Probabilistic Methods for Combinatorial Structures in Isabelle/HOL

Chelsea Edmonds[1] | c.l.edmonds@sheffield.ac.uk

Based on PhD work supervised by Lawrence C. Paulson[2] [Published at CPP2024]

University of Sheffield[1] | University of Cambridge[1,2]

Proof Systems for Mathematics and Verification | EPFL June 2024

Overview

- Motivation: formalising maths, techniques vs theorems, the probabilistic method
- **What we did:**
  i. Combinatorial structure extensions
  ii. **A general framework**: probabilistic spaces for combinatorial structures
  iii. Probability library extensions, including the Lovász local lemma
  iv. A sample application: Hypergraph Colourings
- **What we learnt:**
  - Formalisation insights: modularity, locale techniques etc
  - Mathematical insights: circular reasoning in human intuition

1. The Motivating Problem

The Probabilistic Method - Why Formalise?

The Probabilistic Method is one of the most powerful and widely used tools applied in combinatorics (Alon & Spencer, 2015).

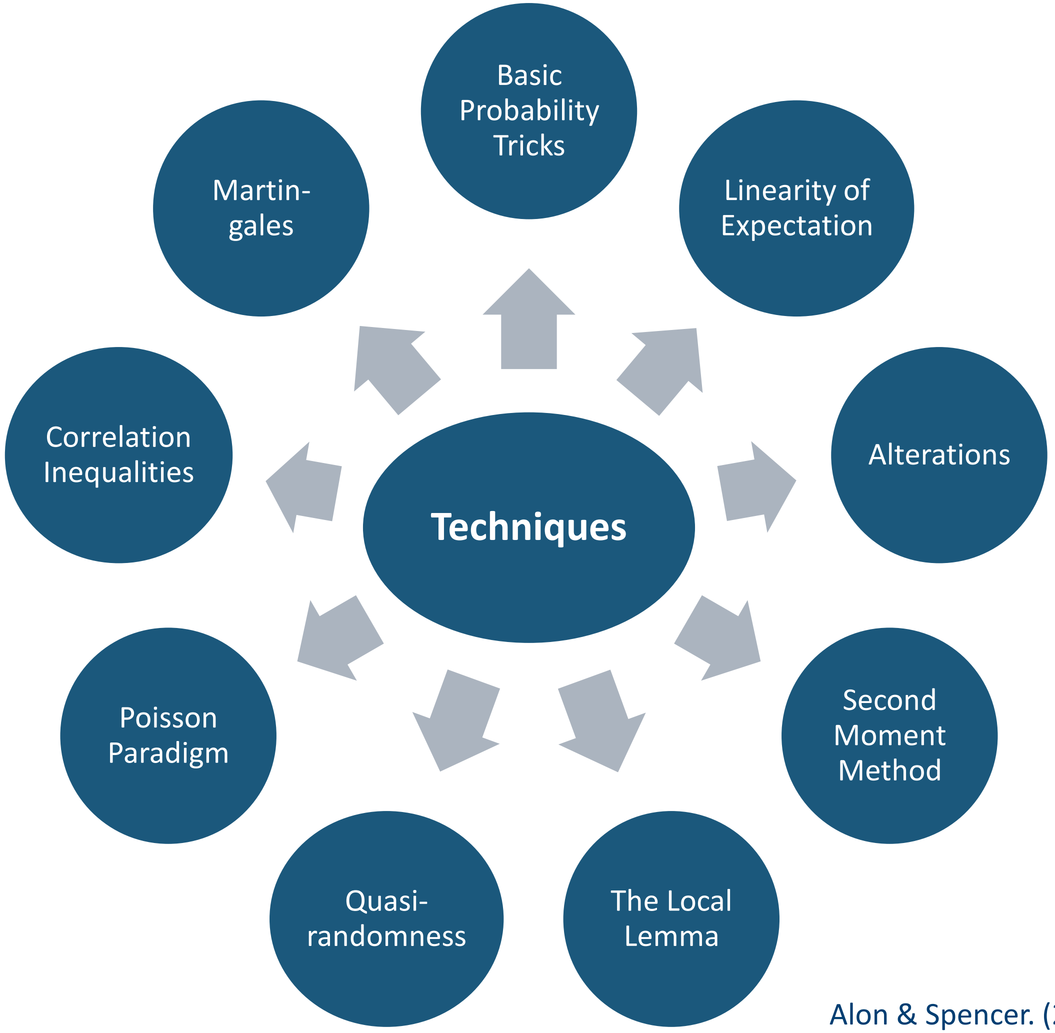
- Interest in formalised maths has grown significantly
- Only three pre-existing formalisations which use the probabilistic method in combinatorics -> *focused on theorems not general techniques.*
- Predominance of method in **modern** combinatorics research motivated by **applications** -> *how can we make it easier to formalise future work?*

What is the Probabilistic Method?

Key Idea

Given a probability space over some combinatorial structure…

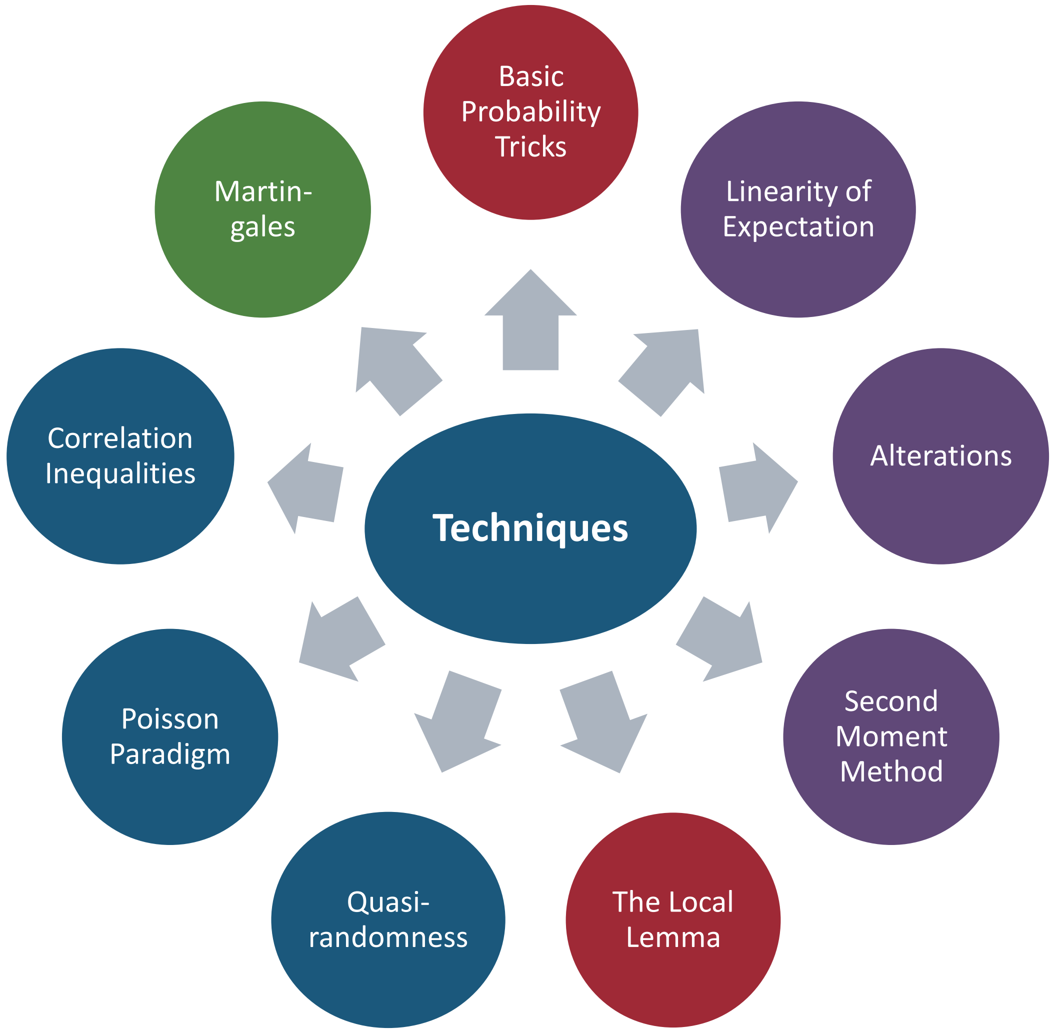
Show the structure has the desired properties with positive probability.
(May be via complement)



Alon & Spencer. (2004).
The Probabilistic Method.

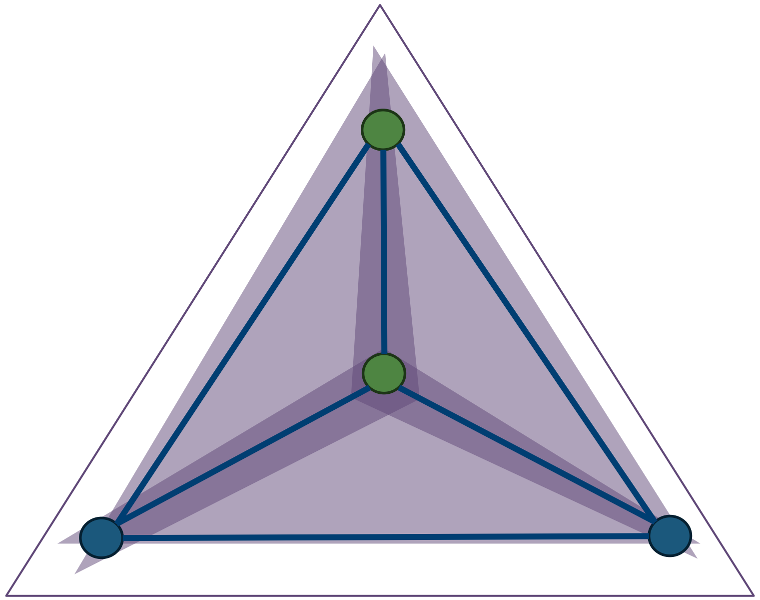
What is the Probabilistic Method?

The Probabilistic Method is one of the most powerful and widely used tools applied in combinatorics (Alon & Spencer, 2015).

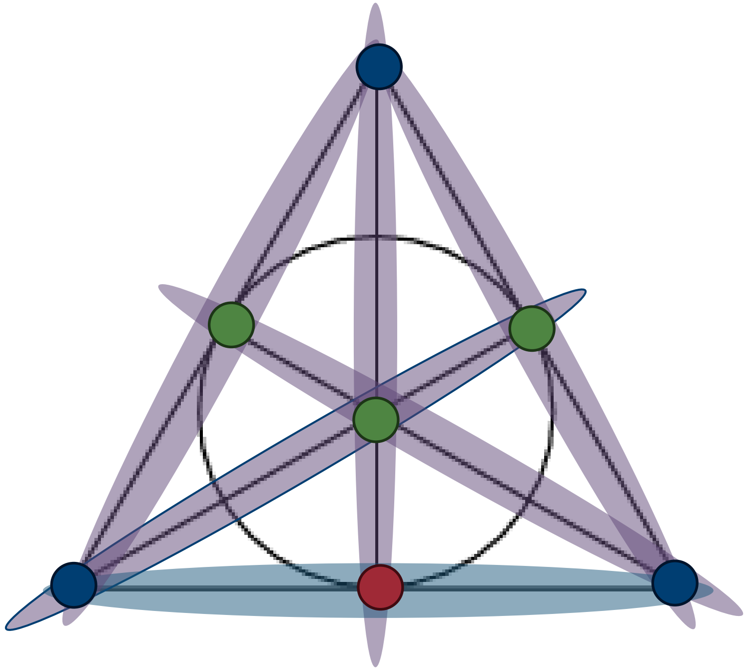


Hypergraph Colourings.

- A *hypergraph* (V, E), where E is a collection of subsets of V of any size, is “colourable” if there is a vertex colouring such that no edge is monochromatic.



2- colourable 3-uniform w/ 4 edges



Not 2- colourable 3-uniform w/ 7 edges

A Basic Proof

The Probabilistic Method:

Prove existence by showing a structure has a desired property with probability > 0

(or avoids bad properties with probability < 1)

Proposition 1.3.1 [Erdős (1963a)] *Every n-uniform hypergraph with less than 2^{n-1} edges has property B. Therefore $m(n) \geq 2^{n-1}$.*

Proof. Let $H = (V, E)$ be an n-uniform hypergraph with less than 2^{n-1} edges. Color V randomly by two colors. For each edge $e \in E$, let A_e be the event that e is monochromatic. Clearly $\Pr[A_e] = 2^{1-n}$. Therefore

$$\Pr\left[\bigvee_{e \in E} A_e\right] \leq \sum_{e \in E} \Pr[A_e] < 1$$

and there is a two-coloring without monochromatic edges. ■

Isabelle/HOL

- Simple type theory
- **Automation:** Sledgehammer
- **Search tools:** Query Search, Find Facts, SErAPIS
- The Isar **structured proof language**
- **IDE:** JEdit & VSCodium
- **Libraries:** Distribution & Archive of Formal Proofs (AFP)
- **Additional features:** Code generation, modularity, polymorphism, documentation generation …

```
theorem  assumes "prime p"  shows "sqrt p ∉ ℚ"
proof
  from ‹prime p› have p: "1 < p" by (simp add: prime_def)
  assume "sqrt p ∈ ℚ"
  then obtain m n :: nat where
      n: "n ≠ 0" and sqrt_rat: "¦sqrt p¦ = m / n"
    and "coprime m n" by (rule Rats_abs_nat_div_natE)
  have eq: "m² = p * n²"
  proof -
    from n and sqrt_rat have "m = ¦sqrt p¦ * n" by simp
    then show "m² = p * n²"
      by (metis abs_of_nat of_nat_eq_iff of_nat_mult power2_eq_square real_sqrt_abs2 rea
  qed
  have "p dvd m ∧ p dvd n"
  proof
    from eq have "p dvd m²" ..
    with ‹prime p› show "p dvd m" by (rule prime_dvd_power_nat)
    then obtain k where "m = p * k" ..
    with eq have "p * n² = p² * k²" by (auto simp add: power2_eq_square ac_simps)
    with ‹prime p› show "p dvd n"
      by (metis dvd_triv_left nat_mult_dvd_cancel1 power2_eq_square prime_dvd_power_nat
  qed
  then have "p dvd gcd m n" by simp
  with ‹coprime m n› have "p = 1" by simp
  with p show False by simp
qed
```

sledgehammer proofs

2. Formally talking about Hypergraphs…

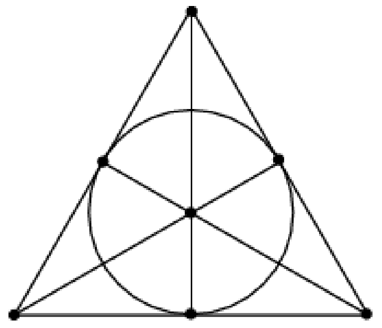
Hypergraph Structures

CHALLENGE 1

- Many combinatorial structures have the same underlying definition.

BUT

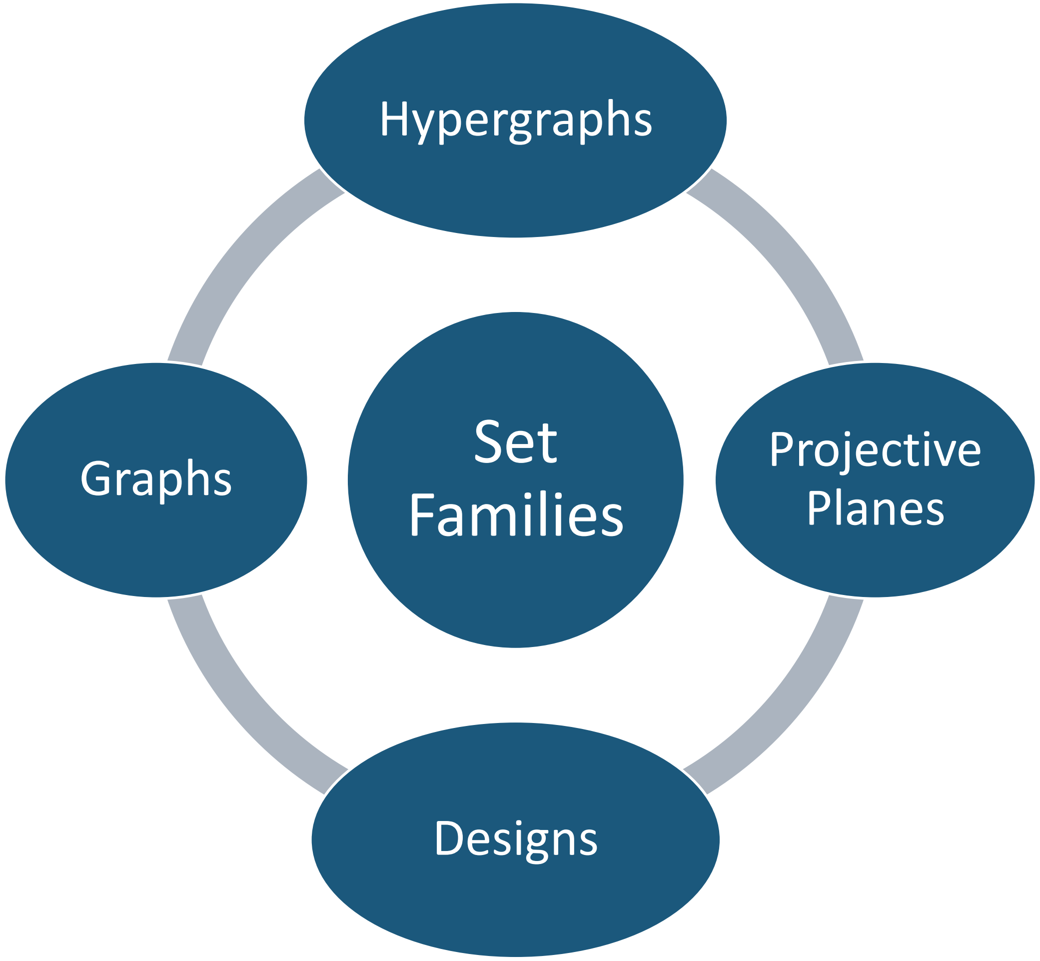
- Different languages/concepts/intuition
- Complex inheritance patterns
- Inconsistencies in definitions



The Fano Plane

{0, 1, 2}, {0, 3, 4}, {0, 5, 6}, {1, 3, 5}, {1, 4, 6}, {2, 3, 6}, {2, 4, 5}

Design Rep



Locales Basics

- Locales are Isabelle's module system. From a logical perspective, they are simply persistent contexts.

$$\bigwedge x_1 \dots x_n. \llbracket A_1; \dots; A_m \rrbracket \Rightarrow C.$$

- A simple example for combinatorics:

Notation

Parameters

```
locale incidence_system =
  fixes point_set :: "'a set" ("𝒱")
  fixes block_collection :: "'a set multiset" ("ℬ")
  assumes wellformed: "b ∈# ℬ ⟹ b ⊆ 𝒱"
```

Assumptions

Locales Basics – Inheritance and Interpretations

- We have direct inheritance

```
locale hypersystem = incidence_system "vertices :: 'a set" "edges :: 'a hyp_edge multiset"
  for "vertices" ("𝒱") and "edges" ("E")
```

- And indirect inheritance (rewriting optional)

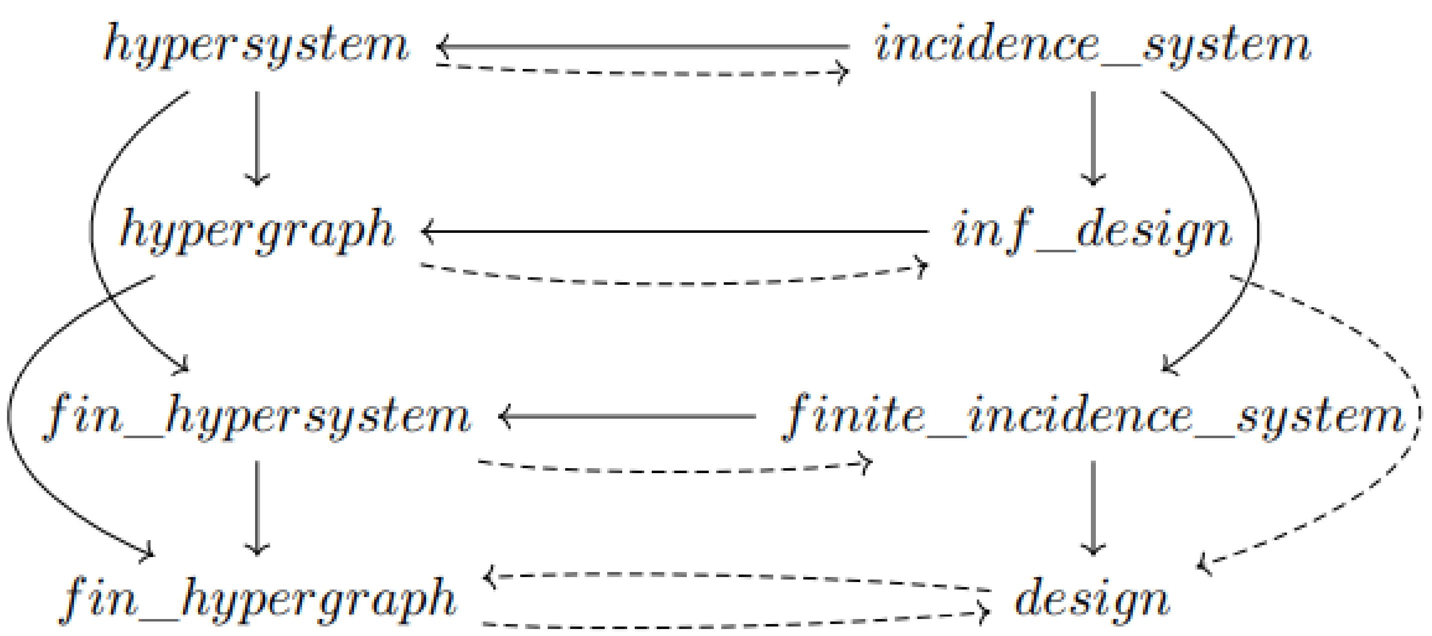
```
sublocale incidence_system ⊆ hypersystem 𝒱 ℬ
  rewrites "hdegree v = point_replication_number ℬ v" and "hdegree_set vs = points_index ℬ vs"
```

- Interpretations (global & local)

```
interpret h: hypergraph "hyp_verts H" "hyp_edges H"
  using assms(3) by simp
```

Designs to Hypergraphs

Locales let us reuse lemmas, theorems, and definitions from prior combinatorial structure formalisations



2. A General Probabilistic Framework

A Basic Proof

The Probabilistic Method:

Prove existence by showing a structure has a desired property with probability > 0

(or avoids bad properties with probability < 1)

Proposition 1.3.1 [Erdős (1963a)] *Every n-uniform hypergraph with less than 2^{n-1} edges has property B. Therefore $m(n) \geq 2^{n-1}$.*

Proof. Let $H = (V, E)$ be an n-uniform hypergraph with less than 2^{n-1} edges. Color V randomly by two colors. For each edge $e \in E$, let A_e be the event that e is monochromatic. Clearly $\Pr[A_e] = 2^{1-n}$. Therefore

$$\Pr\left[\bigvee_{e \in E} A_e\right] \leq \sum_{e \in E} \Pr[A_e] < 1$$

and there is a two-coloring without monochromatic edges. ■

Identified Formalisation Challenges

- Reliance on human intuition
- Complex calculations
- Set up involved
- Definitions and notation

A first attempt at formalising a proof written in 1 line on paper!

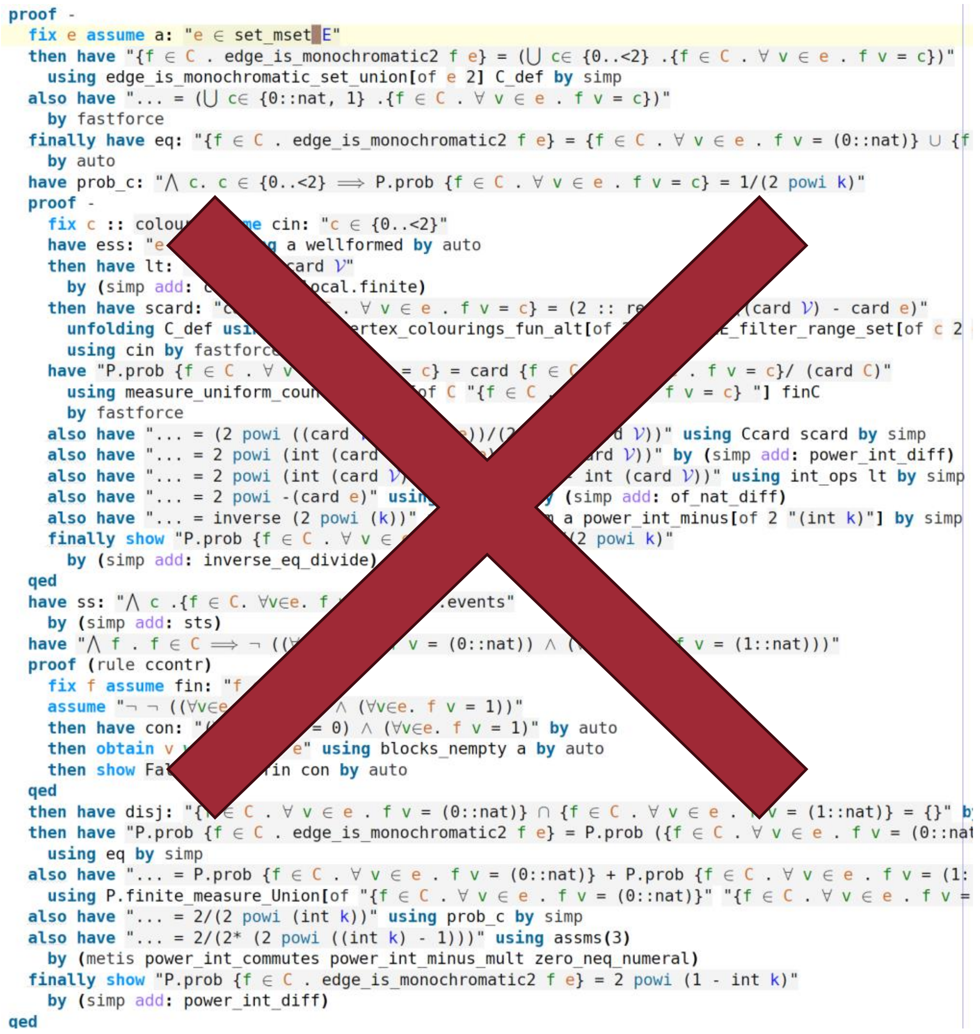
```
proof -
  fix e assume a: "e ∈ set_mset E"
  then have "{f ∈ C . edge_is_monochromatic2 f e} = (⋃ c∈ {0..<2} .{f ∈ C . ∀ v ∈ e . f v = c})"
    using edge_is_monochromatic_set_union[of e 2] C_def by simp
  also have "... = (⋃ c∈ {0::nat, 1} .{f ∈ C . ∀ v ∈ e . f v = c})"
    by fastforce
  finally have eq: "{f ∈ C . edge_is_monochromatic2 f e} = {f ∈ C . ∀ v ∈ e . f v = (0::nat)} ∪ {f
    by auto
  have prob_c: "⋀ c. c ∈ {0..<2} ⟹ P.prob {f ∈ C . ∀ v ∈ e . f v = c} = 1/(2 powi k)"
  proof -
    fix c :: colour assume cin: "c ∈ {0..<2}"
    have ess: "e ⊆ 𝒱" using a wellformed by auto
    then have lt: "card e ≤ card 𝒱"
      by (simp add: card_mono local.finite)
    then have scard: "card {f ∈ C . ∀ v ∈ e . f v = c} = (2 :: real) powi ((card 𝒱) - card e)"
      unfolding C_def using all_n_vertex_colourings_fun_alt[of 2] card_PiE_filter_range_set[of c 2
      using cin by fastforce
    have "P.prob {f ∈ C . ∀ v ∈ e . f v = c} = card {f ∈ C . ∀ v ∈ e . f v = c}/ (card C)"
      using measure_uniform_count_measure[of C "{f ∈ C . ∀ v ∈ e . f v = c} "] finC
      by fastforce
    also have "... = (2 powi ((card 𝒱) - card e))/(2 powi (card 𝒱))" using Ccard scard by simp
    also have "... = 2 powi (int (card 𝒱 - card e) - int (card 𝒱))" by (simp add: power_int_diff)
    also have "... = 2 powi (int (card 𝒱) - int (card e) - int (card 𝒱))" using int_ops lt by simp
    also have "... = 2 powi -(card e)" using assms(1) by (simp add: of_nat_diff)
    also have "... = inverse (2 powi (k))" using uniform a power_int_minus[of 2 "(int k)"] by simp
    finally show "P.prob {f ∈ C . ∀ v ∈ e . f v = c} = 1/(2 powi k)"
      by (simp add: inverse_eq_divide)
  qed
  have ss: "⋀ c .{f ∈ C. ∀v∈e. f v = c} ∈ P.events"
    by (simp add: sts)
  have "⋀ f . f ∈ C ⟹ ¬ ((∀ v ∈ e . f v = (0::nat)) ∧ (∀ v ∈ e . f v = (1::nat)))"
  proof (rule ccontr)
    fix f assume fin: "f ∈ C"
    assume "¬ ¬ ((∀v∈e. f v = 0) ∧ (∀v∈e. f v = 1))"
    then have con: "(∀v∈e. f v = 0) ∧ (∀v∈e. f v = 1)" by auto
    then obtain v where "v ∈ e" using blocks_nempty a by auto
    then show False using fin con by auto
  qed
  then have disj: "{f ∈ C . ∀ v ∈ e . f v = (0::nat)} ∩ {f ∈ C . ∀ v ∈ e . f v = (1::nat)} = {}" b
  then have "P.prob {f ∈ C . edge_is_monochromatic2 f e} = P.prob ({f ∈ C . ∀ v ∈ e . f v = (0::nat
    using eq by simp
  also have "... = P.prob {f ∈ C . ∀ v ∈ e . f v = (0::nat)} + P.prob {f ∈ C . ∀ v ∈ e . f v = (1:
    using P.finite_measure_Union[of "{f ∈ C . ∀ v ∈ e . f v = (0::nat)}" "{f ∈ C . ∀ v ∈ e . f v =
  also have "... = 2/(2 powi (int k))" using prob_c by simp
  also have "... = 2/(2* (2 powi ((int k) - 1)))" using assms(3)
    by (metis power_int_commutes power_int_minus_mult zero_neq_numeral)
  finally show "P.prob {f ∈ C . edge_is_monochromatic2 f e} = 2 powi (1 - int k)"
    by (simp add: power_int_diff)
qed
```

Identified Formalisation Challenges

How can we:

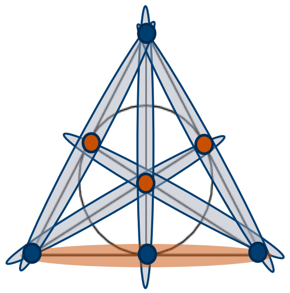
- **Shorten** the formal proof (mirroring natural proof)
- **Generalise techniques** used (formally)
- **Avoid 'hacking'** in future similar proofs

A first attempt at formalising a proof written in 1 line on paper!



The Basic Method

1. Introduce randomness to the problem domain
2. Identify the desired properties/properties to avoid
3. Show object has desired properties with $P > 0$
4. In a finite space, there must then be an element of the space with the property!



Applying the Method

Goal: Prove that every k-uniform hypergraph with fewer than 2^k-1 edges is 2-colourable

1. Colour a graph with 2 colours randomly
2. Property: colouring results in no edges being monochromatic.
3. Show the complement: probability of all edges being monochromatic < 1
4. P(A) = 1 – (¬A). Positive probability, and exemplar colouring can be obtained.

Formalisation Framework - Summary

Formal Framework

1. **Define a probability space**
2. Define object properties
3. *Calculate probability bounds*
4. **Obtain exemplar object**

Traditional Framework

1. Introduce randomness to the Problem Domain
2. Identify the desired properties/properties to avoid
3. Show object has desired properties with $P > 0$
4. In a finite space, there must then be an element of the space with the property!

The Formalisation Framework – Step 1

To "introduce randomness" we must define a probability space $(\Omega, \mathcal{F}, P)$ formally

Define the measure

Define the prob space

Useful lemmas

```
define C where  "C = (all_n_vertex_colourings_fun 2)"
let ?M = "uniform_count_measure C"
interpret P: prob_space ?M
  using assms(1) by (intro prob_space_uniform_count_measure)(simp_all add: C_def vertex
have sp: "space ?M = C"
  by (simp add: space_uniform_count_measure)
have sts: "P.events = Pow C" by (simp add: sets_uniform_count_measure)
have finE: "finite (set_mset E)" by simp
have finC: "finite C" using vertex_colourings_fun_fin C_def by simp
have Ccard: "card C = 2 powi (card 𝒱)" using count_vertex_colourings_fun C_def by auto
```

Can we generalise?

The Formalisation Framework – Step 1

To "introduce randomness" we must define a probability space $(\Omega, \mathcal{F}, P)$ formally

Define the measure

Define the prob space

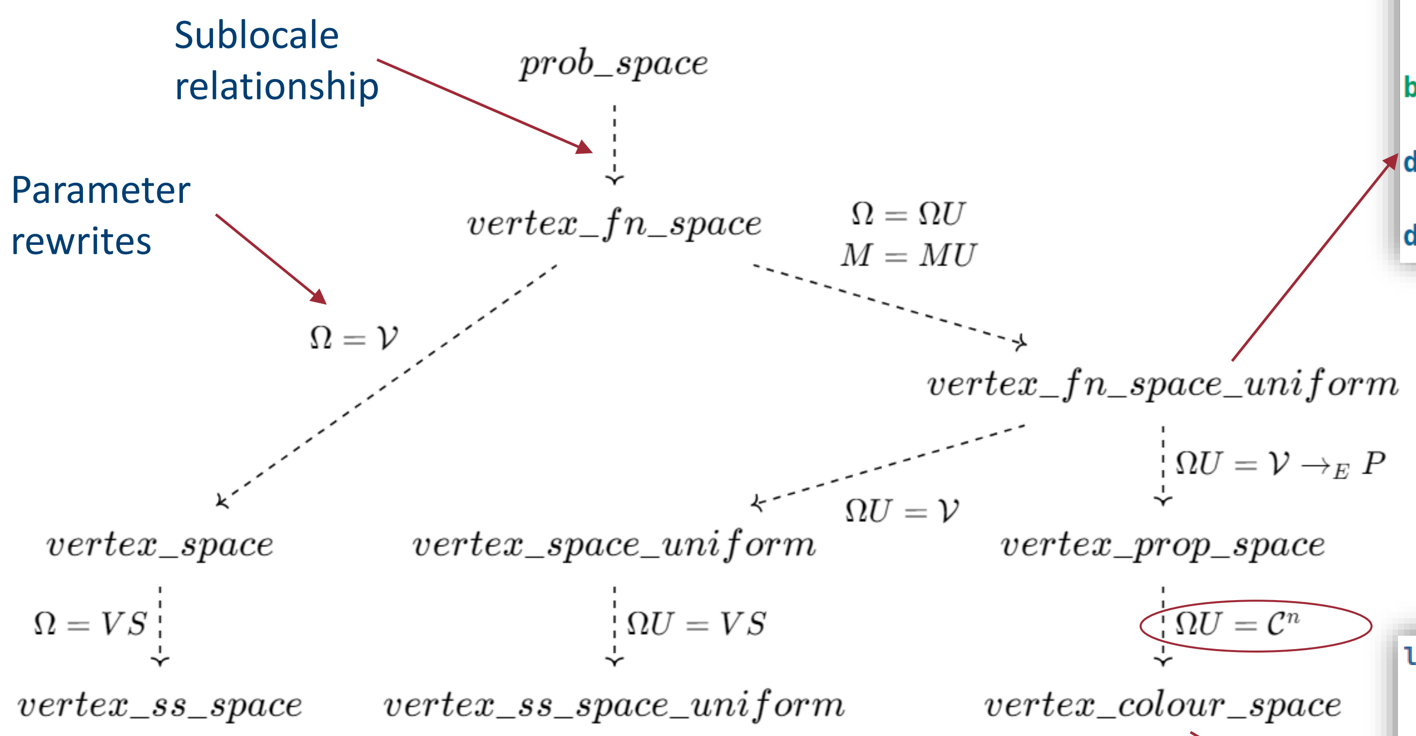
Useful lemmas

Local definitions?

```
define C where  "C = (all_n_vertex_colourings_fun 2)"
let ?M = "uniform_count_measure C"
interpret P: prob_space ?M
  using assms(1) by (intro prob_space_uniform_count_measure)(simp_all add: C_def vertex
have sp: "space ?M = C"
  by (simp add: space_uniform_count_measure)
have sts: "P.events = Pow C" by (simp add: sets_uniform_count_measure)
have finE: "finite (set_mset E)" by simp
have finC: "finite C" using vertex_colourings_fun_fin C_def by simp
have Ccard: "card C = 2 powi (card 𝒱)" using count_vertex_colourings_fun C_def by auto
```

Can we generalise?

The Formalisation Framework – Step 1 General!




```
locale vertex_fn_space_uniform = fin_hypersystem_vne +
  fixes F :: "'a set ⇒ 'b set"
  assumes ne: "F 𝒱 ≠ {}"
  assumes fin: "finite (F 𝒱)"
begin

definition "ΩU ≡ F 𝒱"

definition "MU ≡ uniform_count_measure ΩU"
```

```
locale vertex_colour_space = fin_hypergraph_nt +
  fixes n :: nat (*Number of colours *)
  assumes n_lt_order: "n ≤ horder"
  assumes n_not_zero: "n ≠ 0"


sublocale vertex_colour_space ⊆ vertex_prop_space 𝒱 E "{0..<n}"
  rewrites "ΩU = 𝒞↑n"
```

Application: A Vertex Colouring Space Example

```
locale vertex_colour_space = fin_hypergraph_nt +
  fixes n :: nat (*Number of colours *)
  assumes n_lt_order: "n ≤ order"
  assumes n_not_zero: "n ≠ 0"

sublocale vertex_colour_space ⊆ vertex_prop_space 𝒱 E "{0..<n}"
  rewrites "ΩU = 𝒞ⁿ"
proof -
  have "{0..<n} ≠ {}" using n_not_zero by simp
  then interpret vertex_prop_space 𝒱 E "{0..<n}"
    by (unfold_locales) (simp_all)
  show "vertex_prop_space 𝒱 E {0..<n}" by (unfold_locales)
  show "ΩU = 𝒞ⁿ"
    using Ω_def all_n_vertex_colourings_alt by auto
qed
```

Application: A Vertex Colouring Space Example

```
locale vertex_colour_space = fin_hypergraph_nt +
  fixes n :: nat (*Number of colours *)
  assumes n_lt_order: "n ≤ order"
  assumes n_not_zero: "n ≠ 0"

sublocale vertex_colour_space ⊆ vertex_prop_space 𝒱 E "{0..<n}"
  rewrites "ΩU = 𝒞n"
proof -
  have "{0..<n} ≠ {}" using n_not_zero by simp
  then interpret vertex_prop_space 𝒱 E "{0..<n}"
    by (unfold_locales) (simp_all)
  show "vertex_prop_space 𝒱 E {0..<n}" by (unfold_locales)
  show "ΩU = 𝒞n"
    using Ω_def all_n_vertex_colourings_alt by auto
qed
```

Locale context contains general lemmas on vertex colourings for any future applications of the probabilistic method to colourings!

The Formalisation Framework – Step 3

- The Union bound:

```
lemma Union_bound_avoid:
  assumes "finite A"
  assumes "(∑a ∈ A. prob a) < 1"
  assumes "A ⊆ events"
  shows "prob (space M - ⋃A) > 0"
```

Lemma "lifted" from measure theory libraries

- The Complete Independence Bound

```
lemma complete_indep_bound3:
  assumes "finite A"
  assumes "A ≠ {}"
  assumes "F ` A ⊆ events"
  assumes "indep_events F A"
  assumes "⋀ a . a ∈ A ⟹ prob (F a) < 1"
  shows "prob (⋂a ∈ A. space M - F a) > 0"
```

New formalisation which uses measure theory basics.

The Formalisation Framework – Step 4

- Obtaining an object from a probability!
- Some basic rules

```
lemma prob_lt_one_obtain:
  assumes "{e ∈ space M . Q e} ∈ events"
  assumes "prob {e ∈ space M . Q e} < 1"
  obtains e where "e ∈ space M" and "¬ Q e"
```

```
lemma prob_gt_zero_obtain:
  assumes "{e ∈ space M . Q e} ∈ events"
  assumes "prob {e ∈ space M . Q e} > 0"
  obtains e where "e ∈ space M" and "Q e"
```

- Combining steps 3 & 4!

```
lemma Union_bound_obtain_fun:
  assumes "finite A"
  assumes "(∑a ∈ A. prob (f a)) < 1"
  assumes "f ` A ⊆ events"
  obtains e where "e ∈ space M" and "e ∉ ⋃( f` A)"
```

3. More Probability Extensions (The Lovász Local Lemma (LLL))

LLL Background – Step 3

Given a set of bad events: $A = \{A_1, A_2, \ldots, A_n\}$, to avoid $\mathbb{P}(\bigcap_{i=1}^{n} \overline{A_i}) > 0$

Lovász Local Lemma
(Mutual Independence)

$$\mathbb{P}(A_i) < x_i \prod_{j \in N[i]} (1 - x_j)$$

N[i] = set of events A_i is mutually independent of

Complete Independence Bound

$$\mathbb{P}(A_i) < 1$$

All A_i are independent
$\therefore \mathrm{P}(A_i \cap A_j) = \mathrm{P}(A_i)\mathrm{P}(A_j)$

$$\mathbb{P}(A_i) < x_i \prod_{j \in N[i]} (1 - x_j) \xrightarrow{x_i \in (0,1)} \mathbb{P}\left(\bigcap_{i=1}^{n} \overline{A_i}\right) > 0 \xleftarrow{} \mathbb{P}(A_i) < 1$$

LLL Background – Step 3

Given a set of bad events: $A = \{A_1, A_2, \dots, A_n\}$, to avoid $\mathbb{P}(\bigcap_{i=1}^{n} \overline{A_i}) > 0$

Lovász Local Lemma
(Mutual Independence)

$$\mathbb{P}(A_i) < x_i \prod_{j \in N[i]} (1 - x_j)$$

$x_i \in (0,1)$

N[i] = set of events A_i is mutually independent of

Complete Independence Bound

$$\mathbb{P}(A_i) < 1$$

All A_i are independent
$\therefore \mathrm{P}(A_i \cap A_j) = \mathrm{P}(A_i)\mathrm{P}(A_j)$

$$\mathbb{P}\left(\bigcap_{i=1}^{n} \overline{A_i}\right) > 0$$

An event A is *mutually independent* of a set S if for any $T \subseteq S$,
$\mathbb{P}(A) = P(A \mid T)$

LLL Formal Theorem Statement – Step 3

Lemma 5.1.1 [The Local Lemma; General Case] *Let $A_1, A_2, \ldots, A_n$ be events in an arbitrary probability space. A directed graph $D = (V, E)$ on the set of vertices $V = \{1, 2, \ldots, n\}$ is called a dependency digraph for the events $A_1, \ldots, A_n$ if for each i, $1 \leq i \leq n$, the event A_i is mutually independent of all the events $\{A_j : (i,j) \notin E\}$. Suppose that $D = (V, E)$ is a dependency digraph for the above events and suppose there are real numbers $x_1, \ldots, x_n$ such that $0 \leq x_i < 1$ and* $\Pr[A_i] \leq x_i \prod_{(i,j)\in E}(1 - x_j)$ *for all $1 \leq i \leq n$. Then*

$$\Pr\left[\bigwedge_{i=1}^{n} \overline{A_i}\right] \geq \prod_{i=1}^{n}(1 - x_i).$$

In particular, with positive probability no event A_i holds.

```
theorem lovasz_local_general:
  assumes "A ≠ {}"
  assumes "F ` A ⊆ events"
  assumes "finite A"
  assumes "⋀ Ai . Ai ∈ A ⟹  f Ai ≥ 0 ∧ f Ai < 1"
  assumes "dependency_digraph G M F"
  assumes "⋀ Ai.  Ai ∈ A ⟹ (prob (F Ai) ≤ (f Ai) * (∏ Aj ∈ pre_digraph.neighborhood G Ai. (1 - (f Aj))))"
  assumes "pverts G = A"
  shows "prob (⋂ Ai ∈ A . (space M - (F Ai))) ≥ (∏ Ai ∈ A . (1 - f Ai))" "(∏ Ai ∈ A . (1 - f Ai)) > 0"
```

LLL Formal Theorem Statement – Step 3

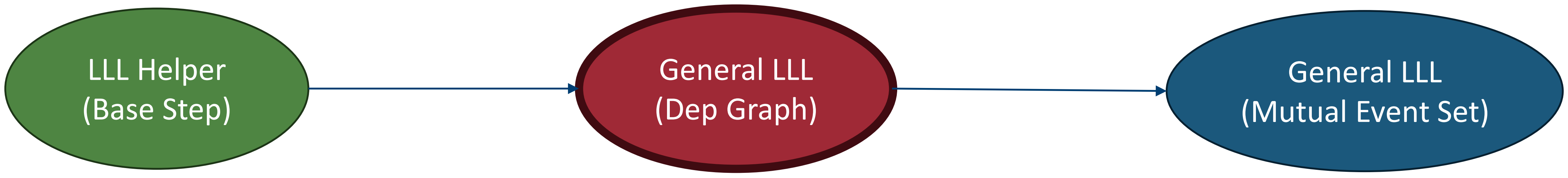
Lemma 5.1.1 [The Local Lemma; General Case] *Let $A_1, A_2, \ldots, A_n$ be events in an arbitrary probability space.* A1 *A directed graph $D = (V, E)$ on the set of vertices $V = \{1, 2, \ldots, n\}$ is called a dependency digraph for the events $A_1, \ldots, A_n$ if for each i, $1 \le i \le n$, the event A_i is mutually independent of all the events $\{A_j : (i,j) \notin E\}$. Suppose that $D = (V, E)$ is a dependency digraph for the above events* A2 *and suppose there are real numbers $x_1, \ldots, x_n$ such that $0 \le x_i < 1$ and* A3 *$\Pr[A_i] \le x_i \prod_{(i,j)\in E}(1 - x_j)$ for all $1 \le i \le n$.* A4 *Then*

$$\Pr\left[\bigwedge_{i=1}^{n} \overline{A_i}\right] \ge \prod_{i=1}^{n}(1 - x_i)\,. \quad \text{C1}$$

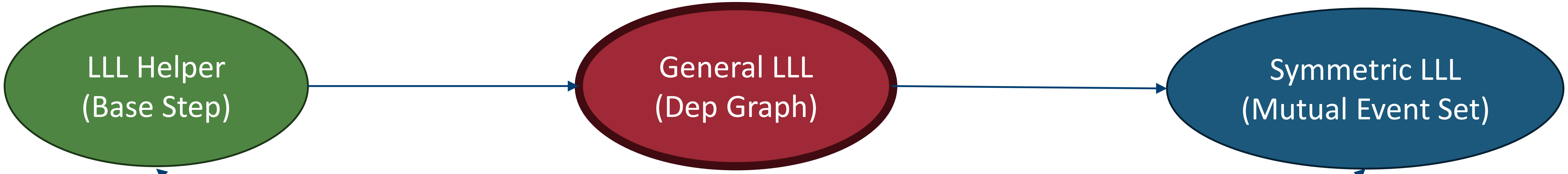
In particular, with positive probability no event A_i holds. C2

```
theorem lovasz_local_general:
  assumes "A ≠ {}"
  assumes "F ` A ⊆ events"                                   A1
  assumes "finite A"
  assumes "⋀ Ai . Ai ∈ A ⟹  f Ai ≥ 0 ∧ f Ai < 1"  A3
  assumes "dependency_digraph G M F"  A2
  assumes "⋀ Ai.  Ai ∈ A ⟹ (prob (F Ai) ≤ (f Ai) * (∏ Aj ∈ pre_digraph.neighborhood G Ai. (1 - (f Aj))))"  A4
  assumes "pverts G = A"  A2
  shows "prob (⋂ Ai ∈ A . (space M - (F Ai))) ≥ (∏ Ai ∈ A . (1 - f Ai))" "(∏ Ai ∈ A . (1 - f Ai)) > 0"
                        C1                                          C2
```

LLL Paper Proof Sketch – Step 3



LLL Paper Proof Sketch – Step 3



Proof. We first prove, by induction on s, that for any $S \subset \{1, \ldots, n\}$, $|S| = s < n$, and any $i \notin S$,

$$\Pr\left[A_i \mid \bigwedge_{j \in S} \overline{A_j}\right] \le x_i \,. \tag{5.1}$$

$|\{j : (i,j) \in E\}| \le d$. The result now follows from Lemma 5.1.1 by taking $x_i = 1/(d+1)$ (< 1) for all i and using the fact that for any $d \ge 1$,

$$\left(1 - \frac{1}{d+1}\right)^d > \frac{1}{e} \,.$$

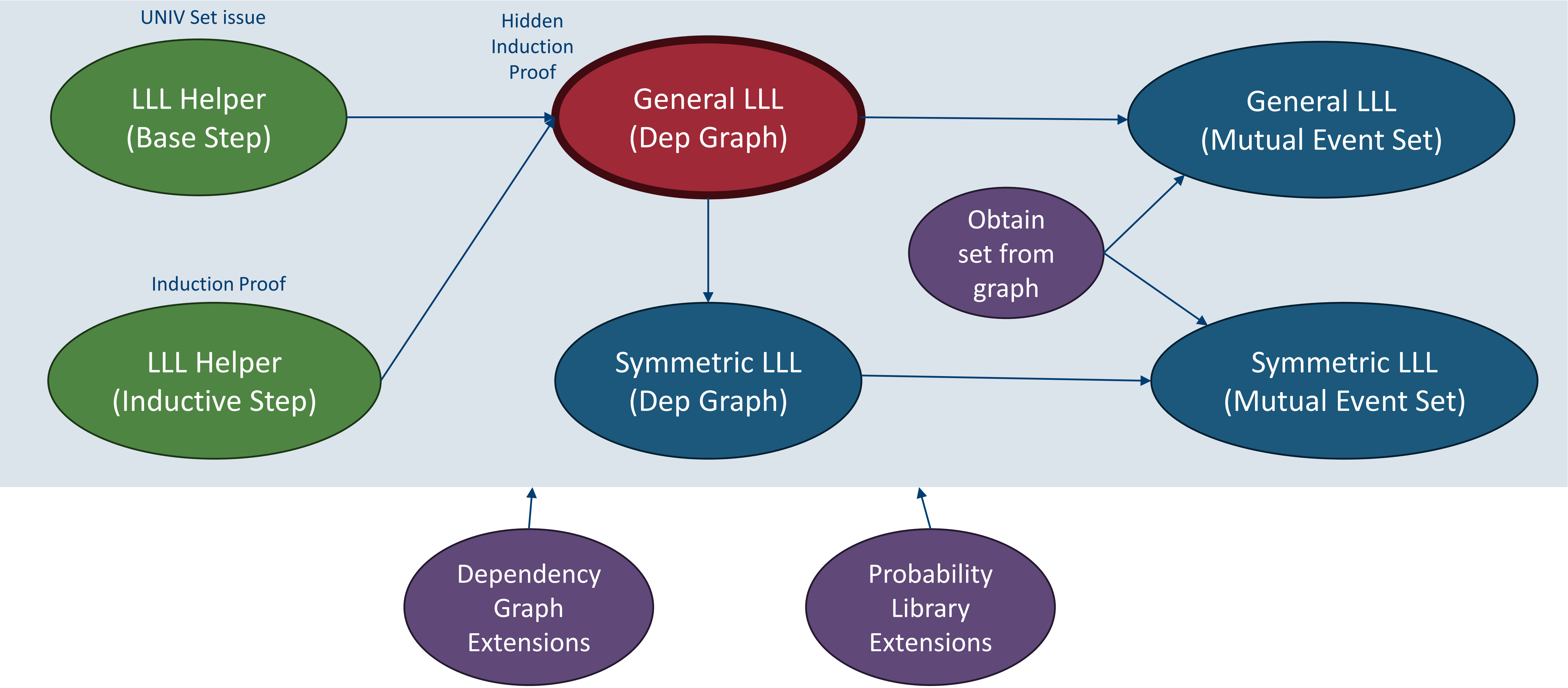
The assertion of Lemma 5.1.1 now follows easily, as

$$\Pr\left[\bigwedge_{i=1}^{n} \overline{A_i}\right] = (1 - \Pr[A_1]) \cdot (1 - \Pr[A_2 \mid \overline{A_1}]) \cdots$$

$$\cdots \left(1 - \Pr\left[A_n \mid \bigwedge_{i=1}^{n-1} \overline{A_i}\right]\right) \ge \prod_{i=1}^{n} (1 - x_i) \,,$$

completing the proof.

LLL Formal Proof Sketch – Step 3



LLL Symmetric – Step 3

Corollary 5.1.2 [The Local Lemma; Symmetric Case] *Let $A_1, A_2, \ldots, A_n$ be events in an arbitrary probability space. Suppose that each event A_i is mutually independent of a set of all the other events A_j but at most d, and that $\Pr[A_i] \leq p$ for all $1 \leq i \leq n$. If*

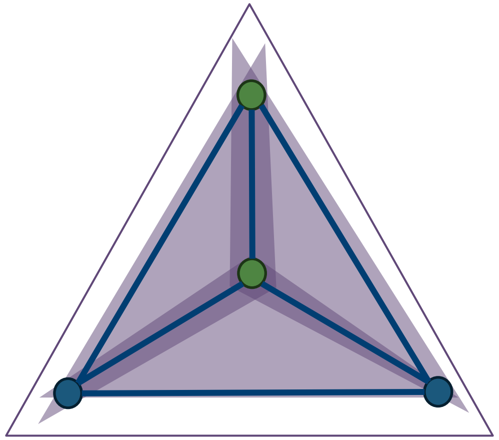
$$ep(d+1) \leq 1 \tag{5.5}$$

then $\Pr\left[\bigwedge_{i=1}^{n} \overline{A_i}\right] > 0$.

```
theorem lovasz_local_symmetric:
  fixes d :: nat
  assumes "A ≠ {}"
  assumes "F ` A ⊆ events"
  assumes "finite A"
  assumes "⋀ Ai. Ai ∈ A ⟹ (∃ S . S ⊆ A - {Ai} ∧ card S ≥ card A - d - 1 ∧ mutual_indep_events (F Ai) F S)"
  assumes "⋀ Ai. Ai ∈ A ⟹ prob (F Ai) ≤ p"
  assumes "exp(1)* p * (d + 1) ≤ 1"
  shows "prob (⋂ Ai ∈ A . (space M - (F Ai))) > 0"
proof -
  obtain G where odg: "dependency_digraph G M F" "pverts G = A" "⋀ Ai.  Ai ∈ A ⟹ out_degree G Ai ≤ d"
    using assms obtain_dependency_graph by metis
  then show ?thesis using odg assms lovasz_local_symmetric_dep_graph[of A F G d p] by auto
qed
```

5. Applications

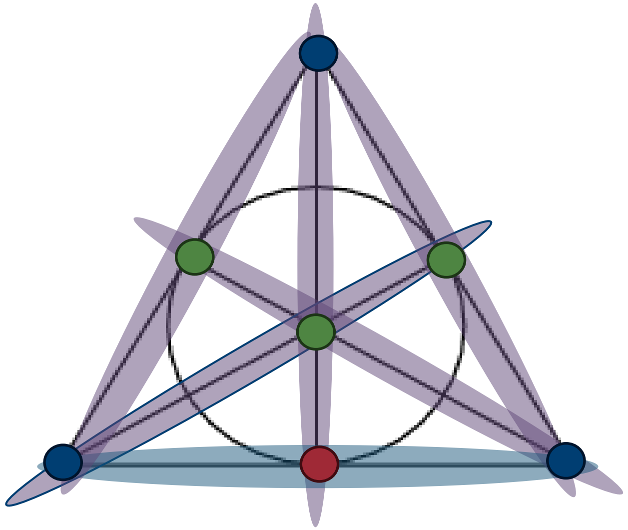
Application: Defining Hypergraph Colourings



```
abbreviation vertex_colouring :: "('a ⇒ colour) ⇒ nat ⇒ bool" where
"vertex_colouring f n ≡ f ∈ 𝒱 →E {0..<n}"

definition mono_edge :: "('a ⇒ colour) ⇒ 'a hyp_edge ⇒ bool" where
"mono_edge f e ≡ ∃ c. ∀ v ∈ e.  f v = c"

definition is_proper_colouring :: "('a ⇒ colour) ⇒ nat ⇒ bool" where
"is_proper_colouring f n ≡ vertex_colouring f n ∧ (∀ e ∈# E. ∀ c ∈ {0..<n}. f ` e ≠ {c})"
```



```
definition is_n_colourable :: "nat ⇒ bool" where
"is_n_colourable n ≡ ∃ f . is_proper_colouring f n"

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 ("(𝒞⇧_)" [502] 500)

abbreviation (in hypergraph) has_property_B :: "bool" where
"has_property_B ≡ is_n_colourable 2"
```

Application: A Vertex Colouring Space Example

```
locale vertex_colour_space = fin_hypergraph_nt +
  fixes n :: nat (*Number of colours *)
  assumes n_lt_order: "n ≤ order"
  assumes n_not_zero: "n ≠ 0"

sublocale vertex_colour_space ⊆ vertex_prop_space 𝒱 E "{0..<n}"
  rewrites "ΩU = 𝒞ⁿ"
proof -
  have "{0..<n} ≠ {}" using n_not_zero by simp
  then interpret vertex_prop_space 𝒱 E "{0..<n}"
    by (unfold_locales) (simp_all)
  show "vertex_prop_space 𝒱 E {0..<n}" by (unfold_locales)
  show "ΩU = 𝒞ⁿ"
    using Ω_def all_n_vertex_colourings_alt by auto
qed
```

Locale contexts contain general lemmas on useful probability calculations for vertex colourings

Application: The basic method

Proposition 1.3.1 [Erdős (1963a)] *Every n-uniform hypergraph with less than 2^{n-1} edges has property B. Therefore $m(n) \geq 2^{n-1}$.*

Proof. Let $H = (V, E)$ be an n-uniform hypergraph with less than 2^{n-1} edges. Color V randomly by two colors. For each edge $e \in E$, let A_e be the event that e is monochromatic. Clearly $\Pr[A_e] = 2^{1-n}$. Therefore

$$\Pr\left[\bigvee_{e \in E} A_e\right] \leq \sum_{e \in E} \Pr[A_e] < 1$$

and there is a two-coloring without monochromatic edges. ∎

```
context fin_kuniform_hypergraph_nt
begin
proposition erdos_propertyB:
  assumes "size E < (2^(k - 1))"
  assumes "k > 0"
  shows "has_property_B"
proof -
(* (1) Set up the probability space: "Colour V randomly with two colours" *)
  interpret P: vertex_colour_space 𝒱 E 2
    by unfold_locales (auto simp add: order_ge_two)
(* (2) define the event to avoid - monochromatic edges *)
  define A where "A ≡(λ e. {f ∈ 𝒞² . mono_edge f e})"
(* (3) Calculation 1: Clearly Pr[Ae] = 2^(1- n). *)
  have pe: "⋀ e. e ∈ set_mset E ⟹ P.prob {f ∈ 𝒞² . mono_edge f e} = 2 powi (1 - int k)"
    using P.prob_monochromatic_edge uniform assms(1) by fastforce
(* (3) Calculation 2: Have Pr (of Ae for any e) ≤ Sum over e (Pr (A e)) < 1 *)
  have "(∑e ∈ set_mset E. P.prob (A e)) < 1"
  proof -
    have "int k - 1 = int (k - 1)" using assms by linarith
    then have "card (set_mset E) < 2 powi (int k - 1)" using card_size_set_mset[of E] assms by simp
    then have "(∑e ∈ (set_mset E). P.prob (A e)) < 2 powi (int k - 1) * 2 powi (1 - int k)"
      unfolding A_def using pe by simp
    moreover have "((2 :: real) powi ((int k) - 1)) * (2 powi (1 - (int k))) = 1"
      using power_int_add[of 2 "int k - 1" "1- int k"] by force
    ultimately show ?thesis using power_int_add[of 2 "int k - 1" "1- int k"] by simp
  qed
  moreover have "A ` (set_mset E) ⊆ P.events" unfolding A_def P.sets_eq by blast
(* (4) obtain a colouring avoiding bad events *)
  ultimately obtain f where "f ∈ 𝒞²" and "f ∉ ⋃(A `(set_mset E))"
    using P.Union_bound_obtain_fun[of "set_mset E" A] finite_set_mset P.space_eq by auto
  thus ?thesis using event_is_proper_colouring A_def is_n_colourable_def by auto
qed
```

Application: the basic method

Proposition 1.3.1 **[Erdős (1963a)]** *Every n-uniform hypergraph with less than 2^{n-1} edges has property B. Therefore $m(n) \geq 2^{n-1}$.*

Proof. Let $H = (V, E)$ be an n-uniform hypergraph with less than 2^{n-1} edges. Color V randomly by two colors. For each edge $e \in E$, let A_e be the event that e is monochromatic. Clearly $\Pr[A_e] = 2^{1-n}$. Therefore

$$\Pr\left[\bigvee_{e \in E} A_e\right] \leq \sum_{e \in E} \Pr[A_e] < 1$$

and there is a two-coloring without monochromatic edges. ■

```
context fin_kuniform_hypergraph_nt
begin
proposition erdos_propertyB:
  assumes "size E < (2^(k - 1))"
  assumes "k > 0"
  shows "has_property_B"
proof -
(* (1) Set up the probability space: "Colour V randomly with two colours" *)
  interpret P: vertex_colour_space 𝒱 E 2
    by unfold_locales (auto simp add: order_ge_two)
(* (2) define the event to avoid - monochromatic edges *)
  define A where "A ≡(λ e. {f ∈ 𝒞² . mono_edge f e})"
(* (3) Calculation 1: Clearly Pr[Ae] = 2^(1- n). *)
  have pe: "⋀ e. e ∈ set_mset E ⟹ P.prob {f ∈ 𝒞² . mono_edge f e} = 2 powi (1 - int k)"
    using P.prob_monochromatic_edge uniform assms(1) by fastforce
(* (3) Calculation 2: Have Pr (of Ae for any e) ≤ Sum over e (Pr (A e)) < 1 *)
  have "(∑e ∈ set_mset E. P.prob (A e)) < 1"
  proof -
    have "int k - 1 = int (k - 1)" using assms by linarith
    then have "card (set_mset E) < 2 powi (int k - 1)" using card_size_set_mset[of E] assms by simp
    then have "(∑e ∈ (set_mset E). P.prob (A e)) < 2 powi (int k - 1) * 2 powi (1 - int k)"
      unfolding A_def using pe by simp
    moreover have "((2 :: real) powi ((int k) - 1)) * (2 powi (1 - (int k))) = 1"
      using power_int_add[of 2 "int k - 1" "1- int k"] by force
    ultimately show ?thesis using power_int_add[of 2 "int k - 1" "1- int k"] by simp
  qed
  moreover have "A ` (set_mset E) ⊆ P.events" unfolding A_def P.sets_eq by blast
(* (4) obtain a colouring avoiding bad events *)
  ultimately obtain f where "f ∈ 𝒞²" and "f ∉ ⋃(A `(set_mset E))"
    using P.Union_bound_obtain_fun[of "set_mset E" A] finite_set_mset P.space_eq by auto
  thus ?thesis using event_is_proper_colouring A_def is_n_colourable_def by auto
qed
```

Application: A more advanced bound

n-uniform hypergraph

Proposition 1.3.1 [Erdős (1963a)] *Every n-uniform hypergraph with less than 2^{n-1} edges has property B. Therefore $m(n) \geq 2^{n-1}$.*

Proof. Let $H = (V, E)$ be an n-uniform hypergraph with less than 2^{n-1} edges. Color V randomly by two colors. For each edge $e \in E$, let A_e be the event that e is monochromatic. Clearly $\Pr[A_e] = 2^{1-n}$. Therefore

$$\Pr\left[\bigvee_{e \in E} A_e\right] \leq \sum_{e \in E} \Pr[A_e] < 1$$

and there is a two-coloring without monochromatic edges. ■

Union Bound

Hypergraph w/ edge conditions

Theorem 5.2.1 *Let $H = (V, E)$ be a hypergraph in which every edge has at least k elements, and suppose that each edge of H intersects at most d other edges. If $e(d+1) \leq 2^{k-1}$ then H has property B.*

Proof. Color each vertex v of H, randomly and independently, either blue or red (with equal probability). For each edge $f \in E$, let A_f be the event that f is monochromatic. Clearly $\Pr[A_f] = 2/2^{|f|} \leq 1/2^{k-1}$. Moreover, each event A_f is clearly mutually independent of all the other events $A_{f'}$ for all edges f' that do not intersect f. The result now follows from Corollary 5.1.2. ■

Lovász Local Lemma

More advanced, but smaller proof?

Application: A more advanced bound

Theorem 5.2.1 *Let $H = (V, E)$ be a hypergraph in which every edge has at least k elements, and suppose that each edge of H intersects at most d other edges. If $e(d+1) \leq 2^{k-1}$ then H has property B.*

Proof. Color each vertex v of H, randomly and independently, either blue or red (with equal probability). For each edge $f \in E$, let A_f be the event that f is monochromatic. Clearly $\Pr[A_f] = 2/2^{|f|} \leq 1/2^{k-1}$. Moreover, each event A_f is clearly mutually independent of all the other events $A_{f'}$ for all edges f' that do not intersect f. The result now follows from Corollary 5.1.2. ■

```
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 -
  — ‹ 1 set up probability space ›
  interpret P: vertex_colour_space 𝒱 E 2 (* Reuse set up *)
    by unfold_locales (auto simp add: order_ge_two)
  let ?N = "{0..<size E}"
  obtain id where ideq: "image_mset id (mset_set ?N) = E" and idin: "id ∈ ?N →E set_mset E"
    using obtain_function_on_ext_funcset[of "?N" E] by auto
  then have iedge: "⋀i. i ∈ ?N ⟶ id i ∈# E" by auto
  — ‹2 define event ›
  define Ae where "Ae ≡ λ i. {f ∈ 𝒞⇧2 . mono_edge f (id i)}"
  — ‹ (3)  Prove each event A is mutually independent of all other mono events for other
    edges that don't intersect. ›
  have "0 < P.prob (⋂Ai∈?N. space P.MU - Ae Ai)"
  proof (intro P.lovasz_local_symmetric[of ?N Ae d "(1/(2 powi (k-1)))"])
    have mis: "⋀i . i ∈ ?N ⟶ P.mutual_indep_events (Ae i) Ae ({j ∈?N . (id j ∩ id i) = {}})"
      using disjoint_set_is_mutually_independent[of _ id Ae] P.MU_def assms idin by (simp add: Ae_def)
    then show "⋀ i . i ∈ ?N ⟶ ∃ S. S ⊆ ?N - {i} ∧ card S ≥ card ?N - d - 1 ∧
      P.mutual_indep_events (Ae i) Ae S"
    proof -
       … (5 lines)
    qed
    show "⋀ i. i ∈ ?N ⟶ P.prob(Ae i) ≤ 1/(2 powi (k-1))"
      unfolding Ae_def using P.prob_monochromatic_edge_bound[of _ k] iedge assms(4) assms(1) by auto
    show "exp(1) * (1 / 2 powi int (k - 1)) * (d + 1) ≤ 1"
      using assms(3) by (simp add: field_simps del:One_nat_def)
      (metis Num.of_nat_simps(2) assms(4) diff_is_0_eq diff_less less_one of_nat_diff power_int_of_nat)
  qed (auto simp add: Ae_def E_nempty P.sets_eq P.space_eq)
  — ‹ 4 obtain ›
  then obtain f where fin: "f ∈ 𝒞⇧2" and "⋀ i. i ∈ ?N ⟶ ¬ mono_edge f (id i)" using  Ae_def
    P.obtain_intersection_prop[of Ae ?N "λ f i. mono_edge f (id i)"] P.space_eq P.sets_eq by auto
  then have "⋀ e. e ∈# E ⟶ ¬ mono_edge f e"
    using  ideq mset_set_implies[of id ?N E "λ e. ¬ mono_edge f e"] by blast
  then show ?thesis unfolding is_n_colourable_def
    using is_proper_colouring_alt2 fin all_n_vertex_colourings_def[of 2] by auto
qed
```

Application: A more advanced bound

Theorem 5.2.1 *Let $H = (V, E)$ be a hypergraph in which every edge has at least k elements, and suppose that each edge of H intersects at most d other edges. If $e(d+1) \leq 2^{k-1}$ then H has property B.*

Proof. Color each vertex v of H, randomly and independently, either blue or red (with equal probability). For each edge $f \in E$, let A_f be the event that f is monochromatic. Clearly $\Pr[A_f] = 2/2^{|f|} \leq 1/2^{k-1}$. Moreover, each event A_f is clearly mutually independent of all the other events $A_{f'}$ for all edges f' that do not intersect f. The result now follows from Corollary 5.1.2. ■

```
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 -
  — ‹ 1 set up probability space ›
  interpret P: vertex_colour_space 𝒱 E 2 (* Reuse set up *)
    by unfold_locales (auto simp add: order_ge_two)
  let ?N = "{0..<size E}"
  obtain id where ideq: "image_mset id (mset_set ?N) = E" and idin: "id ∈ ?N →E set_mset E"
    using obtain_function_on_ext_funcset[of "?N" E] by auto
  then have iedge: "⋀i. i ∈ ?N ⟶ id i ∈# E" by auto
  — ‹2 define event ›
  define Ae where "Ae ≡ λ i. {f ∈ 𝒞⇧2 . mono_edge f (id i)}"
  — ‹ (3)  Prove each event A is mutually independent of all other mono events for other
    edges that don't intersect. ›
  have "0 < P.prob (⋂Ai∈?N. space P.MU - Ae Ai)"
  proof (intro P.lovasz_local_symmetric[of ?N Ae d "(1/(2 powi (k-1)))"])
    have mis: "⋀i . i ∈ ?N ⟶ P.mutual_indep_events (Ae i) Ae ({j ∈?N . (id j ∩ id i) = {}})"
      using disjoint_set_is_mutually_independent[of _ id Ae] P.MU_def assms idin by (simp add: Ae_def)
    then show "⋀ i . i ∈ ?N ⟶ ∃ S. S ⊆ ?N - {i} ∧ card S ≥ card ?N - d - 1 ∧
      P.mutual_indep_events (Ae i) Ae S"
    proof -
       ... (5 lines)
    qed
    show "⋀ i. i ∈ ?N ⟶ P.prob(Ae i) ≤ 1/(2 powi (k-1))"
      unfolding Ae_def using P.prob_monochromatic_edge_bound[of _ k] iedge assms(4) assms(1) by auto
    show "exp(1) * (1 / 2 powi int (k - 1)) * (d + 1) ≤ 1"
      using assms(3) by (simp add: field_simps del:One_nat_def)
      (metis Num.of_nat_simps(2) assms(4) diff_is_0_eq diff_less less_one of_nat_diff power_int_of_nat)
  qed (auto simp add: Ae_def E_nempty P.sets_eq P.space_eq)
  — ‹ 4 obtain ›
  then obtain f where fin: "f ∈ 𝒞⇧2" and "⋀ i. i ∈ ?N ⟶ ¬ mono_edge f (id i)" using  Ae_def
    P.obtain_intersection_prop[of Ae ?N "λ f i. mono_edge f (id i)"] P.space_eq P.sets_eq by auto
  then have "⋀ e. e ∈# E ⟶ ¬ mono_edge f e"
    using  ideq mset_set_implies[of id ?N E "λ e. ¬ mono_edge f e"] by blast
  then show ?thesis unfolding is_n_colourable_def
    using is_proper_colouring_alt2 fin all_n_vertex_colourings_def[of 2] by auto
qed
```

+ Mutual Independence Principle!!!

Mutual Independence Principle

The Mutual Independence Principle *Suppose that* $\mathcal{X} = X_1, \ldots, X_m$ *is a sequence of independent random experiments. Suppose further that* $A_1, \ldots, A_n$ *is a set of events, where each* A_i *is determined by* $F_i \subseteq \mathcal{X}$. *If* $F_i \cap (F_{i_1}, \ldots, F_{i_k}) = \emptyset$ *then* A_i *is mutually independent of* $\{A_{i_1}, \ldots, A_{i_k}\}$.

Claim: Each event A_e is mutually independent of the set of events $\{A_f : f \notin N_e\} \cup A_e$.

Edge index for Ae

Edge index for Ae

```
lemma disjoint_set_is_mutually_independent:
  assumes iin: "i ∈ {0..<(size E)}"
  assumes idffn: "idf ∈ {0..<size E} →E set_mset E"
  assumes Aefn: "⋀ i. i ∈ {0..<size E} ⟹ Ae i = {f ∈ 𝒞⇑2 . mono_edge f (idf i)}"
  shows "prob_space.mutual_indep_events (uniform_count_measure (𝒞⇑2)) (Ae i) Ae
    ({j ∈{0..<(size E)} . (idf j ∩ idf i) = {}})"
```

5. Formalisation Insights

Theme 1: Intuition in Probabilistic Proofs

Intuition is everywhere in probability proofs for combinatorics

Introducing Randomness

"Obtaining" an exemplar structure

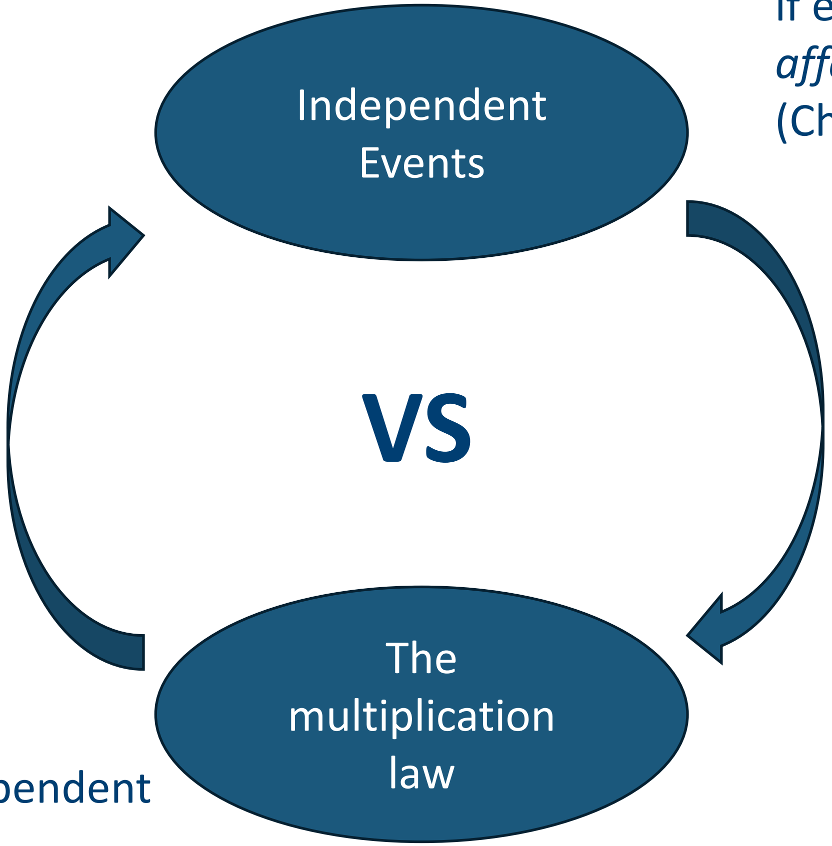
Many calculation specific examples

General framework = partial solution

Theme 1: Intuition in Probabilistic Proofs

Two events are said to be independent if either can occur *without being affected by the occurrence of the other* (Chalmers, 2018).

Formally defining independent events



Physical intuition to establish independence

The basic multiplication law for independent events states: $\mathrm{P}(\mathrm{A} \cap \mathrm{B}) = \mathrm{P}(\mathrm{A})\mathrm{P}(\mathrm{B})$

Theme 1: Intuition in Probabilistic Proofs

Clearly $\mathbf{Pr}[A_e] = 2^{1-n}$

- i.e. Clearly vertex colouring events are independent, so we can just apply P(AB) = P(A)P(B) right?
- **BUT - This is circular reasoning!**
  - To establish independence, we must prove the multiplication rule holds.
  - Use a counting lemma instead on sets of functions

```
lemma prob_edge_colour:
  assumes "e ∈# E" "c ∈ {0..<n}"
  shows "prob {f ∈ 𝒞ⁿ . mono_edge_col f e c} = 1/(n powi (card e))"
proof -
  have "card {0..<n} = n" by simp
  moreover have "𝒞ⁿ = 𝒱 →E {0..<n}" using all_n_vertex_colourings_alt by blast
  moreover have "{0..<n} ≠ {}" using n_not_zero by simp
  ultimately show ?thesis using prob_uniform_ex_fun_space[of 𝒱 _ "{0..<n}" e] n_not_zero
      finite_sets wellformed assms by (simp add: MU_def V_nempty mono_edge_col_def)
qed
```

Theme 2: An Isabelle Probability Library Issue

The issue	For example...	Therefore ...
$\Omega \neq \mathbb{U}$ in Isabelle	$\Omega =$ all vertex colouring functions $\mathbb{U}$ = all functions of type $'a \Rightarrow nat$	$\mathbb{P}\left(\bigcap \emptyset\right) = \mathbb{P}(\mathbb{U}) = \mathbf{0}$
		(in Isabelle)
Typically $\Omega \subset \mathbb{U}$	Clearly have $\Omega \subset \mathbb{U}$	$\neq$
$\mathbb{P}(\mathbb{U}) = 0 \neq 1 = \mathbb{P}(\Omega)$	$\mathbb{P}(\mathbb{U}) = 0 \neq 1 = \mathbb{P}(\Omega)$	$\mathbb{P}\left(\bigcap \emptyset\right) = \mathbb{P}(\Omega) = \mathbf{1}$
		(on paper proofs)

The first time in four years that simple type theory caused me issues…

Solutions: pmf library for discrete results, rework probability definitions, proof work arounds

Theme 3: The importance of generality

- **Modular**, **reusable,** and **extensible** formal libraries
- Reducing duplication in formal libraries
- This work's general framework -> successfully minimised rework.
  - Interesting new use case of Isabelle's **locales.**
- General efforts -> formalise new proofs more naturally.
- Balance **practicality** vs **generality** for ultimate **usability**

Concluding Thoughts

- Key contributions:
  - Significant expansions to **probability and hypergraph libraries** in Isabelle/HOL, including the **LLL.**
  - The **general framework** from the probabilistic method
  - Formal proof of the "**Mutual independence principle**" (for hypergraphs).
  - Applications: formalised several bounds on **hypergraph colourings**.
- Key lessons:
  - Generalisation is both possible & important in formalisation, **particularly as formal mathematical libraries grow at a rapid pace!** The up-front time investment is worth it.
  - Several insights into the mathematical intuition around probabilistic proofs.

Concluding Thoughts

- Other Applications?
  - Already some success with simplifying Balog-Szemeredi-Gowers proof
- Paper published at CPP2024: https://dl.acm.org/doi/10.1145/3636501.3636946
- Full AFP Formalisations available online:
  - https://www.isa-afp.org/entries/Hypergraph_Basics.html
  - https://www.isa-afp.org/entries/Lovasz_Local.html
  - https://www.isa-afp.org/entries/Hypergraph_Colourings.html

Contact: c.l.edmonds@sheffield.ac.uk | cledmonds.github.io