Hypergraph Colouring Bounds using Probabilistic Methods

Chelsea Edmonds and Lawrence C. Paulson

April 18, 2024

Abstract

This library includes several example applications of the probabilistic method for combinatorics to establish bounds for hypergraph colourings. This focuses on *Property B* — the existence of a two-colouring of the vertex set of a hypergraph. A stricter bound was formalised using the Lovász local lemma, which in turn required a surprisingly complex proof of the mutual independence principle for hypergraph edges that is often omitted on paper. The formalisation uncovered several interesting examples of circular intuition on proofs involving independence on paper. The formalisation is based on the textbook proofs from Alon and Spencer's famous textbook, *The Probabilistic Method*[1], further supported by [3]. The mutual independence principle proof is inspired by the less precise proof provided in Molloy and Reed's textbook on graph colourings [2], as it was omitted in all other sources. Additionally, this library demonstrates how locales can be used to establish a reusable probability space framework, thus minimizing the setup required for future formalisations requiring a probability space on numerous possible properties around an incidence system's vertex set.

Contents

1 Hypergraph Colourings

theory *Hypergraph-Colourings* **imports** *Card-Partitions.Card-Partitions*
Hypergraph-Basics.Hypergraph-Variations HOL−Library.Extended-Real
Girth-Chromatic.Girth-Chromatic-Misc
begin

1.1 Function and Number extras

lemma *surj-PiE*:
  assumes $f \in A \to_E B$
  assumes $f ` A = B$
  assumes $b \in B$
  obtains a **where** $a \in A$ **and** $f\ a = b$
  ⟨*proof*⟩

lemma *Stirling-gt-0*: $n \geq k \Longrightarrow k \neq 0 \Longrightarrow Stirling\ n\ k > 0$
  ⟨*proof*⟩

lemma *card-partition-on-ne*:
  assumes $card\ A \geq n \quad n \neq 0$
  shows $\{P.\ partition\text{-}on\ A\ P \wedge card\ P = n\} \neq \{\}$
⟨*proof*⟩

lemma *enat-lt-INF*:
  fixes $f :: 'a \Rightarrow enat$
  assumes $(INF\ x \in S.\ f\ x) < t$
  obtains x **where** $x \in S$ **and** $f\ x < t$
⟨*proof*⟩

1.2 Basic Definitions

context *hypergraph*
begin

Edge colourings - using older partition approach

definition *edge-colouring* :: $('a\ hyp\text{-}edge \Rightarrow colour) \Rightarrow colour\ set \Rightarrow bool$ **where**
$edge\text{-}colouring\ f\ C \equiv partition\text{-}on\text{-}mset\ E\ \{\#\ \{\#h \in\#\ E\ .\ f\ h = c\#\}\ .\ c \in\# (mset\text{-}set\ C)\#\}$

definition *proper-edge-colouring* :: $('a\ hyp\text{-}edge \Rightarrow colour) \Rightarrow colour\ set \Rightarrow bool$
where

proper-edge-colouring f C ≡ *edge-colouring f C* ∧
(∀ *e1 e2 c*. *e1* ∈# *E* ∧ *e2* ∈# *E* − {#*e1*#} ∧ *c* ∈ *C* ∧ *f e1* = *c* ∧ *f e2* = *c* ⟶ *e1* ∩ *e2* = {})

A vertex colouring function with no edge monochromatic requirements

abbreviation *vertex-colouring* :: (*'a* ⇒ *colour*) ⇒ *nat* ⇒ *bool* **where**
vertex-colouring f n ≡ *f* ∈ 𝒱 $\rightarrow_E$ {*0*..<*n*}

lemma *vertex-colouring-union*:
assumes *vertex-colouring f n*
shows ⋃ {{*v* ∈ 𝒱. *f v* = *c*} |*c*. *c* ∈ {*0*..<*n*}} = 𝒱
⟨*proof*⟩

lemma *vertex-colouring-disj*:
assumes *vertex-colouring f n*
assumes *p* ∈ {{*v* ∈ 𝒱. *f v* = *c*} |*c*. *c* ∈ {*0*..<*n*}}
assumes *p'* ∈ {{*v* ∈ 𝒱. *f v* = *c*} |*c*. *c* ∈ {*0*..<*n*}}
assumes *p* ≠ *p'*
shows *p* ∩ *p'* = {}
⟨*proof*⟩

lemma *vertex-colouring-n0*: 𝒱 ≠ {} ⟹ ¬ *vertex-colouring f 0*
⟨*proof*⟩

lemma *vertex-colouring-image*: *vertex-colouring f n* ⟹ *v* ∈ 𝒱 ⟹ *f v* ∈ {*0*..<*n*}
⟨*proof*⟩

lemma *vertex-colouring-image-edge-ss*: *vertex-colouring f n* ⟹ *e* ∈# *E* ⟹ *f* ‘ *e* ⊆ {*0*..<*n*}
⟨*proof*⟩

lemma *vertex-colour-edge-map-ne*: *vertex-colouring f n* ⟹ *e* ∈# *E* ⟹ *f* ‘ *e* ≠ {}
⟨*proof*⟩

lemma *vertex-colouring-ne*: *vertex-colouring f n* ⟹ *f u* ≠ *f v* ⟹ *u* ≠ *v*
⟨*proof*⟩

lemma *vertex-colour-one*: 𝒱 ≠ {} ⟹ *vertex-colouring f 1* ⟹ *v* ∈ 𝒱 ⟹ *f v* = (*0*::*nat*)
⟨*proof*⟩

lemma *vertex-colour-one-alt*:
assumes 𝒱 ≠ {}
shows *vertex-colouring f* (*1*::*nat*) ⟷ *f* = (λ *v* ∈ 𝒱 . *0*::*nat*)
⟨*proof*⟩

lemma *vertex-colouring-partition*:
assumes *vertex-colouring f n*

assumes f ‘ $\mathcal{V} = \{0..<n\}$
shows *partition-on* $\mathcal{V}$ $\{ \{v \in \mathcal{V} . f\ v = c\} \mid c.\ c \in \{0..<n\}\}$
⟨*proof*⟩

1.3 Monochromatic Edges

definition *mono-edge* :: $('a \Rightarrow colour) \Rightarrow 'a\ hyp\text{-}edge \Rightarrow bool$ **where**
mono-edge f e $\equiv \exists\ c.\ \forall\ v \in e.\ f\ v = c$

lemma *mono-edge-single*:
assumes $e \in\#\ E$
shows *mono-edge f e* $\longleftrightarrow$ *is-singleton* (f ‘ e)
⟨*proof*⟩

definition *mono-edge-col* :: $('a \Rightarrow colour) \Rightarrow 'a\ hyp\text{-}edge \Rightarrow colour \Rightarrow bool$ **where**
mono-edge-col f e c $\equiv \forall\ v \in e.\ f\ v = c$

lemma *mono-edge-colI*: $(\bigwedge\ v.\ v \in e \Longrightarrow f\ v = c) \Longrightarrow$ *mono-edge-col f e c*
⟨*proof*⟩

lemma *mono-edge-colD*: *mono-edge-col f e c* $\Longrightarrow (\bigwedge\ v.\ v \in e \Longrightarrow f\ v = c)$
⟨*proof*⟩

lemma *mono-edge-alt-col*: *mono-edge f e* $\equiv \exists\ c.$ *mono-edge-col f e c*
⟨*proof*⟩

1.4 Proper colourings

A proper vertex colouring brings in the monochromatic edge decision. Note that this allows for a colouring of up to n colours, not precisely n colours

definition *is-proper-colouring* :: $('a \Rightarrow colour) \Rightarrow nat \Rightarrow bool$ **where**
is-proper-colouring f n $\equiv$ *vertex-colouring f n* $\wedge$ $(\forall\ e \in\#\ E.\ \forall\ c \in \{0..<n\}.\ f$ ‘ $e \neq \{c\})$

lemma *is-proper-colouring-alt*: *is-proper-colouring f n* $\longleftrightarrow$ *vertex-colouring f n* $\wedge(\forall\ e \in\#\ E.\ \neg$ *is-singleton* (f ‘ e))
⟨*proof*⟩

lemma *is-proper-colouring-alt2*: *is-proper-colouring f n* $\longleftrightarrow$ *vertex-colouring f n* $\wedge(\forall\ e \in\#\ E.\ \neg$ *mono-edge f e*)
⟨*proof*⟩

lemma *is-proper-colouringI*[*intro*]: *vertex-colouring f n* $\Longrightarrow (\bigwedge\ e\ .e \in\#\ E \Longrightarrow$
$\neg$ *is-singleton* (f ‘ e)) $\Longrightarrow$ *is-proper-colouring f n*
⟨*proof*⟩

lemma *is-proper-colouringI2*[*intro*]: *vertex-colouring f n* $\Longrightarrow (\bigwedge\ e\ .e \in\#\ E \Longrightarrow \neg$ *mono-edge f e*)
$\Longrightarrow$ *is-proper-colouring f n*

⟨*proof*⟩

lemma *is-proper-colouring-n0*: $\mathcal{V} \neq \{\} \Longrightarrow \neg$ *is-proper-colouring f 0*
⟨*proof*⟩

lemma *is-proper-colouring-empty*:
assumes $\mathcal{V} = \{\}$
shows *is-proper-colouring f n* $\longleftrightarrow f = (\lambda\ x\ .\ undefined)$
⟨*proof*⟩

lemma *is-proper-colouring-n1*:
assumes $\mathcal{V} \neq \{\}\ E \neq \{\#\}$
shows $\neg$ *is-proper-colouring f 1*
⟨*proof*⟩

lemma (**in** *fin-hypergraph*) *is-proper-colouring-image-card*:
assumes $\mathcal{V} \neq \{\}\ E \neq \{\#\}$
assumes $n > 1$
assumes *is-proper-colouring f n*
shows *card* $(f\ `\ \mathcal{V}) > 1$
⟨*proof*⟩

More monochromatic edges

lemma *no-monochomatic-is-colouring*:
assumes $\forall\ e \in\#\ E\ .\ \neg$ *mono-edge f e*
assumes *vertex-colouring f n*
shows *is-proper-colouring f n*
⟨*proof*⟩

lemma *ex-monochomatic-not-colouring*:
assumes $\exists\ e \in\#\ E\ .$ *mono-edge f e*
assumes *vertex-colouring f n*
shows $\neg$ *is-proper-colouring f n*
⟨*proof*⟩

lemma *mono-edge-colour-obtain*:
assumes *mono-edge f e*
assumes *vertex-colouring f n*
assumes $e \in\#\ E$
obtains *c* **where** $c \in \{0..<n\}$ **and** *mono-edge-col f e c*
⟨*proof*⟩

Complete proper colourings - i.e. when n colours are required

definition *is-complete-proper-colouring*:: $('a \Rightarrow colour) \Rightarrow nat \Rightarrow bool$ **where**
is-complete-proper-colouring f n $\equiv$ *is-proper-colouring f n* $\wedge f\ `\ \mathcal{V} = \{0..<n\}$

lemma *is-complete-proper-colouring-part*:
assumes *is-complete-proper-colouring f n*
shows *partition-on* $\mathcal{V}\ \{\ \{v \in \mathcal{V}\ .\ f\ v = c\}\ |\ c.\ c \in \{0..<n\}\}$

⟨*proof*⟩

lemma *is-complete-proper-colouring-n0*: $\mathcal{V} \neq \{\} \Longrightarrow \neg$ *is-complete-proper-colouring f 0*
⟨*proof*⟩

lemma *is-complete-proper-colouring-n1*:
assumes $\mathcal{V} \neq \{\}$ $E \neq \{\#\}$
shows $\neg$ *is-complete-proper-colouring f 1*
⟨*proof*⟩

lemma (**in** *fin-hypergraph*) *is-proper-colouring-reduce*:
assumes *is-proper-colouring f n*
obtains f' **where** *is-complete-proper-colouring* f' (*card* (f ‘ $\mathcal{V}$))
⟨*proof*⟩

lemma (**in** *fin-hypergraph*) *two-colouring-is-complete*:
assumes $\mathcal{V} \neq \{\}$
assumes $E \neq \{\#\}$
assumes *is-proper-colouring f 2*
shows *is-complete-proper-colouring f 2*
⟨*proof*⟩

1.5 n vertex colourings

definition *is-n-colourable* :: *nat* ⇒ *bool* **where**
is-n-colourable n ≡ ∃ *f* . *is-proper-colouring f n*

definition *is-n-edge-colourable* :: *nat* ⇒ *bool* **where**
is-n-edge-colourable n ≡ ∃ *f C* . *card C* = *n* ⟶ *proper-edge-colouring f C*

definition *all-n-vertex-colourings* :: *nat* ⇒ (′*a* ⇒ *colour*) *set* **where**
all-n-vertex-colourings n ≡ {*f* . *vertex-colouring f n*}

notation *all-n-vertex-colourings* (($\mathcal{C}$⁻) [*502*] *500*)

lemma *all-n-vertex-colourings-alt*: $\mathcal{C}^n = \mathcal{V} \rightarrow_E \{0..<n\}$
⟨*proof*⟩

lemma *vertex-colourings-empty*: $\mathcal{V} \neq \{\} \Longrightarrow$ *all-n-vertex-colourings 0* = {}
⟨*proof*⟩

lemma (**in** *fin-hypergraph*) *vertex-colourings-fin* : *finite* ($\mathcal{C}^n$)
⟨*proof*⟩

lemma (**in** *fin-hypergraph*) *count-vertex-colourings*: *card* ($\mathcal{C}^n$) = *n* ^ *horder*
⟨*proof*⟩

lemma *vertex-colourings-nempty*:

assumes *card* $\mathcal{V} \geq n$
assumes $n \neq 0$
shows $\mathcal{C}^n \neq \{\}$
⟨*proof*⟩

lemma *vertex-colourings-one*:
assumes $\mathcal{V} \neq \{\}$
shows $\mathcal{C}^1 = \{\lambda\ v \in \mathcal{V}\ .\ 0\}$
⟨*proof*⟩

lemma *mono-edge-set-union*:
assumes $e \in\#\ E$
shows $\{f \in \mathcal{C}^n\ .\ \textit{mono-edge}\ f\ e\} = (\bigcup c \in \{0..<n\}.\ \{f \in \mathcal{C}^n\ .\ \textit{mono-edge-col}\ f\ e\ c\})$
⟨*proof*⟩

end

Property B set up

abbreviation (**in** *hypergraph*) *has-property-B* :: *bool* **where**
has-property-B ≡ *is-n-colourable 2*

abbreviation *hyp-graph-order*:: *'a hyp-graph* ⇒ *nat* **where**
hyp-graph-order h ≡ *card* (*hyp-verts h*)

definition *not-col-n-uni-hyps*:: *nat* ⇒ *'a hyp-graph set*
where *not-col-n-uni-hyps n* ≡ { *h* . *fin-kuniform-hypergraph-nt* (*hyp-verts h*) (*hyp-edges h*) *n* ∧
¬ (*hypergraph.has-property-B* (*hyp-verts h*) (*hyp-edges h*)) }

definition *min-edges-colouring* :: *nat* ⇒ *'a itself* ⇒ *enat* **where**
min-edges-colouring n - ≡ *INF h* ∈ ((*not-col-n-uni-hyps n*) :: *'a hyp-graph set*) . *enat* (*size* (*hyp-edges h*))

lemma *obtains-min-edge-colouring*:
fixes *z* :: *'a itself*
assumes *min-edges-colouring n z* < *x*
obtains *h* :: *'a hyp-graph* **where** *h* ∈ *not-col-n-uni-hyps n* **and** *enat* (*size* (*hyp-edges h*)) < *x*
⟨*proof*⟩

1.6 Alternate Partition Definition.

Note that the indexed definition should be used most of the time instead

context *hypergraph*
begin

definition *is-proper-colouring-part* :: *'a set set* ⇒ *bool* **where**
is-proper-colouring-part C ≡ *partition-on* $\mathcal{V}$ *C* ∧ (∀ *c* ∈ *C*. ∀ *e* ∈# *E*. ¬ *e* ⊆ *c*)

definition *is-n-colourable-part* :: *nat* ⇒ *bool* **where**
is-n-colourable-part n ≡ ∃ *C* . *card C* = *n* ⟶ *is-proper-colouring-part C*

abbreviation *has-property-B-part* :: *bool* **where**
has-property-B-part ≡ *is-n-colourable-part 2*

definition *mono-edge-ss* :: *'a set set* ⇒ *'a hyp-edge* ⇒ *bool* **where**
mono-edge-ss C e ≡ ∃ *c* ∈ *C*. *e* ⊆ *c*

lemma *is-proper-colouring-partI*: *partition-on* 𝒱 *C* ⟹ (∀ *c* ∈ *C*. ∀ *e* ∈# *E*. ¬ *e* ⊆ *c*) ⟹
is-proper-colouring-part C
⟨*proof*⟩

lemma *no-monochomatic-is-colouring-part*:
assumes ∀ *e* ∈# *E* . ¬ *mono-edge-ss C e*
assumes *partition-on* 𝒱 *C*
shows *is-proper-colouring-part C*
⟨*proof*⟩

lemma *ex-monochomatic-not-colouring-part*:
assumes ∃ *e* ∈# *E* . *mono-edge-ss C e*
assumes *partition-on* 𝒱 *C*
shows ¬ *is-proper-colouring-part C*
⟨*proof*⟩

definition *all-n-vertex-colourings-part* :: *nat* ⇒ *'a set set set* **where**
all-n-vertex-colourings-part n ≡ {*C* . *partition-on* 𝒱 *C* ∧ *card C* = *n*}

lemma (**in** *fin-hypergraph*) *all-vertex-colourings-part-fin*: *finite* (*all-n-vertex-colourings-part n*)
⟨*proof*⟩

lemma *all-vertex-colourings-part-nempty*: *card* 𝒱 ≥ *n* ⟹ *n* ≠ *0* ⟹ *all-n-vertex-colourings-part n* ≠ {}
⟨*proof*⟩

lemma *disjoint-family-on-colourings*:
assumes *e* ∈# *E*
shows *disjoint-family-on* (λ *c*. {*f* ∈ $\mathcal{C}^n$. *mono-edge-col f e c*}) {*0..<n*}
⟨*proof*⟩

end

end

2 Basic Probabilistic Method Application

This section establishes step (1) of the basic framework for incidence set systems, as well as some basic bounds on hypergraph colourings

theory *Basic-Bounds-Application* **imports** *Lovasz-Local.Basic-Method Hypergraph-Colourings*
begin

2.1 Probability Spaces for Incidence Set Systems

This is effectively step (1) of the formal framework for probabilistic method. Unlike stages (3) and (4), which were formalised in the Lovasz_Local_Lemma AFP entry, this stage required a formalisation of incidence set systems as well as the background probability space locales

A basic probability space for a point measure on a non-trivial structure

locale *vertex-fn-space* = *fin-hypersystem-vne* +
  fixes F :: *'a set* ⇒ *'b set*
  fixes p :: *'b* ⇒ *real*
  assumes *ne*: $F\ \mathcal{V} \neq \{\}$
  assumes *fin*: *finite* $(F\ \mathcal{V})$
  assumes *pgte0*: $\bigwedge fv\ .\ fv \in F\ \mathcal{V} \Longrightarrow p\ fv \geq 0$
  assumes *sump*: $(\sum x \in (F\ \mathcal{V})\ .\ p\ x) = 1$
begin

definition $\Omega \equiv F\ \mathcal{V}$

lemma *fin*-Ω: *finite* Ω
  ⟨*proof*⟩

lemma *ne*-Ω: $\Omega \neq \{\}$
  ⟨*proof*⟩

definition M = *point-measure* Ω p

lemma *space-eq*: *space* $M = \Omega$
  ⟨*proof*⟩

lemma *sets-eq*: *sets* M = *Pow* (Ω)
  ⟨*proof*⟩

lemma *finite-event*: $A \subseteq \Omega \Longrightarrow$ *finite* A
  ⟨*proof*⟩

lemma *emeasure-eq*: *emeasure* $M\ A$ = (*if* $(A \subseteq \Omega)$ *then* $(\sum a{\in}A.\ p\ a)$ *else 0*)
⟨*proof*⟩

lemma *integrable-M*[*intro*, *simp*]: *integrable* M (f::- ⇒ *real*)
  ⟨*proof*⟩

lemma *borel-measurable-M*[*measurable*]: $f \in$ *borel-measurable M*
⟨*proof*⟩

lemma *prob-space-M*: *prob-space M*
⟨*proof*⟩

end

sublocale *vertex-fn-space* $\subseteq$ *prob-space M*
⟨*proof*⟩

A uniform variation of the space

locale *vertex-fn-space-uniform* = *fin-hypersystem-vne* +
fixes F :: *'a set* $\Rightarrow$ *'b set*
assumes *ne*: $F\ \mathcal{V} \neq \{\}$
assumes *fin*: *finite* $(F\ \mathcal{V})$
begin

definition $\Omega U \equiv F\ \mathcal{V}$

definition $MU \equiv$ *uniform-count-measure* ΩU

end

sublocale *vertex-fn-space-uniform* $\subseteq$ *vertex-fn-space* $\mathcal{V}$ E F ($\lambda x.\ 1$ / *card* ΩU)
rewrites $\Omega = \Omega U$ **and** $M = MU$
⟨*proof*⟩

context *vertex-fn-space-uniform*
begin

lemma *emeasure-eq*: *emeasure MU A* = (*if* ($A \subseteq \Omega U$) *then* ((*card A*)/*card* (ΩU)) *else 0*)
⟨*proof*⟩

lemma *measure-eq-valid*: $A \in$ *events* $\Longrightarrow$ *measure MU A* = (*card A*)/*card* (ΩU)
⟨*proof*⟩

lemma *expectation-eq*:
shows *expectation f* = ($\sum$ $x \in \Omega U$. *f x*) / *card* ΩU
⟨*proof*⟩

end

A probability space over the full vertex set

locale *vertex-space* = *fin-hypersystem-vne* +
fixes p :: *'a* $\Rightarrow$ *real*
assumes *pgte0*: $\bigwedge$ *fv* . *fv* $\in \mathcal{V} \Longrightarrow$ *p fv* $\geq$ *0*

assumes *sump*: $(\sum x \in (\mathcal{V}) \ . \ p \ x) = 1$

sublocale *vertex-space* ⊆ *vertex-fn-space* $\mathcal{V}$ *E* λ *i* . *i p*
rewrites $\Omega = \mathcal{V}$
⟨*proof*⟩

A uniform variation of the probability space over the vertex set

locale *vertex-space-uniform* = *fin-hypersystem-vne*

sublocale *vertex-space-uniform* ⊆ *vertex-fn-space-uniform* $\mathcal{V}$ *E* λ *i* . *i*
rewrites $\Omega U = \mathcal{V}$
⟨*proof*⟩

A uniform probability space over a vertex subset

locale *vertex-ss-space-uniform* = *fin-hypersystem-vne* +
fixes *VS*
assumes *vs-ss*: $VS \subseteq \mathcal{V}$
assumes *ne-vs*: $VS \neq \{\}$
begin

lemma *finite-vs*: *finite VS*
⟨*proof*⟩

end

sublocale *vertex-ss-space-uniform* ⊆ *vertex-fn-space-uniform* $\mathcal{V}$ *E* λ *i*. *VS*
rewrites $\Omega = VS$
⟨*proof*⟩

A non-uniform prob space over a vertex subset

locale *vertex-ss-space* = *fin-hypersystem-vne* +
fixes *VS*
assumes *vs-ss*: $VS \subseteq \mathcal{V}$
assumes *ne-vs*: $VS \neq \{\}$
fixes $p :: 'a \Rightarrow real$
assumes *pgte0*: $\bigwedge fv \ . \ fv \in VS \Longrightarrow p \ fv \geq 0$
assumes *sump*: $(\sum x \in (VS) \ . \ p \ x) = 1$
begin

lemma *finite-vs*: *finite VS*
⟨*proof*⟩

end

sublocale *vertex-ss-space* ⊆ *vertex-fn-space* $\mathcal{V}$ *E* λ *i* . *VS p*
rewrites $\Omega = VS$
⟨*proof*⟩

A uniform probability space over a property on the vertex set

locale *vertex-prop-space* = *fin-hypersystem-vne* +
  fixes P :: *'b set*
  assumes *finP*: *finite P*
  assumes *nempty-P*: $P \neq \{\}$

sublocale *vertex-prop-space* ⊆ *vertex-fn-space-uniform* $\mathcal{V}$ E λ V. $V \rightarrow_E P$
rewrites $\Omega U = \mathcal{V} \rightarrow_E P$
⟨*proof*⟩

context *vertex-prop-space*
begin

lemma *prob-uniform-vertex-subset*:
  assumes $b \in P$
  assumes $d \subseteq \mathcal{V}$
  shows *prob* $\{f \in \Omega\ .\ (\forall\ v \in d\ .f\ v = b)\} = 1/((card\ P)\ powi\ (card\ d))$
  ⟨*proof*⟩

lemma *prob-uniform-vertex*:
  assumes $b \in P$
  assumes $v \in \mathcal{V}$
  shows *prob* $\{f \in \Omega U\ .\ f\ v = b\} = 1/(card\ P)$
⟨*proof*⟩

end

A uniform vertex colouring space

locale *vertex-colour-space* = *fin-hypergraph-nt* +
  fixes n :: *nat*
  assumes *n-lt-order*: $n \leq horder$
  assumes *n-not-zero*: $n \neq 0$

sublocale *vertex-colour-space* ⊆ *vertex-prop-space* $\mathcal{V}$ E $\{0..<n\}$
  rewrites $\Omega U = \mathcal{C}^n$
⟨*proof*⟩

This probability space contains several useful lemmas on basic vertex colouring probabilities (and monochromatic edges), which are facts that are typically either not proven, or have very short proofs on paper

context *vertex-colour-space*
begin

lemma *colour-set-event*: $\{f \in \mathcal{C}^n.\ mono\text{-}edge\text{-}col\ f\ e\ c\} \in events$
  ⟨*proof*⟩

lemma *colour-functions-event*: $(\lambda\ c.\ \{f \in \mathcal{C}^n\ .\ mono\text{-}edge\text{-}col\ f\ e\ c\})$ ‘ $\{0..<n\} \subseteq$ *events*
  ⟨*proof*⟩

lemma *prob-vertex-colour*: $v \in \mathcal{V} \Longrightarrow c \in \{0..<n\} \Longrightarrow prob\ \{f \in \mathcal{C}^n\ .\ f\ v = c\} = 1/n$
⟨*proof*⟩

lemma *prob-edge-colour*:
assumes $e \in\#\ E\ c \in \{0..<n\}$
shows $prob\ \{f \in \mathcal{C}^n\ .\ mono\text{-}edge\text{-}col\ f\ e\ c\} = 1/(n\ powi\ (card\ e))$
⟨*proof*⟩

lemma *prob-monochromatic-edge-inv*:
assumes $e \in\#\ E$
shows $prob\{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\} = 1/(n\ powi\ (int\ (card\ e) - 1))$
⟨*proof*⟩

lemma *prob-monochromatic-edge*:
assumes $e \in\#\ E$
shows $prob\{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\} = n\ powi\ (1 - int\ (card\ e))$
⟨*proof*⟩

lemma *prob-monochromatic-edge-bound*:
assumes $e \in\#\ E$
assumes $\bigwedge e.\ e \in\#E \Longrightarrow card\ e \geq k$
assumes $k > 0$
shows $prob\{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\} \leq 1/((real\ n)\ powi\ (k-1))$
⟨*proof*⟩

end

2.2 More Hypergraph Colouring Results

context *fin-hypergraph-nt*
begin

lemma *not-proper-colouring-edge-mono*: $\{f \in \mathcal{C}^n\ .\ \neg\ is\text{-}proper\text{-}colouring\ f\ n\} = (\bigcup e \in (set\text{-}mset\ E).\ \{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\})$
⟨*proof*⟩

lemma *proper-colouring-edge-mono*: $\{f \in \mathcal{C}^n\ .\ is\text{-}proper\text{-}colouring\ f\ n\} = (\bigcap e \in (set\text{-}mset\ E).\ \{f \in \mathcal{C}^n\ .\ \neg\ mono\text{-}edge\ f\ e\})$
⟨*proof*⟩

lemma *proper-colouring-edge-mono-compl*: $\{f \in \mathcal{C}^n\ .\ is\text{-}proper\text{-}colouring\ f\ n\} = (\bigcap e \in (set\text{-}mset\ E).\ \mathcal{C}^n - \{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\})$
⟨*proof*⟩

lemma *event-is-proper-colouring*:
assumes $g \in \mathcal{C}^n$
assumes $g \notin (\bigcup e \in (set\text{-}mset\ E).\ \{f \in \mathcal{C}^n\ .\ mono\text{-}edge\ f\ e\})$

shows *is-proper-colouring g n*
⟨*proof*⟩

end

2.3 The Basic Application

The comments below show the basic framework steps

context *fin-kuniform-hypergraph-nt*
begin
proposition *erdos-propertyB*:
assumes *size E* < (*2*^(*k* − *1*))
assumes *k* > *0*
shows *has-property-B*
⟨*proof*⟩

end

corollary *erdos-propertyB-min*:
fixes *z* :: *'a itself*
assumes *n* > *0*
shows (*min-edges-colouring n z*) ≥ *2*^(*n* − *1*)
⟨*proof*⟩

end

3 Lovasz Local Framework Application

theory *LLL-Applications* **imports** *Lovasz-Local.Lovasz-Local-Lemma*
Lovasz-Local.Indep-Events Twelvefold-Way.Twelvefold-Way-Core
Design-Theory.Multisets-Extras Basic-Bounds-Application
begin

3.1 More set extras

lemma *multiset-remove1-filter*: *a* ∈# *A* ⟹ *P a* ⟹
{#*b* ∈# *A* . *P b*#} = {#*b* ∈# *remove1-mset a A* . *P b*#} + {#*a*#}
⟨*proof*⟩

lemma *card-partition-image*:
assumes *finite C*
assumes *finite* (⋃*c* ∈ *C* . *f c*)
assumes (⋀ *c*. *c*∈*C* ⟹ *card* (*f c*) = *k*)
assumes (⋀ *c1 c2*. *c1* ∈ *C* ⟹ *c2* ∈ *C* ⟹ *c1* ≠ *c2* ⟹ *f c1* ∩ *f c2* = {})
shows *k* ∗ *card* (*f* ‘ *C*) = *card* (⋃*c* ∈ *C* . *f c*)
⟨*proof*⟩

lemma *mset-set-implies*:
  assumes *image-mset f* (*mset-set A*) = *B*
  assumes $\bigwedge$ *a* . *a* ∈ *A* ⟹ *P* (*f a*)
  shows $\bigwedge$ *b*. *b* ∈# *B* ⟹ *P b*
⟨*proof*⟩

lemma *card-partition-image-inj*:
  assumes *finite C*
  assumes *inj-on f C*
  assumes *finite* ($\bigcup$ *c* ∈ *C* . *f c*)
  assumes ($\bigwedge$ *c*. *c*∈*C* ⟹ *card* (*f c*) = *k*)
  assumes ($\bigwedge$ *c1 c2*. *c1* ∈ *C* ⟹ *c2* ∈ *C* ⟹ *c1* ≠ *c2* ⟹ *f c1* ∩ *f c2* = {})
  shows *k* ∗ *card* (*C*) = *card* ($\bigcup$ *c* ∈ *C* . *f c*)
⟨*proof*⟩

lemma *size-big-union-sum2*:
  fixes *M* :: *'a* ⇒ *'b multiset*
  shows *size* ($\sum$ *x* ∈# *X* . *M x*) = ($\sum$ *x* ∈#*X* . *size* (*M x*))
  ⟨*proof*⟩

lemma *size-big-union-sum2-const*:
  fixes *M* :: *'a* ⇒ *'b multiset*
  assumes $\bigwedge$ *x* . *x* ∈# *X* ⟹ *size* (*M x*) = *k*
  shows *size* ($\sum$ *x* ∈# *X* . *M x*) = *size X* ∗ *k*
⟨*proof*⟩

lemma *count-sum-mset2*: *count* ($\sum$ *x* ∈# *X* . *M x*) *a* = ($\sum$ *x* ∈# *X* . *count* (*M* *x*) *a*)
  ⟨*proof*⟩

lemma *mset-subset-eq-elemI*:
  ($\bigwedge$*a* . *a* ∈# *A* ⟹ *count A a* ≤ *count B a*) ⟹ *A* ⊆# *B*
  ⟨*proof*⟩

lemma *mset-obtain-from-filter*:
  assumes *a* ∈# {# *b* ∈# *B* . *P b* #}
  shows *a* ∈# *B* **and** *P a*
  ⟨*proof*⟩

3.2 Mutual Independence Principle for Hypergraphs

context *fin-hypergraph-nt*
begin

definition (**in** *incidence-system*) *block-intersect-count* :: *'a set* ⇒ *nat* **where**
block-intersect-count b ≡ *size* {# *b2* ∈# ($\mathcal{B}$ − {#*b*#}) . *b2* ∩ *b* ≠ {} #}

lemma (**in** *hypergraph*) *edge-intersect-count-inc*:
  assumes *e* ∈# *E*

shows *size* {# *f* ∈# *E* . *f* ∩ *e* ≠ {}#} = *block-intersect-count e* + *1*
⟨*proof*⟩

lemma *disjoint-set-is-mutually-independent*:
assumes *iin*: *i* ∈ {*0*..<(*size E*)}
assumes *idffn*: *idf* ∈ {*0*..<*size E*} $\rightarrow_E$ *set-mset E*
assumes *Aefn*: ⋀ *i*. *i* ∈ {*0*..<*size E*} ⟹ *Ae i* = {*f* ∈ $\mathcal{C}^2$. *mono-edge f* (*idf i*)}
shows *prob-space.mutual-indep-events* (*uniform-count-measure* ($\mathcal{C}^2$)) (*Ae i*) *Ae* ({*j* ∈{*0*..<(*size E*)} . (*idf j* ∩ *idf i*) = {}})
⟨*proof*⟩

lemma *intersect-empty-set-size*:
assumes ⋀ *e* . *e* ∈#*E* ⟹ *size* {# *f* ∈# (*E* − {#*e*#}) . *f* ∩ *e* ≠ {}#} ≤ *d*
assumes *e2* ∈# *E*
shows *size* {#*e* ∈# *E* . *e* ∩ *e2* = {} #} ≥ *size E* − *d* − *1* (**is** *size ?S′* ≥ *size E* − *d* − *1*)
⟨*proof*⟩

3.3 Application Property B

Probabilistic framework clearly notated

proposition *erdos-propertyB-LLL*:
assumes ⋀ *e*. *e* ∈#*E* ⟹ *card e* ≥ *k*
assumes ⋀ *e* . *e* ∈#*E* ⟹ *size* {# *f* ∈# (*E* − {#*e*#}) . *f* ∩ *e* ≠ {}#} ≤ *d*
assumes *exp*(*1*)∗(*d*+*1*) ≤ (*2 powi* (*k* − *1*))
assumes *k* > *0*
shows *has-property-B*
⟨*proof*⟩

end

3.4 Application Corollary

A corollary on hypergraphs where k ≥9

lemma *exp-ineq-k9*:
fixes *k*:: *nat*
assumes *k* ≥ *9*
shows *exp*(*1*::*real*) ∗ (*k* ∗(*k* −*1*) + *1*) < *2*^(*k*−*1*)
⟨*proof*⟩

context *fin-kuniform-regular-hypgraph-nt*
begin

Good example of a combinatorial counting proof in a formal environment

lemma (**in** *fin-dregular-hypergraph*) *hdeg-remove-one*:
assumes *e* ∈# *E*
assumes *v* ∈# *mset-set e*
shows *size* {# *f* ∈# (*E* − {#*e*#}) . *v* ∈ *f*#} = *d* − *1*

⟨*proof*⟩

lemma *max-intersecting-edges*:
assumes $e \in\# E$
shows $size \{\# f \in\# (E - \{\#e\#\}) . f \cap e \neq \{\}\#\} \leq k * (k - 1)$
⟨*proof*⟩

corollary *erdos-propertyB-LLL9*:
assumes $k \geq 9$
shows *has-property-B*
⟨*proof*⟩

end

end
theory *Hypergraph-Colourings-Root*
imports
Hypergraph-Colourings
Basic-Bounds-Application
LLL-Applications
begin
end

References

[1] N. Alon and J. H. Spencer. *The Probabilistic Method*. Wiley-Interscience Series in Discrete Mathematics and Optimization. Wiley, Hoboken, N.J, 4th edition, 2016.

[2] M. Molloy and B. Reed. *Graph Colouring and the Probabilistic Method*. Algorithms and Combinatorics. Springer, 2002.

[3] Y. Zhao. Probabilistic methods in combinatorics, 2020. Lecture notes MIT 18.226, Fall 2020, https://ocw.mit.edu/courses/18-226-probabilistic-method-in-combinatorics-fall-2020/resources/mit18_226f20_full_notes/.