-cancel-ab-semigroup-add, comm-monoid-add\}
  assumes finite I \land i. i \in I \implies 0 \leq f i x < f i
  shows x < setsum f I
proof -
  have 0 + x < setsum f (I \{-i\}) + f i
    using assms by (intro add-le-less-mono setsum-nonneg) auto
  also have setsum f (I \{-i\}) + f i = setsum f I
    using assms by (simp add: setsum.remove_ac_simps)
  finally show ?thesis by simp
qed

lemma setsum-lower-or-eq:
  fixes f :: 'i => 'a :: \{ordered-cancel-ab-semigroup-add, comm-monoid-add\}
  assumes finite I \land i. i \in I \implies 0 \leq f i x \leq f i
  shows x \leq setsum f I
proof -
  have 0 + x \leq setsum f (I \{-i\}) + f i
    using assms by (metis Diff_iff add-mono setsum-nonneg)
  also have setsum f (I \{-i\}) + f i = setsum f I
    using assms by (simp add: setsum.remove_ac_simps)
\begin{align*}
\text{finally show } \& \text{thesis by simp} \\
\text{qed}
\end{align*}

\textbf{lemma} setsum-left-div-distrib:
\begin{align*}
\text{fixes } f : : \i \Rightarrow \text{real} \\
\text{shows } (\sum i \in I. f i / x) &= \text{setsum } f I / x \\
\text{proof} - \\
\text{have } (\sum i \in I. f i / x) &= (\sum i \in I. f i * (1 / x)) \\
\text{by simp} \\
\text{also have } \ldots &= \text{setsum } f I * (1 / x) \\
\text{by } \text{(rule setsum-left-distrib[symmetric])} \\
\text{also have } \ldots &= \text{setsum } f I / x \\
\text{by simp} \\
\text{finally show } \& \text{thesis} \\
\text{qed}
\end{align*}

\textbf{lemma} powr-mono3:
\begin{align*}
\text{fixes } x : : \text{real} \\
\text{assumes } 0 < x x < 1 b \leq a \\
\text{shows } x \text{ powr } a &\leq x \text{ powr } b \\
\text{proof} - \\
\text{have } x \text{ powr } a &= 1 / x \text{ powr } -a \\
\text{by } \text{(simp add: powr-minus-divide)} \\
\text{also have } \ldots &= (1 / x) \text{ powr } -a \\
\text{using } \text{assms} \text{ by } \text{(simp add: powr-divide)} \\
\text{also have } \ldots &\leq (1 / x) \text{ powr } -b \\
\text{using } \text{assms} \text{ by } \text{(simp add: powr-mono)} \\
\text{also have } \ldots &= 1 / x \text{ powr } -b \\
\text{using } \text{assms} \text{ by } \text{(simp add: powr-divide)} \\
\text{also have } \ldots &= x \text{ powr } b \\
\text{by } \text{(simp add: powr-minus-divide)} \\
\text{finally show } \& \text{thesis} \\
\text{qed}
\end{align*}

\textbf{lemma} card-union:
\begin{align*}
\text{finite } A \Rightarrow \text{finite } B \Rightarrow \text{card } (A \cup B) &= \text{card } A + \text{card } B - \text{card } (A \cap B) \\
\text{by } \text{(metis card-Un-Int[symmetric] diff-add-inverse2)} \\
\text{lemma} \text{ card-1-element:} \\
\text{assumes } \text{card } E = 1 \\
\text{shows } \exists a. E = \{a\} \\
\text{proof} - \\
\text{from } \text{assms} \text{ obtain } a \text{ where } a \in E \\
\text{by force} \\
\text{let } ?E' &= E - \{a\} \\
\text{have } \text{finite } ?E' \\
\end{align*}
using assms card-ge-0-finite by force
hence card (insert a ?E') = 1 + card ?E'
using card-insert by fastforce
moreover have E = insert a ?E'
using (a ∈ E) by blast
ultimately have card E = 1 + card ?E'
by simp
hence card ?E' = 0
using assms by simp
hence ?E' = {}
using finite ?E' by simp
thus ?thesis
using (a ∈ E) by blast
qed

definition function fold on set A where fold A a = f "a " for a ∈ A

lemma card-2-elements:
assumes card E = 2
shows ∃ a b. E = {a, b} ∧ a ≠ b
proof –
from assms obtain a where a ∈ E
by force
let ?E' = E - {a}

have finite ?E'
using assms card-ge-0-finite by force
hence card (insert a ?E') = 1 + card ?E'
using card-insert by fastforce
moreover have E = insert a ?E'
using (a ∈ E) by blast
ultimately have card E = 1 + card ?E'
by simp
hence card ?E' = 1
using assms by simp
then obtain b where ?E' = {b}
using card-1-element by blast
hence E = {a, b}
using (a ∈ E) by blast
moreover have a ≠ b
using ⟨?E' = {b}⟩ by blast
ultimately show ?thesis
by blast
qed

lemma bij-lift:
assumes bij-betw f A B
shows bij-betw (λx. f ' x) (Pow A) (Pow B)
proof –
have f: inj-on f A f ' A = B
using assms unfolding bij-betw-def by simp-all
have \( \text{inj-on} \ (\lambda e. f \ e) \) (Pow \( A \))

unfolding \( \text{inj-on-def} \) by clarify (metis \( f(1) \) inv-into-image-cancel)

moreover have \( \lambda e. f \ e \ ' \) (Pow \( A \)) = (Pow \( B \))

by (metis \( f(2) \) image-Pow-surj)

ultimately show \(?thesis\)

unfolding \( \text{bij-betw-def} \) by simp

qed

lemma \( \text{card-inj-subs} \): \( \text{inj-on} \ f \ A \Rightarrow B \subseteq A \Rightarrow \text{card} \ (f \ ' B) = \text{card} B \)

by (metis card-image subset-inj-on)

lemma \( \text{image-comp-cong} \):

\( \forall a. a \in A \Rightarrow f a = f (g a) \Rightarrow f \ ' A = f \ ' (g \ ' A) \)

by (auto simp: image-iff)

abbreviation \( \text{less-fun} \) :: \( \text{(nat} \Rightarrow \text{real}) \Rightarrow \text{(nat} \Rightarrow \text{real}) \Rightarrow \text{bool} \) (infix \( \ll \) \( 50 \)) where

\( f \ll g \equiv (\lambda n. f n \ / \ g n) \Rightarrow 0 \)

context

fixes \( f \) :: \( \text{nat} \Rightarrow \text{real} \)

begin

lemma \( \text{LIMSEQ-power-zero} \): \( f \Rightarrow 0 \Rightarrow 0 < n \Rightarrow (\lambda x. f x \ ^{n} \ : \ \text{real}) \Rightarrow 0 \)

by (metis less-not-refl3 power-eq-0-iff tendsto-power)

lemma \( \text{LIMSEQ-cong} \):

assumes \( f \Rightarrow x \Rightarrow \forall \infty n. f n = g n \)

shows \( g \Rightarrow x \)

by (rule real-tendsto-sandwich[where \( f = f \) and \( h = f \) \( \text{and eventually-elim1[OF assms(2)]} \) \( \text{eventually-elim1[OF assms(2)]}\) (auto simp: assms(1))

print-statement \( \text{Lim-transform-eventually} \)

lemma \( \text{LIMSEQ-le-zero} \):

assumes \( g \Rightarrow 0 \Rightarrow \forall \infty n. 0 \leq f n \ \forall \infty n. f n \leq g n \)

shows \( f \Rightarrow 0 \)

by (rule real-tendsto-sandwich[OF assms(2) assms(3) tendsto-const assms(1)])

lemma \( \text{LIMSEQ-const-mult} \):

assumes \( f \Rightarrow a \)

shows \( (\lambda x. c \ * f x) \Rightarrow c \ * a \)

by (rule tendsto-const[where \( k = c \) \( \text{assms}\) ])

lemma \( \text{LIMSEQ-const-div} \):

assumes \( f \Rightarrow a \ c \neq 0 \)

shows \( (\lambda x. f x / c) \Rightarrow a / c \)

using \( \text{LIMSEQ-const-mult[where} \ c = 1/c \) \( \text{assms}\) \( \text{by simp} \)

end
lemma quot-bounds:
fixes x :: 'a :: linordered-field
assumes x ≤ x' y' ≤ y 0 < y 0 ≤ x 0 < y'
shows x / y ≤ x' / y'
proof (rule order-trans)
  have 0 ≤ y
  using assms by simp
  thus x / y ≤ x' / y
  using assms by (simp add: divide-right-mono)
next
  have 0 ≤ x'
  using assms by simp
  moreover have 0 < y * y'
  using assms by simp
  ultimately show x' / y ≤ x' / y'
  using assms by (simp add: divide-left-mono)
qed

lemma less-fun-bounds:
assumes f' ≪ g'
∀∞ n. f n ≤ f' n
∀∞ n. g' n ≤ g n
∀∞ n. 0 ≤ f n
∀∞ n. 0 < g' n
shows f ≪ g
proof (rule real-tendsto-sandwich)
  show ∀∞ n. 0 ≤ f n / g n
  using assms(4,5) by eventually-elim simp
  next
  show ∀∞ n. f n / g n ≤ f' n / g' n
  using assms(2−) by eventually-elim (simp only: quot-bounds)
qed (auto intro: assms(1))

lemma less-fun-const-quot:
assumes f ≪ g c ≠ 0
shows (λn. b * f n) ≪ (λn. c * g n)
proof
  have (λn. (b * (f n / g n)) / c) −−−−> (b * 0) / c
  using assms by (rule LIMSEQ-const-div[OF LIMSEQ-const-mult])
  hence (λn. (b * (f n / g n)) / c) −−−−> 0
  by simp
  with eventually-sequentiallyI show ?thesis
  by (rule Lim-transform-eventually) simp
qed

lemma partition-set-of-intersecting-sets-by-card:
assumes finite A
shows {B. A ∩ B ≠ {}} = (⋃n ∈ {1..card A}. {B. card (A ∩ B) = n})
proof (rule set-eqI, rule iffI)
  fix B
assume $B \in \{ B. A \cap B \neq \{\}\}$
hence $0 < \text{card} (A \cap B)$
using assms by auto
moreover have $\text{card} (A \cap B) \leq \text{card} A$
using assms by (simp add: card_mono)
ultimately have $\text{card} (A \cap B) \in \{1..\text{card} A\}$
by simp
thus $B \in (\bigcup n \in \{1..\text{card} A\}. \{ B. \text{card} (A \cap B) = n\})$
by blast
qed force

lemma card-set-of-intersecting-sets-by-card:
assumes $A \subseteq I$ finite $I$ $k \leq n$ $n \leq \text{card} I$ $k \leq \text{card} A$
shows $\text{card} \{ B. B \subseteq I \land \text{card} B = n \land \text{card} (A \cap B) = k\} = (\text{card} A \choose k) \times ((\text{card} I - \text{card} A) \choose (n - k))$
proof
¬note finite-A = finite-subset[OF assms(1,2)]

have $\text{card} \{ B. B \subseteq I \land \text{card} B = n \land \text{card} (A \cap B) = k\} = \text{card} \{(K. K \subseteq A \land \text{card} K = k) \times \{B'. B' \subseteq I - A \land \text{card} B' = n - k\}\}$ (is $\text{card} ?lhs = \text{card} ?rhs$)
proof (rule bij_betw_same_card[symmetric])
let $\text{iff} = \lambda (K, B'). K \cup B'$
have inj-on $\text{iff} ?rhs$
by (blast intro: inj-onI)
moreover have $\text{iff} ' ?rhs = ?lhs$
proof (rule set_eqI, rule iffI)
fix $B$
assume $B \in \text{iff} ' ?rhs$
then obtain $K B'$ where $K: K \subseteq A \land \text{card} K = k \land B' \subseteq I - A \land \text{card} B' = n - k$
by blast
show $B \in ?lhs$
proof safe
fix $x$ assume $x \in B$ thus $x \in I$
using $K$ $A \subseteq I$ by blast
next
have $\text{card} B = \text{card} K + \text{card} B' - \text{card} (K \cap B')$
using $K$ assms by (metis card_union finite-A finite-subset finite-Diff)
moreover have $K \cap B' = \{\}$
using $K$ assms by blast
ultimately show $\text{card} B = n$
using $K$ assms by simp
next
have $A \cap B = K$
using $K$ assms(1) by blast
thus $\text{card} (A \cap B) = k$
using $K$ by simp
qed
next
fix B
assume B ∈ ?lhs
hence B: B ⊆ I card B = n card (A ∩ B) = k
by auto
let $?K = A ∩ B
let $?B’ = B – A
have $?K ⊆ A card $?K = k $?B’ ⊆ I – A
using B by auto
moreover have card $?B’ = n – k
using B finite-A assms(1) by (metis Int-commute card-Diff-subset-Int
finite-Un inf .left-idem le_iff_inf sup-absorb2)
ultimately have (?K, $?B’) ∈ ?rhs
by blast
moreover have B = ?f (?K, $?B’)
by auto
ultimately show B ∈ ?f ‘ ?rhs
by blast
qed
ultimately show bij-betw ?f ?rhs ?lhs
unfolding bij-betw-def ..
qed
also have ...
= (∑ K | K ⊆ A ∧ card K = k. card {B’. B’ ⊆ I – A ∧ card
B’ = n – k})
proof (rule card-SigmaI, safe)
show finite {K. K ⊆ A ∧ card K = k}
by (blast intro: finite-subset[where B = Pow A] finite-A)
next
fix K
assume K ⊆ A
thus finite {B’. B’ ⊆ I – A ∧ card B’ = n – k K}
using assms by auto
qed
also have ...
= card {K. K ⊆ A ∧ card K = k} * card {B’. B’ ⊆ I – A ∧ card
B’ = n – k}
by simp
also have ...
= (card A choose k) * (card (I – A) choose (n – k))
by (simp only: n-subsets[OF finite-A] n-subsets[OF finite-Diff[OF assms(2)]])
also have ...
= (card A choose k) * ((card I – card A) choose (n – k))
by (simp only: card-Diff-subset[OF finite-A assms(1)])
finally show ?thesis
.
qed

lemma card-dep-pair-set:
assumes finite A ∧ a. a ⊆ A ⇒ finite (f a)
shows card {(a, b). a ⊆ A ∧ card a = n ∧ b ⊆ f a ∧ card b = g a} = (∑ a | a
⊆ A ∧ card a = n. card (f a) choose g a) (is card ?S = ?C)
proof –
have \( S = \Sigma \{ a. \ a \subseteq A \land \card a = n \} (\lambda a. \ \{ h. \ h \subseteq f a \land \card b = g \ a \}) \) (is \(-\)\( = \Sigma \ A \ ?B \))
by auto

have \( \card (\Sigma \ A \ ?B) = (\sum a \in \{ a. \ a \subseteq A \land \card a = n \}. \ \card (\ ?B a)) \)
proof (rule card-SigmaI, safe)
  show finite \( \ ?A \)
    by (rule finite-subset[OF - finite-Collect-subsets[OF assms(1)]]) blast
next
  fix \( a \)
  assume \( a \subseteq A \)
  hence finite \( f a \)
    by (fact assms(2))
  thus finite \( \ ?B a \)
    by (rule finite-subset[rotated, OF finite-Collect-subsets]) blast
qed
also have \( \ldots = S \)
proof (rule setsum.cong)
  fix \( a \)
  assume \( a \in \{ a. \ a \subseteq A \land \card a = n \} \)
  hence finite \( f a \)
    using assms(2) by blast
  thus \( \card (\ ?B a) = \card (f a) \) choose \( g a \)
    by (fact n-subsets)
qed simp
finally have \( \card (\Sigma \ A \ ?B) = S \)
by (subst \( S \))
qed

lemma setprod-cancel-nat:
— Contributed by Manuel Eberl
fixes \( f::'a \Rightarrow \text{nat} \)
assumes \( B \subseteq A \) and finite \( A \) and \( \forall x \in B. \ f x \neq 0 \)
shows \( \text{setprod} f A / \text{setprod} f B = \text{setprod} f (A - B) \) (is \( \ ?A / \ ?B = S \))
proof –
from \( \text{setprod.subset-diff}[OF assms(1,2)] \) have \( S = \ ?A \ ?B \) by auto
moreover have \( \ ?B \neq 0 \) using \( \text{assms} \) by (simp add: finite-subset)
ultimately show \( \ ?\text{thesis} \) by simp
qed

lemma setprod-id-cancel-nat:
— Contributed by Manuel Eberl
fixes \( A::\text{nat set} \)
assumes \( B \subseteq A \) and finite \( A \) and \( 0 \notin B \)
shows \( \prod A / \prod B = \prod (A - B) \)
using \( \text{assms}(1\sim2) \) by (rule setprod-cancel-nat) (metis \( \text{assms}(3) \))
lemma (in prob-space) integrable-squareD:
— Contributed by Johannes Hlzl

fixes X :: - ⇒ real
assumes integrable M (λx. (X x) ^2) X ∈ borel-measurable M
shows integrable M X
proof
- have integrable M (λx. max 1 ((X x) ^2))
  using assms by auto
then show integrable M X
proof (rule integrable-bound[OF - - always-eventually[OF allI]])
  fix x show norm (X x) ≤ norm (max 1 ((X x) ^2))
    using abs-le-square-iff[of 1 X x] power-increasing[of 1 2 abs (X x)]
    by (auto split: split-max)
qed fact
qed

end
theory Prob-Lemmas
imports Probability
../Girth-Chromatic/Girth-Chromatic
Ugraph-Misc
begin

3 Lemmas about probabilities

In this section, auxiliary lemmas for computing bounds on expectation and
probabilities of random variables are set up.

3.1 Indicator variables and valid probability values

abbreviation rind :: 'a set ⇒ 'a ⇒ real where
  rind ≡ indicator

lemma product-indicator:
  rind A x * rind B x = rind (A ∩ B) x
unfolding indicator-def
by auto

We call a real number ‘valid’ iff it is in the range 0 to 1, inclusively, and
additionally ‘nonzero’ iff it is neither 0 nor 1.

abbreviation valid-prob (p :: real) ≡ 0 ≤ p ∧ p ≤ 1
abbreviation nonzero-prob (p :: real) ≡ 0 < p ∧ p < 1

A function 'a ⇒ real is a ‘valid probability function’ iff each value in the
image is valid, and similarly for ‘nonzero’.

11
abbreviation valid-prob-fun \( f \equiv (\forall n. \text{valid-prob } (f n)) \)

abbreviation nonzero-prob-fun \( f \equiv (\forall n. \text{nonzero-prob } (f n)) \)

lemma nonzero-fun-is-valid-fun: nonzero-prob-fun \( f \Rightarrow \text{valid-prob-fun } f \)
by (simp add: less-imp-le)

3.2 Expectation and variance

context prob-space
begin

Note that there is already a notion of independent sets (see indep-set), but we use the following – simpler – definition:

definition indep \( A \ B \iff \text{prob } (A \cap B) = \text{prob } A \ast \text{prob } B \)

The probability of an indicator variable is equal to its expectation:

lemma expectation-indicator:
\( A \in \text{events } \Rightarrow \text{expectation } (\text{rind } A) = \text{prob } A \)
by simp

For a non-negative random variable \( X \), the Markov inequality gives the following upper bound:

\[ \Pr[X \geq a] \leq \frac{\text{E}[X]}{a} \]

lemma markov-inequality:
assumes \( \bigwedge a. 0 \leq X a \text{ and integrable } M X 0 < t \)
shows \( \text{prob } \{a \in \text{space } M. t \leq X a\} \leq \text{expectation } X / t \)
proof −
— proof adapted from edge-space.Markov-inequality, but generalized to arbitrary prob-spaces
have \( (\int \ast x. \text{ereal } (X x) \partial M) = (\int \ast x. X x \partial M) \)
using assms by (intro nn-integral-eq-integral) auto
thus ?thesis
using assms nn-integral-Markov-inequality[of X M space M 1 / t]
by (auto cong: nn-integral-cong simp: emeasure-eq-measure one-ereal-def)
qed

\[ \text{Var}[X] = \text{E}[X^2] - \text{E}[X]^2 \]

lemma variance-expectation:
fixes \( X :: 'a \Rightarrow \text{real} \)
assumes integrable M \( (\lambda x. (X x)^\ast 2) \) and \( X \in \text{borel-measurable } M \)
shows \( \text{integrable } M \ (\lambda x. (X x - \text{expectation } X)^\ast 2) \) (is ?integrable)
\( \text{variance } X = \text{expectation } (\lambda x. (X x)^\ast 2) - (\text{expectation } X)^\ast 2 \) (is ?variance)
proof −
have int: integrable M X
using integrable-squareD[OF assms] by simp
have \((\lambda x. (X x - \text{expectation } X)^2) = (\lambda x. (X x)^2 + (\text{expectation } X)^2 - (2 * X x \ast \text{expectation } X))\)
by (simp only: power2-diff)

hence

\[ \text{variance } X = \text{expectation } (\lambda x. (X x)^2) + (\text{expectation } X)^2 + \text{expectation } (\lambda x. - (2 * X x \ast \text{expectation } X)) \]

\(\text{?integrable}\)

using integral-add by (simp add: int assms prob-space)+

thus \(\text{?variance } \text{?integrable}\)

by (simp add: int power2-eq-square)+

qed

A corollary from the Markov inequality is Chebyshev’s inequality, which gives an upper bound for the deviation of a random variable from its expectation:

\[ \text{Pr}[|Y - E[Y]| \geq s] \leq \frac{\text{Var}[X]}{s^2} \]

lemma chebyshev-inequality:

fixes \(Y \:: \cdot{\cdot a} \Rightarrow \text{real}\)

assumes \(Y\text{-int}: \text{integrable } M (\lambda y. (Y y)^2)\)

assumes \(Y\text{-borel}: Y \in \text{borel-measurable } M\)

fixes \(s \:: \text{real}\)

assumes \(s\text{-pos}: 0 < s\)

shows \(\text{prob } \{a \in \text{space } M. s \leq |Y a - \text{expectation } Y|\} \leq \text{variance } Y / s^2\)

proof

let \(?X = \lambda a. (Y a - \text{expectation } Y)^2\)

let \(?t = s^2\)

have \(0 < ?t\)

using s-pos by simp

hence \(\text{prob } \{a \in \text{space } M. ?t \leq ?X a\} \leq \text{variance } Y / s^2\)


moreover have \(\{a \in \text{space } M. ?t \leq ?X a\} = \{a \in \text{space } M. s \leq |Y a - \text{expectation } Y|\}\)

using abs-le-square-iff s-pos by force

ultimately show \(\text{?thesis}\)

by simp

qed

Hence, we can derive an upper bound for the probability that a random variable is 0.

corollary chebyshev-prob-zero:

fixes \(Y \:: \cdot{\cdot a} \Rightarrow \text{real}\)

assumes \(Y\text{-int}: \text{integrable } M (\lambda y. (Y y)^2)\)

assumes \(Y\text{-borel}: Y \in \text{borel-measurable } M\)

assumes \(\mu\text{-pos}: \text{expectation } Y > 0\)
shows \( \text{prob} \{ a \in \text{space } M. Y a = 0 \} \leq \text{expectation} \ (\lambda y. (Y y)^2) / (\text{expectation} Y)^2 - 1 \)

proof -
  let \( ?s = \text{expectation} Y \)

have \( \text{prob} \{ a \in \text{space } M. Y a = 0 \} \leq \text{prob} \{ a \in \text{space } M. ?s \leq |Y a - ?s| \} \)
  using Y-borel by (auto intro!: finite-measure-monotone borel-measurable-diff borel-measurable-abs borel-measurable-\leq)
  also have \( \ldots \leq \text{variance } Y / ?s^2 \)
    using assms by (fact chebyshev-inequality)
  also have \( \ldots = (\text{expectation} (\lambda y. (Y y)^2) - ?s^2) / ?s^2 \)
    using Y-int Y-borel by (simp add: variance-expectation)
  also have \( \ldots = \text{expectation} (\lambda y. (Y y)^2) / ?s^2 - 1 \)
    using \( \mu \)-pos by (simp add: field-simps)
  finally show \( \text{?thesis} \).
qed

end

3.3 Sets of indicator variables

This section introduces some inequalities about expectation and other values related to the sum of a set of random indicators.

locale prob-space-with-indicators = prob-space +
  fixes I :: 'i set
  assumes finite-I: finite I

  fixes A :: 'i => 'a set
  assumes A: A 'I \subseteq \text{events}

  assumes prob-non-zero: \( \exists i \in I. 0 < \text{prob } (A i) \)

begin

We call the underlying sets \( A i \) for each \( i \in I \), and the corresponding indicator variables \( X i \). The sum is denoted by \( Y \), and its expectation by \( \mu \).

definition \( X i = \text{rind } (A i) \)
definition \( Y x = \sum i \in I. X i x \)
definition \( \mu = \text{expectation } Y \)

In the lecture notes, the following two relations are called \( \sim \) and \( \nsim \), respectively. Note that they are not the opposite of each other.

abbreviation ineq-indep :: 'i \Rightarrow 'i \Rightarrow bool where
  ineq-indep i j \equiv \( i \neq j \land \text{indep } (A i) (A j) \)\)

abbreviation ineq-dep :: 'i \Rightarrow 'i \Rightarrow bool where
  ineq-dep i j \equiv \( i \neq j \land \neg \text{indep } (A i) (A j) \)
\[
\Delta_a = \left( \sum_{i \in I} \sum_{j : j \in I \land i \neq j} \text{prob}(A_i \cap A_j) \right)
\]

\[
\Delta_d = -\left( \sum_{i \in I} \sum_{j : j \in I \land \text{ineq-dep} \ i \ j} \text{prob}(A_i \cap A_j) \right)
\]

**Lemma \(\Delta\text{-zero}:\)**

assumes \(\bigwedge_{i j} i \in I \implies j \in I \implies i \neq j \implies \text{indep}(A_i) \ (A_j)\)

shows \(\Delta_d = 0\)

proof –

\{
  \text{fix } i
  \text{assume } i \in I
  \text{hence } \{j : j \in I \land \text{ineq-dep} \ i \ j\} = \{}
  \text{using assms by auto}
  \text{hence } \left( \sum_{j \in I \land \text{ineq-dep} \ i \ j} \text{prob}(A_i \cap A_j) \right) = 0
  \text{using setsum.empty by metis}
\}

\text{hence } \Delta_d = (0 :: real) \ast \text{card } I

\text{unfolding } \Delta_d\text{-def by simp}

\text{thus } \text{?thesis}

\text{by simp}

qed

**Lemma \(A\text{-events}[measurable]:\)** \(i \in I \implies A_i \in \text{events}\)

using \(A\) by auto

**Lemma \(expectation-X-Y:\)** \(\mu = \left( \sum_{i \in I} \text{expectation} (X_i) \right)\)

unfolding \(\mu\text{-def} \ Y\text{-def} \ [\text{abs-def} \ X\text{-def}]\)

by simp

**Lemma \(expectation-X\text{-non-zero}:\)** \(\exists i \in I. \ 0 < \text{expectation} (X_i)\)

unfolding \(X\text{-def}\) using \(\text{prob-non-zero} \ \text{expectation-indicator}\) by simp

**Corollary \(\mu\text{-non-zero}[simp]:\)** \(0 < \mu\)

unfolding \(\text{expectation-X-Y}\)

using \(\text{expectation-X\text{-non-zero}}\)

by \(\text{(auto intro!: setsum-lower finite-I auto simp add: expectation-indicator X-def measure-nonneg simp del: integral-indicator)}\)

**Lemma \(\Delta\text{-nonneg}:\)** \(0 \leq \Delta_d\)

unfolding \(\Delta_d\text{-def}\)

by \(\text{(simp add: setsum-nonneg measure-nonneg)}\)

**Corollary \(\mu\text{-sq\text{-non-zero}[simp]}:\)** \(0 < \mu^2\)

by \(\text{(rule zero-less-power)}\) simp

**Lemma \(Y\text{-square-unfold}:\)** \(\lambda x. \ (Y \ x)^2 = \left( \sum_{i \in I} \sum_{j \in I} \text{rind} (A_i \cap A_j) \ x \right)\)

unfolding \(\text{fun-eq-iff Y-def X-def}\)
by (auto simp: setsum-square product-indicator)

lemma integrable-Y-sq[simp]: integrable $M$ $(\lambda y. (Y y)^2)$
unfolding Y-square-unfold
by (simp add: sets.Int)

lemma measurable-Y[measurable]: $Y \in \text{borel-measurable } M$
unfolding Y-def
abs-def X-def by simp

lemma expectation-Y-∆: $\text{expectation} (\lambda x. (Y x)^2) = \mu + \Delta_a$
proof
let $?ei = \lambda i j. \text{expectation} (\text{rind} (A i \cap A j))$

have $\text{expectation} (\lambda x. (Y x)^2) = (\sum i \in I. \sum j \in I. ?ei i j)$
unfolding Y-square-unfold by (simp add: sets.Int)
also have ... $= (\sum i \in I. \sum j \in I. \text{if } i = j \text{ then } ?ei i j \text{ else } ?ei i j)$
by simp
also have ... $= (\sum i \in I. (\sum j \mid j \in I \wedge i = j. ?ei i j) + (\sum j \mid j \in I \wedge i \neq j. ?ei i j))$
by (simp only: setsum-split[OF finite-I])
also have ... $= \mu + \Delta_a$
proof
have $?lhs = \mu$
proof
{
fix $i$
assume $i: i \in I$
have $(\sum j \mid j \in I \wedge i = j. ?ei i j) = (\sum j \mid j \in I \wedge i = j. ?ei i i)$
by simp
also have ... $= (\sum j \mid i = j. ?ei i i)$
using $i$ by metis
also have ... $= \text{expectation} (\text{rind} (A i))$
by auto
finally have $(\sum j \mid j \in I \wedge i = j. ?ei i j) = \ldots .$
}

hence $?lhs = (\sum i \in I. \text{expectation} (\text{rind} (A i)))$
by force
also have ... $= \mu$
unfolding expectation-X-Y X-def ..
finally show $?lhs = \mu$.
qed
moreover have $?rhs = \Delta_a$
proof
{
fix $i j$
assume $i \in I \land j \in I$

16
with \( A \) have \( A \cap A j \in \text{events} \) by blast
hence \( ?e_i \ j = \text{prob} \ (A i \cap A j) \)
by (fact expectation-indicator)
}
thus \( ?\text{thesis} \)
unfolding \( \Delta_a\)-def by simp
qed
ultimately show \( ?\text{lhs} + ?\text{rhs} = \mu + \Delta_a \)
by simp
qed
finally show \( ?\text{thesis} \).
qed

lemma \( \Delta\)-expectation-X: \( \Delta_a \leq \mu^2 + \Delta_d \)

proof –
let \( ?p = \lambda i \ j. \ \text{prob} \ (A i \cap A j) \)
let \( ?p' = \lambda i \ j. \ \text{prob} \ (A i) \ast \text{prob} \ (A j) \)
let \( ?\text{ie} = \lambda i \ j. \ \text{indep} \ (A i) \ (A j) \)

have \( \Delta_a = (\sum i \in I. \sum j : j \in I \land i \neq j \land ?\text{ie} i j \then ?p i j \else ?p i j) \)
unfolding \( \Delta_a\)-def by simp
also have \( \ldots \ = (\sum i \in I. (\sum j : j \in I \land \text{ineq-dep} i j \ \wedge \ ?p i j) + (\sum j : j \in I \land \text{ineq-dep} i j \ \wedge \ ?p i j) + \Delta_d \) (is = - \ ?\text{lhs} + -)
unfolding \( \Delta_d\)-def by (fact setsum.distrib)
also have \( \ldots \leq \mu^2 + \Delta_d \)
proof (rule add-right-mono)
have \( (\sum i \in I. \sum j : j \in I \land \text{ineq-dep} i j \ ?p i j) = (\sum i \in I. \sum j \in I \land \text{ineq-dep} i j \ ?p' i j) \)
unfolding indep-def by simp
also have \( \ldots \leq (\sum i \in I. \sum j \in I \ ?p' i j) \)
proof (rule setsum-mono)
fix \( i \)
assume \( i \in I \)
show \( (\sum j : j \in I \land \text{ineq-dep} i j \ ?p' i j) \leq (\sum j \in I \ ?p' i j) \)
by (rule setsum-upper[OF finite-I]) (simp add: measure-nonneg zero-le-mult-iff)
qed
also have \( \ldots = (\sum i \in I. \text{prob} \ (A i)) \ ?2 \)
by (fact setsum-square[symmetric])
also have \( \ldots = (\sum i \in I. \text{expectation} \ (X i)) \ ?2 \)
unfolding X-def using expectation-indicator A by simp
also have \( \ldots = \mu^2 \)
using expectation-X-Y[symmetric] by simp
finally show \( ?\text{lhs} \leq \mu^2 \).
qed
finally show \( ?\text{thesis} \).
qed
lemma prob-\(\mu\)-\(\Delta_a\): \(\text{prob} \{ a \in \text{space } M. \ Y a = 0 \} \leq 1/\mu + \Delta_a / \mu^2 - 1\)
proof
  have \(\text{prob} \{ a \in \text{space } M. \ Y a = 0 \} \leq \text{expectation} (\lambda y. (Y y)^2) / \mu^2 - 1\)
  unfolding \(\mu\)-def by (rule chebyshev-prob-zero) (simp add: \(\mu\)-def[symmetric])
  also have \(\ldots = (\mu + \Delta_a) / \mu^2 - 1\)
  using expectation-Y-\(\Delta\) by simp
  also have \(\ldots = 1/\mu + \Delta_a / \mu^2 - 1\)
  unfolding power2-eq-square by (simp add: field-simps add-divide-distrib)
finally show \(\therefore\thesis\).
qed

lemma prob-\(\mu\)-\(\Delta_d\): \(\text{prob} \{ a \in \text{space } M. \ Y a = 0 \} \leq 1/\mu + \Delta_d / \mu^2\)
proof
  have \(\text{prob} \{ a \in \text{space } M. \ Y a = 0 \} \leq 1/\mu + \Delta_d / \mu^2 - 1\)
  by (fact prob-\(\mu\)-\(\Delta_a\))
  also have \(\ldots = (1/\mu - 1) + \Delta_d / \mu^2\)
  by simp
  also have \(\ldots \leq (1/\mu - 1) + (\mu^2 + \Delta_d)/\mu^2\)
  using divide-right-mono[OF \(\Delta\)-expectation-X] by simp
  also have \(\ldots = 1/\mu + \Delta_d / \mu^2\)
  using \(\mu\)-sq-non-zero by (simp add: field-simps)
finally show \(\therefore\thesis\).
qed

end

end

4 Lemmas about undirected graphs

theory Ugraph-Lemmas
imports
  Prob-Lemmas
  ../Girth-Chromatic/Girth-Chromatic
  Lattices-Big
begin

The complete graph is a graph where all possible edges are present. It is wellformed by definition.

definition complete :: nat set \(\Rightarrow\) ugraph where
  complete \(V = (V, \text{all-edges } V)\)

lemma complete-wellformed: wellformed (complete \(V\))
unfolding complete-def wellformed-def all-edges-def
by simp

If the set of vertices is finite, the set of edges in the complete graph is finite.

lemma all-edges-finite: finite \(V \Rightarrow\) finite (all-edges \(V\))
A graph is called ‘finite’ if its set of edges and its set of vertices are finite.

**Definition** finite-graph $G \equiv$ finite (uverts $G$) $\land$ finite (uedges $G$)

The complete graph is finite.

**Corollary** complete-finite: finite $V \implies$ finite-graph (complete $V$)

A graph is called ‘nonempty’ if it contains at least one vertex and at least one edge.

**Definition** nonempty-graph $G \equiv$ uverts $G \neq \{\} \land$ uedges $G \neq \{\}$

A random graph is both wellformed and finite.

**Lemma** (in edge-space) wellformed-and-finite:
- **Assumes** $E \in$ Pow S-edges
- **Shows** finite-graph (edge-ugraph $E$) $\land$ wellformed (edge-ugraph $E$)

**Proof**
- **Unfolding** edge-ugraph-def $S$-verts-def by simp
- **Show** finite (uverts (edge-ugraph $E$))
- **Using** asms **Unfolding** edge-ugraph-def $S$-edges-def by (auto intro: all-edges-finite)
- **Next** show wellformed (edge-ugraph $E$)
  - **Using** complete-wellformed **Unfolding** edge-ugraph-def $S$-edges-def complete-def
  - **Unfolding** wellformed-def by force

**Qed**

The probability for a random graph to have $e$ edges is $p^e$.

**Lemma** (in edge-space) cylinder-empty-prob:
- $A \subseteq S$-edges $\implies$ prob (cylinder $S$-edges $A$ $\{\}$) $=$ $p^{-}$ (card $A$)

**Using** cylinder-prob by auto
4.1 Subgraphs

**definition** subgraph :: ugraph ⇒ ugraph ⇒ bool

**where**

subgraph G' G ≡ uverts G' ⊆ uverts G ∧ uedges G' ⊆ uedges G

**lemma** subgraph-refl: subgraph G G

**unfolding** subgraph-def by simp

**lemma** subgraph-trans: subgraph G'' G' ⇒ subgraph G' G ⇒ subgraph G'' G

**unfolding** subgraph-def by auto

**lemma** subgraph-antisym: subgraph G G' ⇒ subgraph G' G ⇒ G = G'

**unfolding** subgraph-def by (auto simp add: Product-Type.prod-eqI)

**lemma** subgraph-complete:

**assumes** uwellformed G

**shows** subgraph G (complete (uverts G))

**proof** –

{  
  fix e
  assume e ∈ uedges G
  with assms have card e = 2 and u: \( \forall u. u ∈ e \Rightarrow u ∈ uverts G \)
  unfolding uwellformed-def by auto
  moreover then obtain u v where e = \{ u, v \} u ≠ v
  by (metis card-2-elements)
  ultimately have e = mk-uedge (u, v) u ∈ uverts G v ∈ uverts G
  by auto
  hence e ∈ all-edges (uverts G)
  unfolding all-edges-def using (u ≠ v; by fastforce)
}

thus ?thesis

**unfolding** complete-def subgraph-def by auto

qed

**corollary** wellformed-all-edges: uwellformed G ⇒ uedges G ⊆ all-edges (uverts G)

**using** subgraph-complete subgraph-def complete-def by simp

**lemma** subgraph-finite: [ finite-graph G; subgraph G' G ] ⇒ finite-graph G'

**unfolding** finite-graph-def subgraph-def by (metis rev-finite-subset)

**corollary** wellformed-finite:

**assumes** finite (uverts G) and uwellformed G

**shows** finite-graph G

**proof** (rule subgraph-finite[where G = complete (uverts G)])

  show subgraph G (complete (uverts G))

  20
using assms by (simp add: subgraph-complete)

next
have finite (uedges (complete (uverts G)))
  using complete-finite-edges[OF assms(1)] .
thus finite-graph (complete (uverts G))
  unfolding finite-graph-def complete-def using assms(1) by auto

qed

definition subgraphs :: ugraph ⇒ ugraph set where
subgraphs G = {G′. subgraph G′ G}

definition nonempty-subgraphs :: ugraph ⇒ ugraph set where
nonempty-subgraphs G = {G′. uwellformed G′ ∧ subgraph G′ G ∧ nonempty-graph G′}

lemma subgraphs-finite: 
  assumes finite-graph G
  shows finite (subgraphs G)
proof –
  have subgraphs G = {(V′, E′). V′ ⊆ uverts G ∧ E′ ⊆ uedges G}
    unfolding subgraphs-def subgraph-def by force
  moreover have finite (uverts G) finite (uedges G)
    using assms unfolding finite-graph-def by auto
  ultimately show ?thesis
    by simp
qed

corollary nonempty-subgraphs-finite: finite-graph G ⇒ finite (nonempty-subgraphs G)
using subgraphs-finite
unfolding nonempty-subgraphs-def subgraphs-def
by auto

4.2 Induced subgraphs

definition induced-subgraph :: uvert set ⇒ ugraph ⇒ ugraph where
induced-subgraph V G = (V, uedges G ∩ all-edges V)

lemma induced-is-subgraph:
  V ⊆ uverts G ⇒ subgraph (induced-subgraph V G) G
  V ⊆ uverts G ⇒ subgraph (induced-subgraph V G) (complete V)
unfolding subgraph-def induced-subgraph-def complete-def
by simp

lemma induced-wellformed: uwellformed G ⇒ V ⊆ uverts G ⇒ uwellformed (induced-subgraph V G)
unfolding uwellformed-def induced-subgraph-def all-edges-def
by force
lemma subgraph-union-induced:
  assumes uverts \( H_1 \subseteq S \) and uverts \( H_2 \subseteq T \)
  assumes uwellformed \( H_1 \) and uwellformed \( H_2 \)
  shows subgraph \( H_1 \) \((\text{induced-subgraph} \ S \ G) \) \& subgraph \( H_2 \) \((\text{induced-subgraph} \ T \ G) \) \iff
  \( \text{subgraph} \ (\text{uverts} \ H_1 \cup \text{uverts} \ H_2, \text{uedges} \ H_1 \cup \text{uedges} \ H_2) \) \((\text{induced-subgraph} \ (S \cup T) \ G) \)
unfolding induced-subgraph-def subgraph-def
apply auto
using all-edges-mono apply blast
using all-edges-mono apply blast
using assms\((1,2)\) wellformed-all-edges[OF assms\((3)\)] wellformed-all-edges[OF assms\((4)\)]
all-edges-mono[OF assms\((1)\)] all-edges-mono[OF assms\((2)\)]
apply auto
done

lemma (in edge-space) induced-subgraph-prob:
  assumes uverts \( H \subseteq V \) and uwellformed \( H \) and \( V \subseteq S\text{-verts} \)
  shows prob \( \{e \in \text{space} \ P. \ \text{subgraph} \ H \ (\text{induced-subgraph} \ V \ (\text{edge-ugraph} \ e))\} \)
  \( = p \ ^{\ \text{card} \ (\text{uedges} \ H)} \) \((\text{is prob} \ ?A = -)\)
proof −
  have prob \( ?A = \text{prob} \ (\text{cylinder} \ S\text{-edges} \ (\text{uedges} \ H) \ \{\}\) \)
  unfolding cylinder-def space-eq subgraph-def induced-subgraph-def edge-ugraph-def S-edges-def
  by \((\text{rule arg-cong}[OF Collect-cong]) \ (\text{metis \ (no-types) \ assms(1,2) \ Pow-iff})\)
all-edges-mono \( \text{fst}-conv \ \text{inf-absorb}1 \ \text{inf-bot-left} \ \text{le-inf-iff} \ \text{snd-conv \ wellformed-all-edges})\)
also have \( \ldots = p \ ^{\ \text{card} \ (\text{uedges} \ H)} \)
proof \((\text{rule cylinder-empty-prob})\)
  have uedges \( H \subseteq \text{all-edges} \ (\text{uverts} \ H) \)
  by \((\text{rule wellformed-all-edges}[OF assms(2)])\)
also have all-edges \((\text{uverts} \ H) \subseteq \text{all-edges} \ S\text{-verts})\)
using assms by \((\text{auto simp: \ all-edges-mono}[OF \ subset-trans])\)
finally show uedges \( H \subseteq S\text{-edges} \)
  unfolding S-edges-def .
qed
finally show \( ?\text{thesis} \)
qed

4.3 Graph isomorphism

We define graph isomorphism slightly different than in the literature. The usual definition is that two graphs are isomorphic iff there exists a bijection between the vertex sets which preserves the adjacency. However, this complicates many proofs.

Instead, we define the intuitive mapping operation on graphs. An isomorphism between two graphs arises if there is a suitable mapping function from the first to the second graph. Later, we show that this operation can
be inverted.

```plaintext
fun map-ugraph :: (nat ⇒ nat) ⇒ ugraph ⇒ ugraph where
map-ugraph f (V, E) = (f ` V, (λe. f ` e) ` E)

definition isomorphism :: ugraph ⇒ ugraph ⇒ (nat ⇒ nat) ⇒ bool where
isomorphism G₁ G₂ f ≡ bij-betw f (uverts G₁) (uverts G₂) ∧ G₂ = map-ugraph f G₁

abbreviation isomorphic :: ugraph ⇒ ugraph ⇒ bool (-≃-) where
G₁ ≃ G₂ ≡ uwellformed G₁ ∧ uwellformed G₂ ∧ (∃ f. isomorphism G₁ G₂ f)
```

```plaintext
lemma map-ugraph-id: map-ugraph id = id
unfolding fun-eq-iff
by simp

lemma map-ugraph-trans: map-ugraph (g ◦ f) = (map-ugraph g) ◦ (map-ugraph f)
unfolding fun-eq-iff
by auto (metis imageI image-comp)+

lemma map-ugraph-wellformed:
  assumes uwellformed G and inj-on f (uverts G)
  shows uwellformed (map-ugraph f G)
unfolding uwellformed-def
proof safe
  fix e'
  assume e' ∈ uedges (map-ugraph f G)
  hence e' ∈ (λe. f ` e) ` (uedges G)
    by (metis map-ugraph.simps snd-conv surjective-pairing)
  then obtain e where e: e' = f ` e e ∈ uedges G
    by blast
  hence card e = 2 e ⊆ uverts G
    using assms(1) unfolding uwellformed-def by blast+
  thus card e' = 2
    using e(1) by (simp add: card-inj-subs[OF assms(2)])

  fix u'
  assume u' ∈ e'
  hence u' ∈ f ` e
    using e by force
  then obtain u where u: u' = f u u ∈ e
    by blast
  hence u ∈ uverts G
    using assms(1) u(2) unfolding uwellformed-def by blast
  hence u' ∈ f ` uverts G
    using u(1) by simp
  thus u' ∈ uverts (map-ugraph f G)
    by (metis map-ugraph.simps fst-conv surjective-pairing)
qed
```
lemma map-ugraph-finite: finite-graph \( G \implies \text{finite-graph} \ (\text{map-ugraph} \ f \ G) \)
unfolding finite-graph-def
by (metis finite-imageI fst-conv map-ugraph.simps snd-conv surjective-pairing)

lemma map-ugraph-preserves-sub:
assumes subgraph \( G_1 \subseteq G_2 \)
shows subgraph \( (\text{map-ugraph} \ f \ G_1) \ (\text{map-ugraph} \ f \ G_2) \)
proof
  have \( f \cdot \text{uverts} \ G_1 \subseteq f \cdot \text{uverts} \ G_2 \ (\lambda e. f \cdot e) \cdot \text{edges} \ G_1 \subseteq (\lambda e. f \cdot e) \cdot \text{edges} \ G_2 \)
    using assms(1) unfolding subgraph-def by auto
  thus \( ?\text{thesis} \)
    unfolding subgraph-def by (metis map-ugraph.simps fst-conv snd-conv surjective-pairing)
qed

lemma isomorphic-refl: uwellformed \( G \implies G \simeq G \)
unfolding isomorphism-def
by (metis bij-betw-id id-def map-ugraph-id)

lemma isomorphic-trans:
assumes \( G_1 \simeq G_2 \) and \( G_2 \simeq G_3 \)
shows \( G_1 \simeq G_3 \)
proof
  from assms obtain \( f_1 \ f_2 \) where
    bij: bij-betw \( f_1 \) (uverts \( G_1 \)) (uverts \( G_2 \)) bij-betw \( f_2 \) (uverts \( G_2 \)) (uverts \( G_3 \)) and
    map: \( G_2 = \text{map-ugraph} \ f_1 \ G_1 \ G_3 = \text{map-ugraph} \ f_2 \ G_2 \)
    unfolding isomorphism-def by blast
  let \( \overline{f} = f_2 \circ f_1 \)
  have bij-betw \( \overline{f} \) (uverts \( G_1 \)) (uverts \( G_3 \))
    using bij by (simp add: bij-betw-comp-iff)
  moreover have \( G_3 = \text{map-ugraph} \ f_2 \ G_1 \)
    using map by (simp add: map-ugraph-trans)
  moreover have uwellformed \( G_1 \) uwellformed \( G_3 \)
    using assms unfolding isomorphism-def by simp+
  ultimately show \( G_1 \simeq G_3 \)
    unfolding isomorphism-def by blast
qed

lemma isomorphic-sym:
assumes \( G_1 \simeq G_2 \)
shows \( G_2 \simeq G_1 \)
proof safe
  from assms obtain \( f \) where isomorphism \( G_1 \ G_2 \ f \)
    by blast
  hence bij: bij-betw \( f \) (uverts \( G_1 \)) (uverts \( G_2 \)) and map: \( G_2 = \text{map-ugraph} \ f \ G_1 \)
    unfolding isomorphism-def by auto

24
let \( f' = \text{inv-into} \ (\text{uverts } G_1) \ f \)

have bij': bij-betw \( f' \ (\text{uverts } G_2) \ (\text{uverts } G_1) \)
  by (rule bij-betw-inv-into) fact

moreover have uverts \( G_1 = f' \ uverts G_2 \)
  using bij' unfolding bij-betw-def by force

moreover have uedges \( G_1 = (\lambda e. \ f' \ e) \ uedges G_2 \)

proof –
  have uedges \( G_1 = \text{id} \ uedges G_1 \)
    by simp
  also have \( \ldots = (\lambda e. \ f' \ (f' \ e)) \ uedges G_1 \)
    proof (rule image-cong)
      fix a
      assume a ∈ uedges \( G_1 \)
      hence a ⊆ uverts \( G_1 \)
      using assms unfolding isomorphism-def uwellformed-def by blast
      thus id a = inv-into \((\text{uverts } G_1) \ f' \ f' \ a\)
        by (metis (full-types) id-def bij bij-betw-inj-on inv-into-image-cancel)
    qed simp
  also have \( \ldots = (\lambda e. \ f' \ e) \ uedges G_2 \)
    using bij map by (metis map-ugraph.simps pair-collapse snd-eqD)

finally show \( \text{thesis} \).

qed

ultimately have isomorphism \( G_2 \overset{?}{\cong} G_1 \ ?f' \)
  unfolding isomorphism-def by (metis map-ugraph.simps split-pairs)

thus \( \exists f. \text{isomorphism } G_2 \overset{?}{\cong} G_1 \ f \)
  by blast

qed (auto simp: assms)

lemma isomorphic-cards:
  assumes \( G_1 \overset{?}{\cong} G_2 \)
  shows \( \text{card} \ (\text{uverts } G_1) = \text{card} \ (\text{uverts } G_2) \ \text{(is } ?V) \)
    \( \text{card} \ (\text{uedges } G_1) = \text{card} \ (\text{uedges } G_2) \ \text{(is } ?E) \)

proof –
  from assms obtain f where
    bij: bij-betw f \((\text{uverts } G_1) \ (\text{uverts } G_2) \) and
    map: \( G_2 = \text{map-ugraph} \ f \ G_1 \)

    unfolding isomorphism-def by blast
  from assms have wellformed: uwellformed \( G_1 \) \( \text{uwellformed } G_2 \)
    by simp

  show ?V
    by (rule bij-betw-same-card[OF bij])

  let \( g = \lambda e. \ f' \ e \)
  have bij-betw \( g \ (\text{Pow} \ (\text{uverts } G_1)) \ (\text{Pow} \ (\text{uverts } G_2)) \)
by (rule bij-lift[OF bij])
moreover have uedges \( G_1 \subseteq \text{Pow}(\text{uverts} \ G_1) \)
using wellformed(1) unfolding uwellformed-def by blast
ultimately have \( \text{card}(\exists g. \text{uedges} \ G_1) = \text{card}(\text{uedges} \ G_1) \)
unfolding bij-betw-def by (metis card-inj-subs)
thus \( ?E \)
by (metis map map-ugraph.simps snd-cone surjective-pairing)
\( \text{qed} \)

4.4 Isomorphic subgraphs

The somewhat sloppy term ‘isomorphic subgraph’ denotes a subgraph which is isomorphic to a fixed other graph. For example, saying that a graph contains a triangle usually means that it contains any triangle, not the specific triangle with the nodes 1, 2 and 3. Hence, such a graph would have a triangle as an isomorphic subgraph.

definition subgraph-isomorphic :: \( \text{ugraph} \Rightarrow \text{ugraph} \Rightarrow \text{bool} \) where
\( G' \subseteq G \equiv \text{uwellformed} \ G \land (\exists G''. G' \cong G'' \land \text{subgraph} \ G'' \ G) \)

lemma subgraph-is-subgraph-isomorphic: \[ \forall \ (\text{uwellformed} \ G'; \ \text{uwellformed} \ G; \ \text{subgraph} \ G' \ G) \Rightarrow G' \subseteq G \]
unfolding subgraph-isomorphic-def
by (metis isomorphic-refl)

lemma isomorphic-is-subgraph-isomorphic: \( G_1 \cong G_2 \Rightarrow G_1 \subseteq G_2 \)
unfolding subgraph-isomorphic-def
by (metis subgraph-refl)

lemma subgraph-isomorphic-refl: \( \text{uwellformed} \ G \Rightarrow G \subseteq G \)
unfolding subgraph-isomorphic-def
by (metis isomorphic-refl subgraph-refl)

lemma subgraph-isomorphic-pre-iso-closed:
assumes \( G_1 \cong G_2 \land G_2 \subseteq G_3 \)
shows \( G_1 \subseteq G_3 \)
unfolding subgraph-isomorphic-def
proof
  show \( \text{uwellformed} \ G_3 \)
    using assms unfolding subgraph-isomorphic-def by blast
  next
  from assms(2) obtain \( G_2' \) where \( G_2 \simeq G_2' \) subgraph \( G_2' \ G_3 \)
  unfolding subgraph-isomorphic-def by blast
  moreover with assms(1) have \( G_1 \cong G_2' \)
    by (metis isomorphic-trans)
  ultimately show \( \exists G''. G_1 \cong G'' \land \text{subgraph} \ G'' \ G_3 \)
    by blast
  qed
lemma subgraph-isomorphic-pre-subgraph-closed:
assumes uwellformed $G_1$ and subgraph $G_1 G_2$ and $G_2 \subseteq G_3$
shows $G_1 \subseteq G_3$
unfolding subgraph-isomorphic-def
proof
  show uwellformed $G_3$
    using assms unfolding subgraph-isomorphic-def by blast
next
  from assms(3) obtain $G_2'$ where $G_2 \simeq G_2'$ subgraph $G_2' G_3$
  unfolding subgraph-isomorphic-def by blast
then obtain $f$ where bij: $bij\text{-}betw f$ (uverts $G_2$) (uverts $G_2'$) $G_2' = \text{map-ugraph}$ $f G_2$
  unfolding isomorphism-def by blast
  let $?G_1' = \text{map-ugraph}$ $f G_1$
  have bij-betw $f$ (uverts $G_1$) ($f \setminus$ uverts $G_1$)
    using bij(1) assms(2) unfolding subgraph-def by (auto intro: bij-betw-subset)
  moreover hence $G_1' \simeq G_2'$
    using $\text{map-ugraph}$-wellformed[of assms(1)] unfolding bij-betw-def ..
ultimately have $G_1 \simeq G_1'$
    using assms(1) unfolding isomorphism-def by (metis $\text{map-ugraph}$-simps
       fst-cone surjective-pairing)
  moreover have subgraph $?G_1' G_3$
    using subgraph-trans[of $\text{map-ugraph}$-preserves-sub[of assms(2)]] bij(2) subgraph $G_2' G_3$ by simp
  ultimately show $\exists G'$. $G_1 \simeq G''$ subgraph $G'' G_3$
    by blast
qed

lemmas subgraph-isomorphic-pre-closed = subgraph-isomorphic-pre-subgraph-closed
subgraph-isomorphic-pre-iso-closed

lemma subgraph-isomorphic-trans[trans]:
assumes $G_1 \subseteq G_2$ and $G_2 \subseteq G_3$
shows $G_1 \subseteq G_3$
proof
  from assms(1) obtain $G$ where $G_1 \simeq G$ subgraph $G G_2$
    unfolding subgraph-isomorphic-def by blast
  thus $?\text{thesis}$
    using assms(2) by (metis subgraph-isomorphic-pre-closed)
qed

lemma subgraph-isomorphic-post-iso: $H \subseteq G; G \simeq G' \implies H \subseteq G'$
using isomorphic-is-subgraph-isomorphic subgraph-isomorphic-trans
by blast


4.5 Density

The density of a graph is the quotient of the number of edges and the number of vertices of a graph.

**Definition** density :: ugraph ⇒ real where
density G = card (uedges G) / card (uverts G)

The maximum density of a graph is the density of its densest nonempty subgraph.

**Definition** max-density :: ugraph ⇒ real where
max-density G = Lattices-Big.Max (density ' nonempty-subgraphs G)

We prove some obvious results about the maximum density, such as that there is a subgraph which has the maximum density and that the (maximum) density is preserved by isomorphisms. The proofs are a bit complicated by the fact that most facts about linorder-class.Max require non-emptiness of the target set, but we need that anyway to get a value out of it.

**Lemma** subgraph-has-max-density:
assumes finite-graph G and nonempty-graph G and uwellformed G
shows ∃ G'. density G' = max-density G ∧ subgraph G' G ∧ nonempty-graph G' ∧ finite-graph G' ∧ uwellformed G'

**Proof**
- have G ∈ nonempty-subgraphs G
  unfolding nonempty-subgraphs-def using subgraph-refl assms by simp
- hence density G ∈ density ' nonempty-subgraphs G by simp
- hence (density ' nonempty-subgraphs G) ≠ {} by fast
- hence max-density G ∈ (density ' nonempty-subgraphs G)
  unfolding max-density-def by (auto simp add: nonempty-subgraphs-finite[OF assms(1)] Max.closed)
- thus ?thesis
  unfolding nonempty-subgraphs-def using subgraph-finite[OF assms(1)] by force
qed

**Lemma** max-density-is-max:
assumes finite-graph G and finite-graph G' and nonempty-graph G' and uwellformed G' and subgraph G' G
shows density G' ≤ max-density G

**Proof** (rule Max-ge)
using assms(1) by (simp add: nonempty-subgraphs-finite)

next
show density G' ∈ density ' nonempty-subgraphs G
  unfolding nonempty-subgraphs-def using assms by blast
qed
lemma max-density-gr-zero:
  assumes finite-graph G and nonempty-graph G and uwellformed G
  shows 0 < max-density G
proof
  have 0 < card (uverts G) 0 < card (uedges G)
  using assms unfolding finite-graph-def nonempty-graph-def by auto
  hence 0 < density G
  unfolding density-def by simp
also have density G ≤ max-density G
  using assms by (simp add: max-density-is-max subgraph-refl)
finally show ?thesis.
qed

lemma isomorphic-density:
  assumes G_1 ≃ G_2
  shows density G_1 = density G_2
unfolding density-def
using isomorphic-cards[OF assms]
by simp

lemma isomorphic-max-density:
  assumes G_1 ≃ G_2 and nonempty-graph G_1 and nonempty-graph G_2 and
  finite-graph G_1 and finite-graph G_2
  shows max-density G_1 = max-density G_2
proof
  — The proof strategy is not completely straightforward. We first show that if
  two graphs are isomorphic, the maximum density of one graph is less or equal than
  the maximum density of the other graph. The reason is that this proof is quite
  long and the desired result directly follows from the symmetry of the isomorphism
  relation.\footnote{Some famous mathematician once said that if you prove that a ≤ b and b ≤ a, you
  know \emph{that} these numbers are equal, but not \emph{why}. Since many proofs in this work are
  mostly opaque to me, I can live with that.}
  
  { 
  fix A B 
  assume A: nonempty-graph A finite-graph A 
  assume iso: A ≃ B 

  then obtain f where f: B = map-ugraph f A bij_betw f (uverts A) (uverts B)
  unfolding isomorphism-def by blast
  have wellformed: uwellformed A
  using iso unfolding isomorphism-def by simp
  — We observe that the set of densities of the subgraphs does not change if we
  map the subgraphs first.
  have density ' nonempty-subgraphs A = density ' (map-ugraph f ' nonempty-subgraphs A)
  proof (rule image-comp-cong)
fix \( G \)

\begin{align*}
\text{assume} & \quad G \in \text{nonempty-subgraphs} \ A \\
\text{hence} & \quad \text{uverts} \ G \subseteq \text{uverts} \ A \ \text{uwellformed} \ G \\
\text{hence} & \quad \text{unfolding} \ \text{nonempty-subgraphs-def} \ \text{subgraph-def} \ \text{by} \ \text{simp+} \\
\text{hence} & \quad \text{inj-on} \ f \ (\text{uverts} \ G) \\
& \quad \text{using} \ f(2) \ \text{unfolding} \ \text{bij-betw-def} \ \text{by} \ (\text{metis subset-inj-on}) \\
\text{hence} & \quad G \simeq \text{map-ugraph} \ f \ G \\
\text{unfolding} & \quad \text{isomorphism-def} \ \text{bij-betw-def} \\
& \quad \text{by} \ (\text{metis map-ugraph.simps} \ \text{fst-conv} \ \text{surjective-pairing} \ \text{map-ugraph-wellformed}) \\
& \quad \text{thus} \ \text{density} \ G = \text{density} \ (\text{map-ugraph} \ f \ G) \\
& \quad \text{by} \ (\text{fact isomorphic-density}) \\
\text{qed}
\end{align*}

— Additionally, we show that the operations \text{nonempty-subgraphs} and \text{map-ugraph}
can be swapped without changing the densities. This is an obvious result, because
\text{map-ugraph} does not change the structure of a graph. Still, the proof is a bit
hairy, which is why we only show inclusion in one direction and use symmetry of
isomorphism later.

\begin{align*}
\text{also have} & \quad \cdots \subseteq \text{density} \ ' \ \text{nonempty-subgraphs} \ (\text{map-ugraph} \ f \ A) \\
\text{proof} & \quad (\text{rule image-mono}, \ \text{rule subsetI}) \\
& \quad \text{fix} \ G'' \\
& \quad \text{assume} \ G'' \in \text{map-ugraph} \ f \ ' \ \text{nonempty-subgraphs} \ A \\
\text{then obtain} & \quad G' \ \text{where} \ G\text{-subst} : G'' = \text{map-ugraph} \ f \ G' \ G' \in \text{nonempty-subgraphs} \\
& \quad \text{by} \ \text{blast} \\
& \quad \text{hence} \ G' : \ \text{subgraph} \ G' \ A \ \text{nonempty-graph} \ G' \ \text{uwellformed} \ G' \\
& \quad \text{unfolding} \ \text{nonempty-subgraphs-def} \ \text{by} \ \text{auto} \\
& \quad \text{hence} \ \text{inj-on} \ f \ (\text{uverts} \ G') \\
& \quad \text{using} \ f \ \text{unfolding} \ \text{bij-betw-def} \ \text{subgraph-def} \ \text{by} \ (\text{metis subset-inj-on}) \\
& \quad \text{hence} \ \text{uwellformed} \ G'' \\
& \quad \text{using} \ \text{map-ugraph-wellformed} \ G' \ G\text{-subst} \ \text{by} \ \text{simp} \\
& \quad \text{moreover have} \ \text{nonempty-graph} \ G'' \\
& \quad \text{using} \ G' \ G\text{-subst} \ \text{unfolding} \ \text{nonempty-graph-def} \ \text{by} \ (\text{metis map-ugraph.simps} \ \text{fst-cone} \ \text{snd-conv} \ \text{surjective-pairing} \ \text{empty-is-image}) \\
& \quad \text{moreover have} \ \text{subgraph} \ G'' \ (\text{map-ugraph} \ f \ A) \\
& \quad \text{using} \ \text{map-ugraph-preserves-sub} \ G' \ G\text{-subst} \ \text{by} \ \text{simp} \\
& \quad \text{ultimately show} \ G'' \in \text{nonempty-subgraphs} \ (\text{map-ugraph} \ f \ A) \\
& \quad \text{unfolding} \ \text{nonempty-subgraphs-def} \ \text{by} \ \text{simp} \\
\text{qed}
\end{align*}

\begin{align*}
\text{finally have} & \quad \text{density} \ ' \ \text{nonempty-subgraphs} \ A \subseteq \text{density} \ ' \ \text{nonempty-subgraphs} \\
& \quad (\text{map-ugraph} \ f \ A) \\
& \quad \text{hence} \ \text{max-density} \ A \leq \text{max-density} \ (\text{map-ugraph} \ f \ A) \\
& \quad \text{unfolding} \ \text{max-density-def} \\
\text{proof} & \quad (\text{rule Max-mono}) \\
& \quad \text{have} \ A \in \text{nonempty-subgraphs} \ A \\
& \quad \text{using} \ A \ \text{iso} \ \text{unfolding} \ \text{nonempty-subgraphs-def} \ \text{by} \ (\text{simp add: subgraph-refl}) \\
& \quad \text{thus} \ \text{density} \ ' \ \text{nonempty-subgraphs} \ A \neq \{\} \\
& \quad \text{by} \ \text{blast}
\end{align*}
next
  have finite (nonempty-subgraphs (map-ugraph f A))
  by (rule nonempty-subgraphs-finite[OF map-ugraph-finite[OF A(2)]])
  thus finite (density ' nonempty-subgraphs (map-ugraph f A))
  by blast
qed

hence max-density A ≤ max-density B
by (subst f)
}

note le = this

show ?thesis
  using le[OF assms(2) assms(4)] le[OF assms(3) assms(5) isomorphic-sym[OF assms(1)]]
  by (fact antisym)
qed

4.6 Fixed selectors

In the proof of the main theorem in the lecture notes, the concept of a “fixed
 copy” of a graph is fundamental.

Let $H$ be a fixed graph. A ‘fixed selector’ is basically a function mapping
a set with the same size as the vertex set of $H$ to a new graph which is
isomorphic to $H$ and its vertex set is the same as the input set.\footnotemark

\footnotetext{We call such a selector \textit{fixed} because its result is deterministic.}

\begin{definition}

is-fixed-selector $H$ $f$ = ($\forall V.\ finite\ V \land card\ (uverts\ H) = card\ V \to H \isom f\ V \land uverts\ (f\ V) = V$)

\end{definition}

Obviously, there may be many possible fixed selectors for a given graph.
First, we show that there is always at least one. This is sufficient, because
we can always obtain that one and use its properties without knowing exactly
which one we chose.

\begin{lemma}
ex-fixed-selector:
  assumes wellformed $H$ and finite-graph $H$
of obtains $f$ where is-fixed-selector $H$ $f$
\end{lemma}

proof

— I guess this is the only place in the whole work where we make use of a nifty
little HOL feature called \textit{SOME}, which is basically Hilbert’s choice operator. The
reason is that any bijection between the the vertex set of $H$ and the input set gives
rise to a fixed selector function. In the lecture notes, a specific bijection was defined,
but this is shorter and more elegant.

let $?bij = \lambda V.\ SOME\ g.\ bij-betw\ g\ (uverts\ H)\ V$

let $?f = \lambda V.\ map-ugraph\ (?bij\ V)\ H$

{ fix $V :: uvert$ set
  assume finite $V$ card (uverts $H$) = card $V$
  moreover have finite (uverts $H$)}
using assms unfolding finite-graph-def by simp
ultimately have bij-betw (?bij V) (uverts H) V
by (metis finite-same-card-bij someI-ex)
moreover hence *: uverts (?f V) = V ∧ uwellformed (?f V)
using map-ugraph-wellformed[OF assms(1)]
by (metis bij-betw-def map-ugraph.simps fst-conv surjective-pairing)
ultimately have **: H ∼ ?f V
unfolding isomorphism-def using assms(1) by auto
note * **
}
thus is-fixed-selector H ?f
unfolding is-fixed-selector-def by blast
qed

lemma fixed-selector-induced-subgraph:
assumes is-fixed-selector H f and card (uverts H) = card V and finite V
assumes sub: subgraph (f V) (induced-subgraph V G) and V: V ⊆ uverts G
and G: uwellformed G
shows H ⊑ G
proof –
have post: H ∼ f V uverts (f V) = V
using assms unfolding is-fixed-selector-def by auto
have H ⊆ f V
by (rule isomorphic-is-subgraph-isomorphic)
(simp add: post)
also have f V ⊆ induced-subgraph V G
by (rule subgraph-is-subgraph-isomorphic)
(auto simp: induced-wellformed[OF G V] post sub)
also have ... ⊆ G
by (rule subgraph-is-subgraph-isomorphic[OF induced-wellformed])
(auto simp: G V induced-is-subgraph(1)[OF V])
finally show H ⊆ G
qed

end

5 Classes and properties of graphs

theory Ugraph-Properties
imports
  Ugraph-Lemmas
  ../Girth-Chromatic/Girth-Chromatic
begin

A “graph property” is a set of graphs which is closed under isomorphism.
type-synonym ugraph-class = ugraph set
### Definition

**ugraph-property :: ugraph-class ⇒ bool where**

\[
\text{ugraph-property } C \equiv \forall G \in C. \forall G'. G \simeq G' \rightarrow G' \in C
\]

**Abbreviation**  
**prob-in-class :: (nat ⇒ real) ⇒ ugraph-class ⇒ nat ⇒ real where**

\[
\text{prob-in-class } p \ c \ n \equiv \text{probGn } p \ n \ (\lambda \text{es. edge-space.edge-ugraph } n \ \text{es} \in c)
\]

From now on, we consider random graphs not with fixed edge probabilities but rather with a probability function depending on the number of vertices. Such a function is called a "threshold" for a graph property iff

- for asymptotically larger probability functions, the probability that a random graph is an element of that class tends to 1 ("1-statement"), and
- for asymptotically smaller probability functions, the probability that a random graph is an element of that class tends to 0 ("0-statement").

### Definition

**is-threshold :: ugraph-class ⇒ (nat ⇒ real) ⇒ bool where**

\[
\text{is-threshold } c \ t \equiv \text{ugraph-property } c \land (\forall p. \text{nonzero-prob-fun } p \rightarrow (p \ll t \rightarrow \text{prob-in-class } p \ c \rightarrow 0) \land (t \ll p \rightarrow \text{prob-in-class } p \ c \rightarrow 1))
\]

### Lemma

**is-thresholdI [intro]:**

- Assumes **ugraph-property** c
- Assumes \((\forall p. [ \text{nonzero-prob-fun } p; p \ll t ] \implies \text{prob-in-class } p \ c \rightarrow 0)\)
- Assumes \((\forall p. [ \text{nonzero-prob-fun } p; t \ll p ] \implies \text{prob-in-class } p \ c \rightarrow 1)\)
- Shows **is-threshold** c t

Using assms unfolding is-threshold-def by blast

### 6 The subgraph threshold theorem

**Theory** Subgraph-Threshold

**Imports**

Ugraph-Properties

**Begin**

**Lemma** (in edge-space) measurable-pred[measurable]: Measurable.pred P Q

by (simp add: P-def sets-point-measure space-point-measure subset-eq)

This section contains the main theorem. For a fixed nonempty graph \( H \), we consider the graph property of ‘containing an isomorphic subgraph of \( H \)’. This is obviously a valid property, since it is closed under isomorphism. The corresponding threshold function is

\[
t(n) = n^{-\frac{1}{\rho[H]}},
\]

33
where $\rho'$ denotes max-density.

**definition** subgraph-threshold :: ugraph $\Rightarrow$ nat $\Rightarrow$ real where
subgraph-threshold $H$ n = n powr (-(1 / max-density $H$))

**theorem**
assumes nonempty: nonempty-graph $H$ and finite: finite-graph $H$ and well-formed: wellformed $H$
shows is-threshold \{ $G$, $H$ $\subseteq$ $G$ \} (subgraph-threshold $H$)
proof
unfolding ugraph-property \{ $G$, $H$ $\subseteq$ $G$ \} using subgraph-isomorphic-closed by blast
next

— To prove the 0-statement, we introduce the subgraph with the maximum density as $H_0$. Note that $\rho(H_0) = \rho'(H)$.

fix $p$ :: nat $\Rightarrow$ real

obtain $H_0$ where $H_0$: density $H_0$ = max-density $H$ subgraph $H_0$ $H$ nonempty-graph
$H_0$ finite-graph $H_0$ wellformed $H_0$
using subgraph-has-max-density assms by blast
hence card: 0 < card (uverts $H_0$) 0 < card (uedges $H_0$)
unfolding nonempty-graph-def finite-graph-def by auto

let $?v = card$ (uverts $H_0$)
let $?e = card$ (uedges $H_0$)

assume p-nz: nonzero-prob-fun $p$
hence $p$: valid-prob-fun $p$
by (fact nonzero-fun-is-valid-fun)

— Firstly, we follow from the assumption that $p$ is asymptotically less than the threshold function that the product

\[ p(n) \left| E(H_0) \right| \cdot n \left| V(H_0) \right| \]

tends to 0.

assume $p \ll$ subgraph-threshold $H$
moreover
\{ fix $n$
    have $p$ $n$ / $n$ powr (-(1 / max-density $H$)) = $p$ $n$ * $n$ powr (1 / max-density $H$)
    by (simp add: powr-minus-divide)
    also have \ldots $p$ $n$ * $n$ powr (1 / density $H_0$)
    using $H_0$ by simp
    also have \ldots $p$ $n$ * $n$ powr ($?v$ / $?e$)
    using card unfolding density-def by simp
    finally have $p$ $n$ / $n$ powr (-(1 / max-density $H$)) = \ldots
\}
ultimately have \((\lambda n. \, p \, n \, n \, powr \, (?v / ?e)) --- > 0\)

unfolding subgraph-threshold-def by simp
moreover have \(\wedge n. \, 1 \leq n \Longrightarrow 0 < p \, n \, n \, powr \, (?v / ?e)\)
by (auto simp: p-nz)
ultimately have \((\lambda n. \, (p \, n \, n \, powr \, (?v / ?e)) \, powr \, ?e) > 0\)

by unfolding subgraph-threshold-def

Moreover have \(\wedge n. \, I \leq n \Longrightarrow 0 < p \, n \, n \, powr \, (?v / ?e)\)
using card(2) by (force intro: tendsto-zero-powrI[OF eventually-sequentiallyI])
hence limit: \((\lambda n. \, p \, n \, n \, powr \, (?v / ?e) \, powr \, ?e) > 0\)
by (rule LIMSEQ-cong[OF - eventually-sequentiallyI[OF where c = 1]])
(auto simp: p card p-nz powr-powr powr-mult)

{ fix n
  assume n: ?v \leq n

interpret ES: edge-space n (p n)
by unfold-locales (auto simp: p)

let ?graph-of = ES.edge-uigraph

— After fixing an \(n\), we define a family of random variables \(X\) indexed by a set of vertices \(v\) and a set of edges \(e\). Each \(X\) is an indicator for the event that \((v, e)\) is isomorphic to \(H_0\) and a subgraph of a random graph. The sum of all these variables is denoted by \(Y\) and counts the total number of copies of \(H_0\) in a random graph.

let ?X = \(\lambda H_0'. \, \text{rind} \{ es \in \text{space ES.P. subgraph } H_0' \,(?graph-of es) \land H_0 \cong H_0'\}\)
let ?I = \{\((v, e). \, v \subseteq \{1..n\} \land \text{card } v = ?v \land e \subseteq \text{all-edges } v \land \text{card } e = ?e\}\)
let ?Y = \(\lambda es. \, \sum_{H_0' \in \, ?I. \, ?X \, H_0' \, es}\)

— Now we prove an upper bound for the probability that a random graph contains a copy of \(H\). Observe that in that case, \(Y\) takes a value greater or equal than 1.

have prob-in-class p { G. \(H \subseteq G\) } n = probGn p n (\lambda es. \, H \subseteq ?graph-of es)
by simp
also have \(\ldots \leq \text{probGn p n (\lambda es. 1 \leq ?Y es)}\)
proof (rule ES.finite-measure-mono, safe)
  fix es
  assume es: es \in space (MGn p n)

  assume H \subseteq ?graph-of es
  hence H_0 \subseteq ?graph-of es — since H_0 is a subgraph of H
  using H_0 by (fast intro: subgraph-isomorphic-pred-subgraph-closed)
  then obtain H_0' \where H_0'. subgraph H_0' (?graph-of es) H_0 \cong H_0'
  unfolding subgraph-isomorphic-def
  by blast

35
show \( 1 \leq \?Y \) es

proof (rule setsum-lower-or-eq)

— The only relevant step here is to provide the specific instance of \((v, e)\) such that \(X_{(v,e)}\) takes a value greater or equal than 1. This is trivial, as we already obtained that one above (i.e. \(H_0'\)). The remainder of the proof is just bookkeeping.

show \( 1 \leq \?X H_0'\) es — by definition of \(X\)

using \(H_0'\) es by simp

next

have "werts \(H_0'\) \(\subseteq\) \{1..\(n\}\) uedges \(H_0'\) \(\subseteq\) es

using \(H_0'(1)\) unfolding subgraph-def ES.edge-ugraph-def ES.S-verts-def ES.S-edges-def by simp+

moreover have "card \((werts H_0') = \?v\) card \((uedges H_0') = \?e\"

by (simp add: isomorphic-cards[OF \(H_0 \simeq H_0'\)])+

moreover have "uedges \(H_0'\) \(\subseteq\) all-edges \((werts H_0')\)

using \(H_0'\) by (simp add: wellformed-all-edges)

ultimately show \(H_0' \in \?I\)

by auto

next

have "\(?I \subseteq\) subgraphs \((\text{complete \{1..\(n\}\})\)"

unfolding complete-def subgraphs-def subgraph-def using all-edges-mono

by auto blast

moreover have "finite \((\text{subgraphs \{complete \{1..\(n\}\}})\)"

by (simp add: complete-finite subgraphs-finite)

ultimately show "finite \(?I\)"

by (fact finite-subset)

qed simp

qed simp

— Applying Markov’s inequality leaves us with estimating the expectation of \(Y\), which is the sum of the individual \(X\).

also have \(\ldots \leq ES.\text{expectation} \(?Y\) / 1\)

by (rule prob-space.markov-inequality) (auto simp: ES.prob-space-P setsum-nonneg)

also have \(\ldots = ES.\text{expectation} \(?Y\)\)

by simp

also have \(\ldots = (\sum \(H_0' \in \?I. ES.\text{expectation} \(\?X H_0'\))\)

by (rule integral-setsum(1)) simp

— Each expectation is bound by \(p(n)^{|E(H_0)|}\). For the proof, we ignore the fact that the corresponding graph has to be isomorphic to \(H_0\), which only increases the probability and thus the expectation. This only leaves us to compute the probability that all edges are present, which is given by edge-space.cylinder-prob.

also have \(\ldots \leq (\sum \(H_0' \in \?I. p n ^ \?e\))\)

proof (rule setsum-mono)

fix \(H_0'\)

assume \(H_0': H_0' \in \?I\)

have "ES.\text{expectation} \(\?X H_0'\) = ES.\text{prob} \{es \in \text{space ES.P. subgraph \(H_0'\) \(\{\text{graph-of es}\} \land H_0 \simeq H_0'\})\}"

36
by (rule \texttt{ES.expectation-indicator}) (auto simp: ES.sets-eq ES.space-eq)
also have \ldots \leq ES.prob \{ es \in \text{space } ES.P. \text{ uedges } H_0' \subseteq es \}
unfolding subgraph-def by (rule \texttt{ES.finite-measure-mono}) (auto simp: ES.sets-eq ES.space-eq)
also have \ldots = ES.prob \{ es \in \text{space } ES.P. \text{ uedges } H_0' \subseteq es \}
proof (rule \texttt{ES.cylinder-empty-prob})
  have uverts H_0' \subseteq \{1..n\} uedges H_0' \subseteq \text{all-edges } (\text{uverts } H_0')
    using H_0' by auto
  hence uedges H_0' \subseteq \text{all-edges } \{1..n\}
    using all-edges-mono by blast
  thus uedges H_0' \subseteq \text{ES.S-edges}
    unfolding ES.S-edges-def ES.S-verts-def by simp
qed
also have \ldots = p n ^ ?e
  unfolding cylinder-def ES.space-eq by simp
proof (rule \texttt{ES.cylinder-empty-prob})
  have uverts H_0' \subseteq \{1..n\} uedges H_0' \subseteq \text{all-edges } (\text{uverts } H_0')
    using H_0' by auto
  hence uedges H_0' \subseteq \text{all-edges } \{1..n\}
    using all-edges-mono by blast
  thus uedges H_0' \subseteq \text{ES.S-edges}
    unfolding ES.S-edges-def ES.S-verts-def by simp
qed
also have \ldots = p n ^ ?e
  unfolding \texttt{real-of-nat-def} by (rule setsum-constant)
— We have to count the number of possible pairs \((v, e)\). From the definition of the index set, note that we first choose \(|V(H_0)|\) elements out of a set of \(n\) vertices and then \(|E(H_0)|\) elements out of all possible edges over these vertices.
also have \ldots = (\binom{n}{\text{？v}} * ((\binom{\text{？v} \choose 2} \text{ choose } ?e)) * p n ^ ?e
proof (rule arg-cong[where \(x = \text{card } ?I\)])
  have card ?I = (\sum v | v \subseteq \{1..n\} \land \text{card } v = ?v. \text{ card } (\text{all-edges } v) \text{ choose } \text{？e})
    by (rule \texttt{card-dep-pair-set}[where \(A = \{1..n\} \land n = ?v \text{ and } f = \text{ all-edges}\])
    (auto simp: \texttt{finite-subset all-edges-finite})
  also have \ldots = (\sum v | v \subseteq \{1..n\} \land \text{card } v = ?v. (\text{？v choose 2} \text{ choose } ?e)
proof (rule \texttt{setsum.cong})
  fix v
  assume v \in \{v. v \subseteq \{1..n\} \land \text{card } v = ?v\}
  hence v \subseteq \{1..n\} \text{ card } v = ?v
    by auto
  thus \text{card } (\text{all-edges } v) \text{ choose } ?e = (\text{？v choose 2} \text{ choose } ?e)
    by (simp add: \texttt{card-all-edges finite-subset})
  qed rule
  also have \ldots = \text{card } (\{v. v \subseteq \{1..n\} \land \text{card } v = ?v\}) * ((\text{？v choose 2} \text{ choose } ?e)
proof (simp add: \texttt{n-subsets})
37
finally show card ?I = ...

qed
also have ... = (n choose ?v) * (((?v choose 2) choose ?e) * p n ~ ?e)
by simp

— Here, we use n^k as an upper bound for \( \binom{n}{k} \).
also have ... \leq (n \choose ?v) * (((?v choose 2) choose ?e) * p n \choose ?v) (is - \leq - * ?r)

proof (rule mult-right-mono)
  have n choose ?v \leq n \choose ?v
  by (rule binomial-le-pow) (rule n)
  thus real (n choose ?v) \leq real (n \choose ?v)
  by (metis real-of-nat-le-iff)
next
  show 0 \leq ?r using p by simp
qed
also have ... \leq ((?v choose 2) choose ?e) * (p n \choose ?e * n \choose ?v) (is - \leq ?factor * -)
by simp
also have ... = ?factor * (p n powr ?e * n powr ?v)
  using n card(1) :nonzero-prob-fun p; by (simp add: powr-realpow)

finally have prob-in-class p \{ G. H \subseteq G \} n \leq ?factor * (p n powr ?e * n powr ?v)
.

— The final upper bound is a multiple of the expression which we have proven to tend to 0 in the beginning.
  thus prob-in-class p \{ G. H \subseteq G \} ----> 0
  by (rule LIMSEQ-le-zero [OF tendsto-mult-right-zero [OF limit eventually-sequentiallyI [OF measure-nonneg eventually-sequentiallyI]]])
next
  fix p :: nat \Rightarrow real
  assume p-threshold: subgraph-threshold H \ll p

— To prove the 1-statement, we obtain a fixed selector f as defined in section 4.6.
from assms obtain f where f: is-fixed-selector H f
  using ex-fixed-selector by blast

let ?v = card (uverts H)
let ?e = card (uedges H)

— We observe that several terms involving |V(H)| are positive.
  have v-e-nz: 0 < real ?v 0 < real ?e
    using nonempty finite unfolding nonempty-graph-def finite-graph-def by auto
  hence 0 < real ?v ~ ?v by simp

38
— For a given \( n \), let \( A \) be a family of events indexed by a set \( S \). Each \( A \) contains the graphs whose induced subgraphs over \( S \) contain the selected copy of \( H \) by \( f \) over \( S \).

\[
\text{let } \mathcal{A} = \lambda n. \lambda S. \{ es \in \text{space (edge-space.} P \ n \ (p \ n)). \text{subgraph (} f \ S) \ (\text{induced-subgraph} \ S \ (\text{edge-space.edge-agraph n es}))\}\]

\[
\text{let } \mathcal{I} = \lambda n. \{ S. S \subseteq \{1..n\} \land \text{card } S = ?v\}
\]

\begin{itemize}
\item \textbf{assume} \( p\nspace{+}nz\): \textit{nonzero-prob-fun} \( p \)
\item \textbf{hence} \( p\nspace{+}n\): \textit{valid-prob-fun} \( p \)
\begin{itemize}
\item by (\textit{fact nonzero-fun-is-valid-fun})
\end{itemize}
\end{itemize}

\begin{itemize}
\item \textbf{fix} \( n \)
\end{itemize}

— At this point, we can assume almost anything about \( n \): We only have to show that a function converges, hence the necessary properties are allowed to be violated for small values of \( n \).

\begin{itemize}
\item \textbf{assume} \( n\nspace{+}2v\): \( 2 \times ?v \leq n \)
\item \textbf{hence} \( n\nspace{+}n\): \( ?v \leq n \)
\end{itemize}

\begin{itemize}
\item by simp
\end{itemize}

\begin{itemize}
\item \textbf{have} \( \text{is-es: edge-space (} p \ n) \)
\item by \textit{unfold-locales (auto simp: p)}
\end{itemize}

\textbf{then interpret} \( \text{edge-space n p n} \)

\begin{itemize}
\item let \( \mathcal{A} = ?A \ n \)
\item let \( \mathcal{I} = ?I \ n \)
\end{itemize}

— A nice potpourri with some technical facts about \( S \).

\begin{itemize}
\item \textbf{fix} \( S \)
\item \textbf{assume} \( S \in \mathcal{I} \)
\item \textbf{hence} \( 0\nspace{+}S\subseteq \{1..n\} \ ?v = \text{card } S \ \text{finite} \)
\item by (\textit{auto intro: finite-subset})
\item \textbf{hence} \( 1\nspace{+}H \simeq f \ S \text{ werts (} f \ S) = S \)
\item using \( f\text{ wellformed-finite unfolding finite-graph-def is-fixed-selector-def by auto} \)
\item \textbf{have} \( 2\nspace{+}f\text{inite-graph (} f \ S) \)
\item \textbf{using} \( 0(3) \ (1,2) \text{ by (metis wellformed-finite)} \)
\item \textbf{have} \( 3\nspace{+}nonempty-graph (} f \ S) \)
\item \textbf{using} \( 0(2) \ (1,2) \text{ by (metis card-eq-0-iff finite finite-graph-def isomorphic-cards(2) nonempty nonempty-graph-def pair-collapse snd-conv)} \)
\item \textbf{note} \( 0 \ 1 \ 2 \ 3 \)
\end{itemize}

\begin{itemize}
\item note \( I = \text{this} \)
\end{itemize}

— In the following two blocks, we prove the probabilities of the events \( A \) and
the probability of the intersection of two events $A$. For both cases, we employ the auxiliary lemma \textit{edge-space} \textit{induced-subgraph-prob} which is not very interesting. For the latter however, the tricky part is to argue that such an intersection is equivalent to the union of the desired copies of $H$ to be contained in the union of the induced subgraphs.

$$\{
\text{fix } S
\text{ assume } S \in ?I
\text{ note } S' = I[\text{OF } S]
\text{ have } \text{prob} (\forall A \ S) = p \ n \ \hat{e}
\text{ using isomorphic-cards(2)}(\text{OF } S'(4)) S' \text{by (simp add: } S\text{-verts-def induced-subgraph-prob})
\}
\text{ note } \text{prob-A} = \text{this}
$$

$$\{
\text{fix } S \ T
\text{ assume } S \in ?I \text{ note } S = I[\text{OF this}]
\text{ assume } T \in ?I \text{ note } T = I[\text{OF this}]
\text{ — Note that we do not restrict } S \text{ and } T \text{ to be disjoint, since we need the general case later to determine when two events are independent. Additionally, it would be unneeded at this point.}
\text{ have } \text{prob} (\forall A \ S \cap \forall A \ T) = \text{prob} \{es \in \text{space } P. \ \text{subgraph} (S \cup T, \ \text{uedges} (f \ S) \cup \text{uedges} (f \ T)) (\text{induced-subgraph} (S \cup T) \ (\text{edge-ugraph} es))\} (\text{is} - = \text{prob } ?M)
\text{ proof (rule arg-cong[where } f = \text{prob})}
\text{ have } \forall A \ S \cap \forall A \ T = \{es \in \text{space } P. \ \text{subgraph} (f \ S) (\text{induced-subgraph} S (\text{edge-ugraph es})) \land \text{subgraph} (f \ T) (\text{induced-subgraph} T (\text{edge-ugraph es})))\}
\text{ by blast}
\text{ also have } \ldots = ?M
\text{ using } S \ T \text{ by (auto simp: subgraph-union-induced)}
\text{ finally show } \forall A \ S \cap \forall A \ T = \ldots
\}
\text{ qed}
\text{ also have } \ldots = p \ n \ \hat{e} \text{card} (\text{uedges} (S \cup T, \ \text{uedges} (f \ S) \cup \text{uedges} (f \ T)))
\text{ proof (rule induced-subgraph-prob)}
\text{ show } \text{uwellformed} (S \cup T, \ \text{uedges} (f \ S) \cup \text{uedges} (f \ T))
\text{ using } S(4,5) \ T(4,5) \text{ unfolding } \text{uwellformed-def by auto}
\text{ next}
\text{ show } S \cup T \subseteq S\text{-verts}
\text{ using } S(1) \ T(1) \text{ unfolding } S\text{-verts-def by simp}
\text{ qed simp}
\text{ also have } \ldots = p \ n \ \hat{e} \text{card} (\text{uedges} (f \ S) \cup \text{uedges} (f \ T))
\text{ by simp}
\text{ finally have } \text{prob} (\forall A \ S \cap \forall A \ T) = p \ n \ \hat{e} \text{card} (\text{uedges} (f \ S) \cup \text{uedges} (f \ T))
\}
\text{ note } \text{prob-A-intersect} = \text{this}
— Another technical detail is that our family of events $A$ are a valid instantiation for the “$\Delta$ lemmas” from section 3.3.

**have** \textit{is-psi}: \textit{prob-space-with-indicators $P$ ?I ?A}

**proof**

\hspace{1em} \textit{show finite ?I}

\hspace{2em} \textit{by (rule finite-subset[where $B = \text{Pow \{1..n\}}$]) auto}

**next**

\hspace{1em} \textit{show $\neg A \vDash \neg I \subseteq \text{sets } P$}

\hspace{2em} \textit{unfolding sets-eq space-eq by blast}

**next**

\hspace{1em} \textit{let $?V = \{1..?v\}$}

\hspace{2em} \textit{have $\neg V < \text{prob (}\neg A ?V\)}$

\hspace{3em} \textit{by (simp add: prob-A n p-nz)}

\hspace{2em} \textit{moreover have $?V \in \neg I$}

\hspace{3em} \textit{using $n$ by force}

\hspace{2em} \textit{ultimately show $\exists i \in \neg I. \neg V < \text{prob (}\neg A i\)}$

\hspace{4em} \textit{by blast}

**qed**

**then interpret** \textit{prob-space-with-indicators $P$ ?I ?A}

— We proceed by reducing the claim of the 1-statement that the probability tends to 1 to showing that the expectation that the sum of all indicators of the respective events $A$ tends to 0. (The actual reduction is done at the end of the proof, we merely collect the facts here.)

**have** \textit{compl-prob}: \textit{1 - \text{prob \{es $\in$ space $P$. $\neg H \subseteq u\text{-ugraph es} \} = \text{prob-in-class p \{G. H \subseteq G\} n}$}

\hspace{2em} \textit{by (subst prob-compl[symmetric]) (auto simp: space-eq sets-eq intro: arg-cong[where \textit{f = prob}])}

**have** \textit{prob \{es $\in$ space $P$. $\neg H \subseteq u\text{-ugraph es} \} \leq \text{prob \{es $\in$ space $P$. Y es = 0\} (is $\neg$compl $\leq$ -)}$}

**proof** \textit{(rule finite-measure-mono, safe)}

\hspace{2em} \textit{fix es}

\hspace{3em} \textit{assume es $\in$ space $P$}

\hspace{4em} \textit{hence es: uwellformed (edge-ugraph es)}

\hspace{5em} \textit{unfolding space-eq by (rule wellformed-and-finite(2))}

\hspace{6em} \textit{assume H: $\neg H \subseteq u\text{-ugraph es}$}

\hspace{7em} \textit{\{}

\hspace{8em} \textit{fix S}

\hspace{9em} \textit{assume S $\subseteq \{1..n\}$ card S = $?v$}

\hspace{10em} \textit{moreover hence finite S S $\subseteq$ uverts (edge-ugraph es)}

\hspace{11em} \textit{unfolding uverts-edge-ugraph S-verts-def by (auto intro: finite-subset)}

\hspace{12em} \textit{ultimately have $\neg$ subgraph (f S) (induced-subgraph S (edge-ugraph es))}

\hspace{13em} \textit{using H es by (metis fixed-selector-induced-subgraph[OF f])}

\hspace{14em} \textit{hence X S es = 0}

\hspace{15em} \textit{unfolding X-def by simp}

41
\[
\text{thus } Y \varepsilon = 0 \\
\text{unfolding } Y\text{-def by simp} \\
\text{qed simp}
\]

— By applying the $\Delta$ lemma, we obtain our central inequality. The rest of the proof gives bounds for $\mu$, $\Delta_d$ and quotients which occur on the right hand side.

\text{hence } \text{compl-upper: } ?\text{compl} \leq 1 / \mu + \Delta_d / \mu^{-2}

by (rule order-trans) (fact prob-$\mu$-$\Delta_d$)

— Lower bound for the expectation. We use \( \binom{n}{k}^k \) as lower bound for \( \binom{n}{k} \).

\text{have } 1 / ?v \cdot ?v \ast (\text{real } n \cdot ?v \ast p n \cdot ?e) = (n / ?v) \cdot ?v \ast p n \cdot ?e

by (simp add: power-divide)

\text{also have } \ldots \leq (n \choose ?v) \ast p n \cdot ?e

\text{unfolding real-of-nat-def}

\text{proof (rule mult-right-mono, rule binomial-ge-n-over-k-pow-k)}

\text{show } ?v \leq n

\text{using } n .

\text{show } 0 \leq p n \cdot ?e

\text{using } p \text{ by simp}

\text{qed}

\text{also have } \ldots = (\sum S \in ?I. p n \cdot ?e)

\text{unfolding real-of-nat-def by (simp add: n-subsets)}

\text{also have } \ldots = (\sum S \in ?I. \text{prob } (?A S))

\text{by (simp add: prob-A)}

\text{also have } \ldots = \mu

\text{unfolding expectation-$X$-$Y$ X-def using expectation-indicator by force}

\text{finally have ex-lower: } 1 / (?v \cdot ?v) \ast (\text{real } n \cdot ?v \ast p n \cdot ?e) \leq \mu

— Upper bound for the inverse expectation. Follows trivially from above.

\text{have ex-lower-pos: } 0 < 1 / (?v \cdot ?v) \ast (\text{real } n \cdot ?v \ast p n \cdot ?e)

\text{proof (rule mult-pos-pos[OF vpowz-inv-gr-zero mult-pos-pos])}

\text{have } 0 < \text{real } n

\text{using } n \text{ nonempty finite unfolding nonempty-graph-def finite-graph-def}

\text{by auto}

\text{thus } 0 < \text{real } n \cdot ?v

\text{by simp}

\text{next}

\text{show } 0 < p n \cdot \text{card } (\text{uedges } H)

\text{using } p-nz \text{ by simp}

\text{qed}

\text{hence } 1 / \mu \leq 1 / (1 / ?v \cdot ?v \ast (\text{real } n \cdot ?v \ast p n \cdot ?e))

by (rule divide-left-mono[OF ex-lower zero-le-one mult-pos-pos[OF $\mu$-non-zero]])

\text{hence inv-ex-upper: } 1 / \mu \leq ?v \cdot ?v \ast (1 / (\text{real } n \cdot ?v \ast p n \cdot ?e))

\text{by simp}
— Recall the definition of $\Delta_d$:

$$\Delta_d = \sum_{S \in I, T \in I, S \neq T, A_S, A_T \text{ not independent}} \Pr[A_S \cap A_T]$$

We are going to prove an upper bound for that sum, so we can safely augment the index set by replacing it with a necessary condition.

The idea is that if the two sets $S$ and $T$ are not independent, their intersection is not empty. We prove that by contraposition, i.e. if the intersection is empty, then they are independent. This in turn can be shown using some basic properties of $f$.

```plaintext
{ fix S T assume S ∈ ?I T ∈ ?I hence prob (?A S) * prob (?A T) = p n ^ (2 * ?e) using prob-A by (simp add: power-even-eq power2-eq-square)

note S = I[OF ⟨S ∈ ?I⟩]
note T = I[OF ⟨T ∈ ?I⟩]
assume disj: S ∩ T = {}

have prob (?A S ∩ ?A T) = p n ^ card (uedges (f S) ∪ uedges (f T)) using ⟨S ∈ ?I⟩ ⟨T ∈ ?I⟩ by (fact prob-A-intersect)
also have ... = p n ^ (card (uedges (f S)) + card (uedges (f T))))
proof (rule arg-cong [OF card-Un-disjoint])

have finite-graph (f S) finite-graph (f T)
  using S T by (auto simp: wellformed-finite)
thus finite (uedges (f S)) finite (uedges (f T)) unfolding finite-graph-def by auto

next have uedges (f S) ⊆ all-edges S uedges (f T) ⊆ all-edges T
  using S(4,5) T(4,5) by (metis wellformed-all-edges+)
moreover have all-edges S ∩ all-edges T = {}
  by (fact all-edges-disjoint[OF disj])
ultimately show uedges (f S) ∩ uedges (f T) = {}
  by blast
qed

also have ... = p n ^ (2 * ?e)
using isomorphic-cards(2)[OF isomorphic-sym[OF S(4)] isomorphic-cards(2)][OF
isomorphic-sym[OF T(4)]] by (simp add: mult-2)
finally have **: prob (?A S ∩ ?A T) = ...
```

from ** have indep (?A S) (?A T)
  unfolding indep-def by force

```plaintext
note indep = this
```

— Now we prove an upper bound for $\Delta_d$.

have $\Delta_d = \sum_{S \in ?I} \sum_{T \in ?I} \Pr[A_S \cap A_T]$. prob (?A S ∩ ?A T)
unfolding $\Delta_d$-def ..

— Augmenting the index set as described above.

also have $\ldots \leq (\sum S \in I. \sum T \mid T \in I \land S \cap T \neq \emptyset). \text{prob} (?A S \cap ?A T))$

by (rule setsum-mono[OF setsum-mono3]) (auto simp: indep measure-nonneg)

— So far, we are adding the intersection probabilities over pairs of sets which have a nonempty intersection. Since we know that these intersections have at least one element (as they are nonempty) and at most $|V(H)|$ elements (by definition of $I$). In this step, we will partition this sum by cardinality of the intersections.

also have $\ldots = (\sum S \in I. \sum T \in (\bigcup k \in \{1..v\}. \{T \in I. \text{card} (S \cap T) = k\}). \text{prob} (?A S \cap ?A T))$

proof (rule setsum.cong, rule refl, rule setsum.cong)

fix $S$

assume $S \in I$

note $I(2,3)[OF this]$

hence $\{T. S \cap T \neq \emptyset\} = (\bigcup k \in \{1..v\}. \{T. \text{card} (S \cap T) = k\})$

by (simp add: partition-set-of-intersecting-sets-by-card)

thus $\{T \in I. S \cap T \neq \emptyset\} = (\bigcup k \in \{1..v\}. \{T \in I. \text{card} (S \cap T) = k\})$

by blast

qed simp

also have $\ldots = (\sum k = 1..v. \sum S \in I. \sum T \mid T \in I \land \text{card} (S \cap T) = k. \text{prob} (?A S \cap ?A T))$

by (rule setsum.cong, rule refl, rule setsum.UNION-disjoint) auto

also have $\ldots = (\sum S \in I. \sum T \mid T \in I \land \text{card} (S \cap T) = k. \text{prob} (?A S \cap ?A T))$

by (rule setsum.commute)

— In this step, we compute an upper bound for the intersection probability and argue that it only depends on the cardinality of the intersection.

also have $\ldots \leq (\sum k = 1..v. \sum S \in I. \sum T \mid T \in I \land \text{card} (S \cap T) = k. p n powr (2 * v - \text{max-density} H * k))$

proof (rule setsum-mono)+

fix $k$

assume $k: k \in \{1..v\}$

fix $S T$

assume $S \in I. T \in \{T. T \in I \land \text{card} (S \cap T) = k\}$

hence $T \in I \land ST-k: \text{card} (S \cap T) = k$

by auto

note $S = I[OF : S \in I]$

note $T = I[OF : T \in I]$ 

let $?cST = \text{card} (\text{uwedges (f S) \cap uwedges (f T)})$

— We already know the intersection probability.

have $\text{prob} (?A S \cap ?A T) = p n \times \text{card} (\text{uwedges (f S) \cup uwedges (f T)})$

using $S \in I \land T \in I$ by (fact prob-A-intersect)
— Now, we consider the number of edges shared by the copies of $H$ over $S$ and $T$.

**also have** \(\cdots \Rightarrow (\text{card}(\text{uedges}(f\ S)) + \text{card}(\text{uedges}(f\ T)) - ?cST)\)

*using* \(S\ T\)  *unfolding* \(\text{finite-graph-def}\) *by* \((\text{simp add: card-union})\)

**also have** \(\cdots \Rightarrow (?e + ?e - ?cST)\)

*by* \((\text{metis isomorphic-cards}(2)[\text{OF}\ S(4)]\) isomorphic-cards(2)[\text{OF}\ T(4)])\)

**also have** \(\cdots \Rightarrow (2 * ?e - ?cST)\)

*by* \((\text{simp add: mult-2})\)

**also have** \(\cdots \Rightarrow (2 * ?e - ?cST)\)

*using* \(\text{p-nz}\) *by* \((\text{simp add: pow-realpow})\)

**also have** \(\cdots \Rightarrow (\text{real} (2 * ?e) - \text{real} ?cST)\)

*using* \(\text{isomorphic-cards}[\text{OF}\ S(4)]\) \(S(6)\) *by* \((\text{metis real-of-nat-diff card-mono finite-graph-def inf-le1 mult-le-mono mult-numeral-1 numeral-1-eq-1 one-le-numeral})\)

— Since the intersection graph is also an isomorphic subgraph of $H$, we know that its density has to be less than or equal to the maximum density of $H$. The proof is quite technical.

**also have** \(\cdots \Rightarrow \text{p powr} (2 * ?e - \text{max-density} \ H \ast k)\)

*proof* \((\text{rule powr-mono3})\)

**have** \(?cST = \text{density} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T)) \ast k\)

*unfolding* \(\text{density-def}\) *using* \(k\ ST-k\) *by* \(\text{simp}\)

**also have** \(\cdots \Rightarrow \text{max-density} (f\ S) \ast k\)

*proof* \((\text{rule mult-right-mono, cases uedges (f S) \cap uedges (f T) = {}})\)

**case** \(\text{True}\)

*hence* \(\text{density} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T)) = 0\)

*unfolding* \(\text{density-def}\) *by* \(\text{simp}\)

**also have** \(0 \leq \text{density} (f\ S)\)

*unfolding* \(\text{density-def}\) *by* \(\text{simp}\)

**also have** \(\text{density} (f\ S) \leq \text{max-density} (f\ S)\)

*using* \(S\) *by* \((\text{simp add: max-density-is-max subgraph-refl})\)

**finally show** \(\text{density} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T)) \leq \text{max-density} (f\ S)\)

, then

**next**

**case** \(\text{False}\)

**show** \(\text{density} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T)) \leq \text{max-density} (f\ S)\)

*proof* \((\text{rule max-density-is-max})\)

**show** \(\text{finite-graph} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T))\)

*using* \(T(4,5)\) *by* \((\text{metis finite-Int finite-graph-def fst-eqD snd-conv})\)

**show** \(\text{nonempty-graph} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T))\)

*unfolding* \(\text{nonempty-graph-def}\) *using* \(k\ ST-k\ False\) *by* \(\text{force}\)

**show** \(\text{uwellformed} (S \cap T, \text{uedges}(f\ S) \cap \text{uedges}(f\ T))\)

*using* \(S(4,5)\) *by* \((\text{metis fst-eqD inf-sup-ord(1) snd-conv subgraph-def})\)

**qed** *by* \((\text{simp add: S})\)
and bounds for the binomial coefficients as for the 0-statement.

\[
\begin{align*}
\text{finally have } & \ ?cST \leq \text{max-density } H \ast k \\
& \text{by linarith} \\
\text{finally show } & \text{prob } (\forall A \cap ?A T) \leq \ldots
\end{align*}
\]

Further rewriting the index sets.

\[
\begin{align*}
\text{also have } & \ldots = (\Sigma \{ k = 1..?v, \Sigma (S, T) \in (\Sigma \text{MA } S : ?I. \{ T \in ?I. \text{card } (S \cap T) = k \}) \}) \ast p n powr (2 \ast ?e - \text{max-density } H \ast k) \\
& \text{by (rule setsum.cong, rule refl, rule setsum.Sigma) auto} \\
\text{also have } & \ldots = (\Sigma \{ k = 1..?v. \text{card } (\Sigma \text{MA } S : ?I. \{ T \in ?I. \text{card } (S \cap T) = k \}) \}) \ast p n powr (2 \ast ?e - \text{max-density } H \ast k)) \\
\text{unfolding } & \text{real-of-nat-def by (rule setsum.cong) auto}
\end{align*}
\]

Here, we compute the cardinality of the index sets and use the same upper bounds for the binomial coefficients as for the 0-statement.

\[
\begin{align*}
\text{also have } & \ldots \leq (\Sigma \{ k = 1..?v. \ ?v \ast k \ast (\text{real } n \ast (2 \ast ?v - k)) \ast p n powr (2 \ast ?e - \text{max-density } H \ast k)) \\
& \text{by (rule setsum-mono)} \\
& \text{fix } k \\
& \text{assume } k: k \in \{1..?v\} \\
& \text{let } ?p = p n powr (2 \ast ?e - \text{max-density } H \ast k) \\
& \text{have } \text{card } (\Sigma \text{MA } S : ?I. \{ T \in ?I. \text{card } (S \cap T) = k \}) \ast \text{(is } ?lhs = -) \\
& \text{by simp} \\
& \text{also have } \ldots = (\Sigma \{ S \in ?I. \ (\?v choose k) \ast ((n \ast ?v) \choose (?v - k))) \\
& \text{using } n \ast k \text{ by (fastforce simp: card-set-of-intersecting-sets-by-card) auto} \\
& \text{also have } \ldots = (\text{n choose ?v} \ast ((\?v choose k) \ast ((n \ast ?v) \choose (?v - k))) \\
& \text{by (auto simp: n-subsets)} \\
& \text{also have } \ldots \leq n \ast ?v \ast ((?v choose k) \ast ((n \ast ?v) \choose (?v - k))) \\
& \text{using } n \ast \text{ by (simp add: binomial-le-pow)} \\
& \text{also have } \ldots \leq n \ast ?v \ast ?v \ast k \ast ((n \ast ?v) \choose (?v - k)) \\
& \text{using } k \text{ by (simp add: binomial-le-pow)} \\
& \text{also have } \ldots \leq n \ast ?v \ast ?v \ast k \ast n \ast (?)v - k) \\
& \text{using } n \ast ?v \ast k \ast ((?v + (?v - k)) \\
& \text{by (simp add: power-mono)} \\
& \text{also have } \ldots = (?v \ast k \ast n \ast (2 \ast ?v - k) \text{ (is } - = ?rhs)}
\end{align*}
\]
using $k$ by (simp add: mult-2)
finally have \(?lhs \leq \?rhs\)
.

hence real \(?lhs \leq \?rhs\)
by blast
moreover have \(0 \leq \?p\)
by simp
ultimately have \(?lhs \cdot \?p \leq \?rhs \cdot \?p\)
by (rule mult-right-mono)
also have \(\ldots = \?v \cdot k \cdot \text{(real n} \cdot(2 \cdot \?v - k)\cdot \?p)\)
by simp
finally show \(?lhs \cdot \?p \leq \ldots\).
.
qed

finally have \(\text{delta-upper: } \Delta_d \leq (\sum_{k=1}^{\infty} \?v \cdot k \cdot (\text{real n} \cdot(2 \cdot \?v - k)\cdot \?p \cdot n \text{ powr } (2 \cdot \?e - \text{max-density } H \cdot k))\))
.

— At this point, we have established all necessary bounds.

note is-es is-psi compl-prob compl-upper ex-lower ex-lower-pos inv-ex-upper

delta-upper
}

note facts = this

— Recall our central inequality. We now prove that both summands tend to 0. This is mainly an exercise in bookkeeping and real arithmetics as no intelligent ideas are involved.

have \((\lambda n. \frac{1}{\text{prob-space-with-indicators}}. \mu (\text{MGn p n}) (\?I n) (\?A n)) \longrightarrow \infty \text{ le-zero}\)
.

proof (rule LIMSEQ-le-zero)
have \((\lambda n. \frac{1}{\text{prob-space-with-indicators}}. \mu (\text{MGn p n}) (\?I n) (\?A n)) \longrightarrow \infty \text{ le-zero}|(\text{OF - eventually-sequentiallyI eventually-sequentiallyI})\)
fix \(n\)
show \(0 \leq \frac{1}{\text{prob-space-with-indicators}}. \mu (\text{MGn p n}) (\?I n) (\?A n)) \longrightarrow \infty \text{ le-zero}\)
using \(p\) by simp

assume \(n: 1 \leq n\)

have \(1 = (\text{real n} \cdot \?v \cdot p \cdot n \cdot \?e) = \frac{1}{\text{(real n powr } \?v \cdot p \cdot n \text{ powr } \?e)}\)
using \(p\-nz\) by (simp add: powr-realpow[symmetric])
also have \(\ldots = \text{real n powr } -(\text{real n powr } \?v \cdot p \cdot n \text{ powr } \text{real } ?e)\)
by (simp add: powr-minus-divide)
also have \(\ldots = (\text{real n powr } -(\text{?v / ?e})) \text{ powr } ?e \cdot (\text{p n powr } -1) \text{ powr } \?e\)
using \(v\-e\-nz\) by (simp add: powr-powr)
also have \(\ldots = (\text{real n powr } -(\text{?v / ?e}) \cdot p \cdot n \text{ powr } -1) \text{ powr } ?e\)
by (simp add: power-mult)
also have \(\ldots = (\text{real n powr } -(1 / (\text{?e / ?v})) \cdot p \cdot n \text{ powr } -1) \text{ powr } ?e\)
by simp
also have \(\ldots \leq (\text{real n powr } -(1 / \text{max-density } H) \cdot p \cdot n \text{ powr } -1) \text{ powr } \?e\)

47
apply (rule powr-mono2[OF - - mult-right-mono[OF powr-mono[OF le-imp-neg-le[OF divide-left-mono]]]]))

  using n v-e-nz p p-nz
  by (auto simp:
    max-density-is-max[unfolded density-def, OF finite finite nonempty wellformed subgraph-refl]
    max-density-gr-zero[OF finite nonempty wellformed])

also have ... = (real n powr -(1 / max-density H) * (1 / p n powr 1))

powr ?e

  by (simp add: powr-minus-divide[ symmetric])

also have ... = (subgraph-threshold H n / p n) powr ?e

unfolding subgraph-threshold-def ..

finally show 1 / (real n ^ ?v * p n ^ ?e) ≤ (subgraph-threshold H n / p n) powr ?e

.

next

  show (λn. (subgraph-threshold H n / p n) powr real (card (uedges H)))
  ----> 0
  by auto fastforce

qed

hence (λn. ?v ^ ?v * (1 / (real n ^ ?v * p n ^ ?e)))) ----> 0

  by (rule LIMSEQ-const-mult)

thus (λn. ?v ^ ?v * (1 / (real n ^ ?v * p n ^ ?e)))) ----> 0

  by simp

next

  show ∀∞n. 0 ≤ 1 / prob-space-with-indicators.µ (MGn p n) (?I n) (?A n)

  by (rule eventually-sequentiallyI[OF less-imp-le[OF divide-pos-pos]OF - prob-space-with-indicators.µ-non-zero[OF facts(2)]]]) simp+

next

  show ∀∞n. 1 / prob-space-with-indicators.µ (MGn p n) (?I n) (?A n) ≤ ?v ^ ?v * (1 / (real n ^ ?v * p n ^ ?e))

  using facts(7) by (rule eventually-sequentiallyI)

qed

moreover have (λn. prob-space-with-indicators.∆t (MGn p n) (?I n) (?A n)) << (λn. (prob-space-with-indicators.µ (MGn p n) (?I n) (?A n)) ^2)

proof (rule less-fun-bounds)

let ?num = λn k. ?v ^ k * (real n ^ (2 * ?v - k) * p n powr (2 * ?e - max-density H * k))


— We have to show that a sum is asymptotically smaller than a constant term. We do that by showing that each summand is asymptotically smaller than
the term.

\[
\begin{align*}
\{ & \text{fix } k \\
& \text{assume } k \in \{1..v\} \\
& \text{let } \delta n' = \lambda n. (1 / (v - v) - 2) \cdot (\text{real } n - (2 \cdot v) \cdot p \cdot n - (2 \cdot e)) \\
& \text{have } \delta n' = \delta n \\
& \text{by subst power-mult-distinct} (\text{simp add: power-mult-distinct proves even-odd}) \\
& \text{have } (\lambda n. (?num n k) \ll ?den') \\
& \text{proof (rule less-fan-const-quot)} \\
& \text{have } (\lambda n. (\text{subgraph-threshold } H \cdot n / p \cdot n) \cdot \text{powr } (\text{max-density } H * k)) \\
& \text{----- > 0} \\
& \text{using tendsto-zero-power \OOF eventually-sequentially \OOF divide-pos-pos} \\
& \text{p-threshold mult-pos-pos[\OOF max-density-gr-zero \OOF finite nonempty wellformed]]} \\
& \text{subgraph-threshold-def powr-law-zero p-nz k} \\
& \text{apply (simp add Suc-le-eq)} \\
& \text{using real-of-nat-gr-zero-cancel-iff by blast} \\
& \text{thus } (\lambda n. (\text{real } n - (2 \cdot v - k) \cdot p \cdot n \cdot \text{powr } (2 \cdot e - \text{max-density } H * k)) / ((\text{real } n - (2 \cdot v) \cdot p \cdot n - (2 \cdot e))) \text{----- > 0} \\
& \text{proof (rule LIMSEQ-cong \OOF - eventually-sequentially[\OOF]} \\
& \text{fix } n :: \text{nat} \\
& \text{assume } n: 1 \leq n \\
& \text{have } ((\text{real } n - (2 \cdot v - k) \cdot p \cdot n \cdot \text{powr } (2 \cdot e - \text{max-density } H * k)) / ((\text{real } n - (2 \cdot v) \cdot p \cdot n - (2 \cdot e))) = \\
& \text{by (simp add power-mult-r) \\
& \text{also have } \ldots = (\text{n powr } (2 \cdot v - k) / \text{n powr } (2 \cdot v)) * (p \cdot n \cdot \text{powr } (2 \cdot e - \text{max-density } H * k)) / ((\text{n powr } (2 \cdot v) * (p \cdot n - (2 \cdot e)))) \\
& \text{by simp}} \text{also have } \ldots = \text{n powr } -\text{real } k * p \cdot n \cdot \text{powr } ((2 \cdot e - \text{max-density } H * k) - (2 \cdot e)) \text{----- > 0} \\
& \text{apply (rule arg-cong[where } y = - \text{real } k)] \\
& \text{using } k \text{ by fastforce} \\
& \text{also have } \ldots = \text{n powr } -\text{real } k * p \cdot n \cdot \text{powr } - (\text{max-density } H * k) \\
& \text{by simp} \\
& \text{also have } \ldots = (\text{n powr } - (1 / \text{max-density } H)) \cdot \text{powr } (\text{max-density } H * k) * p \cdot n \cdot \text{powr } - (\text{max-density } H * k) \\
& \text{using max-density-gr-zero \OOF finite nonempty wellformed} \text{ by (simp add: powr-mult)} \\
& \text{also have } \ldots = (\text{n powr } - (1 / \text{max-density } H)) \cdot \text{powr } (\text{max-density } H * k) * (p \cdot n \cdot \text{powr } - 1) \cdot \text{powr } (\text{max-density } H * k) \\
& \text{by (simp add: powr-mult)} \\
& \text{also have } \ldots = (\text{n powr } - (1 / \text{max-density } H)) * p \cdot n \cdot \text{powr } - 1) \cdot \text{powr } (\text{max-density } H * k) \\
& \text{by (simp add: powr-mult)} \\
\end{align*}
\]
also have \(\ldots = (n \ powr (1 \ / \ max-density H) \ * \ (1 \ / \ p \ n \ powr 1))\) 
powr (max-density \(H * k\)) 
by (simp add: powr-minus-divide[symmetric]) 
also have \(\ldots = (n \ powr -(1 \ / \ max-density H) \ / \ p \ n)\) powr 
(max-density \(H * k\)) 
by (simp add: p p-nz) 
also have \(\ldots = (\text{subgraph-threshold} \ H n / p n)\) powr (max-density 
\(H * k\)) (is - = ?rhs) 
unfolding subgraph-threshold-def .. 
finally have ?lhs = ?rhs 
thus ?rhs = ?lhs 
by simp 
qed

next 
show \((1 / (?v \ \div \ ?v)) \ ^2 \neq 0\) 
by (rule field-power-not-zero[OF less-imp-neq[symmetric]]) (rule 
vpowv-inv-gr-z) 
qed

hence \((\lambda n. \ ?num n k) \ll ?den\) 
by (rule subst[OF den']) 
}
hence \((\lambda n. \ ?num n k / ?den n) \longrightarrow (\sum k = 1..?v. 0)\) 
by (rule tendsto-setsum) 
hence \((\lambda n. \ ?num n k / ?den n) \longrightarrow 0\) 
by simp 
moreover have \((\lambda n. \ ?num n k / ?den n) = (\lambda n. (\sum k = 1..?v. \ ?num n k) / ?den n)\) 
by (simp add: setsum-left-div-distrib) 
ultimately show \((\lambda n. \ ?num n k) \ll ?den\) 
by metis 

show \(\forall \infty n. \ \text{prob-space-with-indicators}.\Delta_d (MGn p n) (\?I n) (\?A n) \leq (\sum k = 1..?v. \ ?num n k)\) 
using facts(8) by (rule eventually-sequentiallyI) 

show \(\forall \infty n. \ ?den n \leq (\text{prob-space-with-indicators}.\mu (MGn p n) (\?I n) (\?A n)) \ ^2\) 
using facts(5) facts(6) by (rule eventually-sequentiallyI[OF power-mono[OF 
- less-imp-le]]) 

show \(\forall \infty n. 0 \leq \text{prob-space-with-indicators}.\Delta_d (MGn p n) (\?I n) (\?A n)\) 
using facts(2) by (rule eventually-sequentiallyI[OF prob-space-with-indicators.D_d-nonneg]) 

show \(\forall \infty n. 0 < (\text{prob-space-with-indicators}.\mu (MGn p n) (\?I n) (\?A n)) ^{\ ^2}\) 
using facts(2) by (rule eventually-sequentiallyI[OF prob-space-with-indicators.\mu-sq-non-zero]) 

show \(\forall \infty n. 0 < ?den n\)
using facts(6) by (rule eventually-sequentiallyI[OF zero-less-power])

qed

ultimately have (λn).

1 / prob-space-with-indicators.µ (MGn p n) (?I n) (?A n) +
prob-space-with-indicators.Δd (MGn p n) (?I n) (?A n) / (prob-space-with-indicators.µ
(MGn p n) (?I n) (?A n))^2

) −−−−> 0
by (subst add-0-left[where a = 0, symmetric]) (rule tendsto-add)

— By now, we can actually perform the reduction mentioned above.
hence (λn. probGn p n (λes. ¬ H ⊑ edge-space.edge-ugraph n es)) −−−−> 0

proof (rule LIMSEQ-le-zero)

show ∀ ∞ n. 0 ≤ probGn p n (λes. ¬ H ⊑ edge-space.edge-ugraph n es)
by (rule eventually-sequentiallyI) (rule measure-nonneg)

next

show ∀ ∞ n.

probGn p n (λes. ¬ H ⊑ edge-space.edge-ugraph n es) ≤
1 / prob-space-with-indicators.µ (MGn p n) (?I n) (?A n) +
prob-space-with-indicators.Δd (MGn p n) (?I n) (?A n) / (prob-space-with-indicators.µ
(MGn p n) (?I n) (?A n))^2

by (rule eventually-sequentiallyI[OF facts(4)])

qed

hence (λn. 1 − probGn p n (λes. ¬ H ⊑ edge-space.edge-ugraph n es)) −−−−> 1

using tendsto-diff[OF tendsto-const] by fastforce

thus prob-in-class p (G. H ⊑ G) −−−−> 1
by (rule LIMSEQ-cong[OF - eventually-sequentiallyI[OF facts(3)]]

qed

end