# Computer Algebra and the Formalisation of New Mathematics

Lawrence Paulson, Computer Laboratory, University of Cambridge

*Formalisation Seminar, DPMMS, 21 November 2024*

# Computer Algebra and Formal Proof

# *A* and *B* are both nice: why not *A*&*B*?

*Three decades* of trying to combine CA and ATP

“Computer algebra is unsound”

“CA tools can’t reason logically”

*Approaches*: certificates, tightly constrained oracles, reflection

Often missing: a compelling ***application***

# Computer algebra techniques within Isabelle/HOL

- Differentiation and integration
- Automatic asymptotic/limit proofs
- Arbitrary precision calculations by interval arithmetic
- Real & complex root-finding, counting winding numbers, and other specialised proof methods

# Symbolic differentiation

Let's differentiate $e^{-t}\cos(2\pi t)$ by proof alone

```
lemma "∃f'. ((λx. exp(-x)*cos(2*pi*x)) has_real_derivative f' t) (at t) ∧ P(f' t)" for t
  apply (rule exI conjI derivative_eq_intros)+
```

(just a partial step to reveal what's going on:)

```
goal (6 subgoals):
 1. 1 = ?f'15
 2. - ?f'15 = ?Db11
 3. exp (- t) * ?Db11 = ?Da6
 4. ((λx. cos (2 * pi * x)) has_real_derivative ?Db6) (at t)
 5. ?Da6 * cos (2 * pi * t) + ?Db6 * exp (- t) = ?f' t
 6. P (?f' t)
```

To do it properly, we must supply a *tactic* to prove the equality subgoals

```
lemma "∃f'. ((λx. exp(-x)*cos(2*pi*x)) has_real_derivative f' t) (at t) ∧ P(f' t)" for t
  apply (rule exI conjI derivative_eq_intros | force)+
```

The result is (sometimes) even simplified!

```
goal (1 subgoal):
 1. P (- (exp (- t) * cos (2 * pi * t)) -
       sin (2 * pi * t) * (2 * pi) * exp (- t))
```

$$-e^{-t}\cos(2\pi t) - \sin(2\pi t)\cdot 2\pi e^{-t}$$

# Symbolic integration (cheating with Maple)

$$x^2 \cdot \cos(4 \cdot x) \xrightarrow{\text{integrate w.r.t. x}} \frac{x^2 \sin(4x)}{4} - \frac{\sin(4x)}{32} + \frac{x \cos(4x)}{8}$$

Just ask Isabelle to *check* Maple by taking the derivative:

```
lemma "((λx. x^2 * sin(4*x)/4 - sin(4*x)/32 + x * cos(4*x)/8)
        has_real_derivative x^2 * cos(4*x)) (at x)" for x
  apply (rule exI conjI derivative_eq_intros refl | force)+
```

# This time, the output is ugly

```
goal (1 subgoal):
 1. ((real 2 * (1 * x ^ (2 - Suc 0)) * sin (4 * x) +
       cos (4 * x) * (0 * x + 1 * 4) * x²) *
      4 -
      x² * sin (4 * x) * 0) /
     (4 * 4) -
     (cos (4 * x) * (0 * x + 1 * 4) * 32 - sin (4 * x) * 0) /
     (32 * 32) +
     ((1 * cos (4 * x) + - sin (4 * x) * (0 * x + 1 * 4) * x) * 8 -
      x * cos (4 * x) * 0) /
     (8 * 8) =
     x² * cos (4 * x)
```

… but easy to fix:

```
apply (simp add: field_simps)
done
```

We can even evaluate *definite integrals* via the fundamental theorem of calculus

# Eberl's real asymptotics package

- Proves claims about **limits**, properties holding **in the limit**, claims involving **Landau symbols**
- … by computing *multiseries expansions* for a variety of real-valued functions (cf Richardson et al., 1996).
- All by inference alone!

$$\lim_{x \to 0} \frac{1 - \frac{1}{2}x^2 - \cos\left(\frac{x}{1-x^2}\right)}{x^4} = \frac{23}{24}$$

```
lemma "(λx::real. (1 - 1/2 * x^2 - cos (x / (1 - x^2))) / x^4) ─0→ 23/24"
  by real_asymp
```

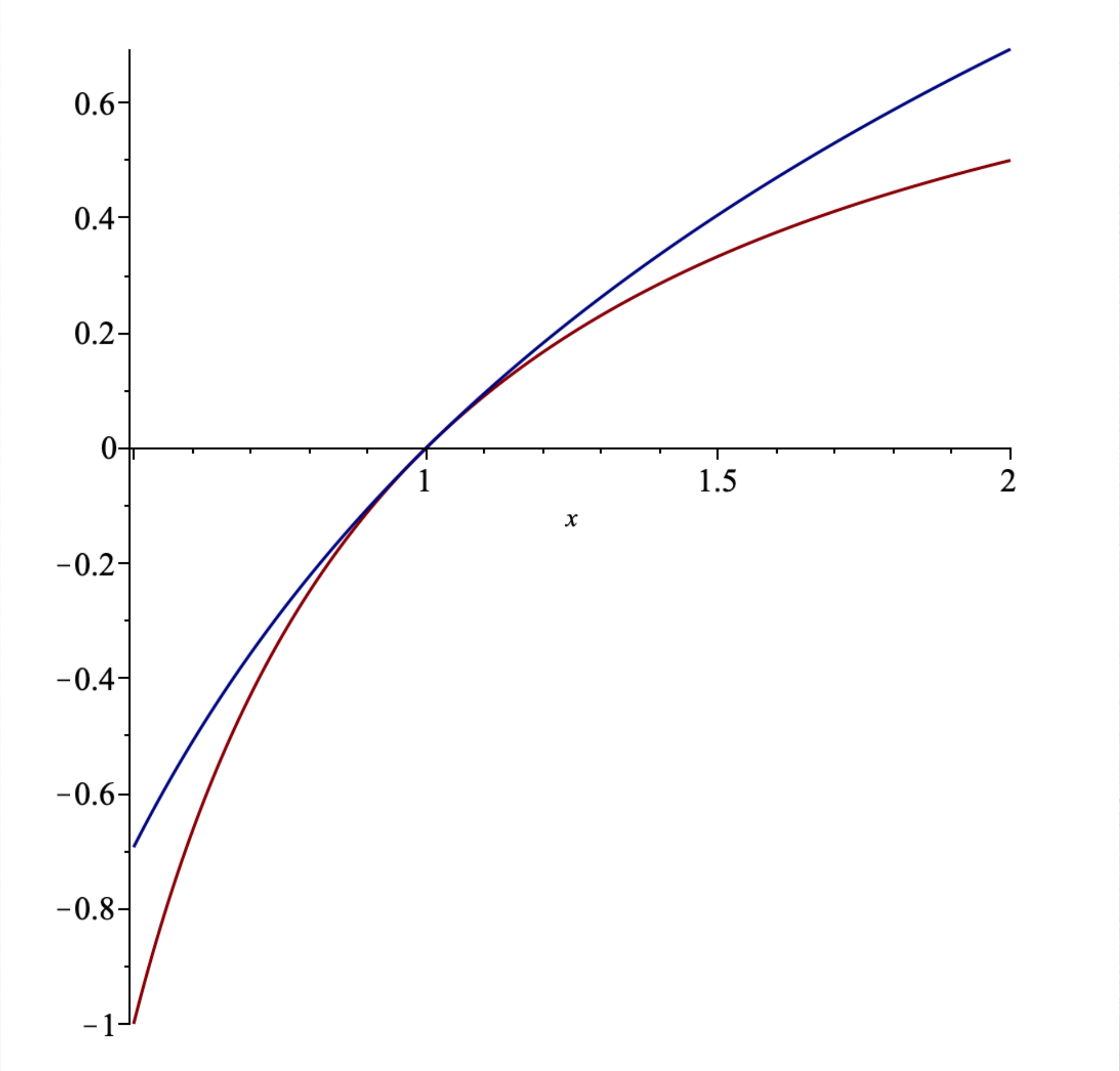
$$n^k = o(c^n)$$

```
lemma "c > 1 ⟹ (λn. real n ^ k) ∈ o(λn. c^n)"
  by real_asymp
```

# Can even do one-sided limits

```
lemma "eventually (λx::real. 1 - 1/x ≤ ln(x)) (at_right 0)"
  by real_asymp
```

$$1 - \frac{1}{x} \le \ln(x) \quad \text{as } x \to 0^+$$



# Exact numeric calculations

Simple inequalities:

```
lemma "¦ sin 100 + 0.50636564110975879 ¦ < (inverse 10 ^ 17 :: real)"
  by (approximation 70)
```

Inequalities over a range of inputs:

```
lemma "0.5 ≤ x ∧ x ≤ 4.5 ⟹ ¦ arctan x - 0.91 ¦ < 0.455"
  by (approximation 10)
```

Going beyond interval arithmetic:

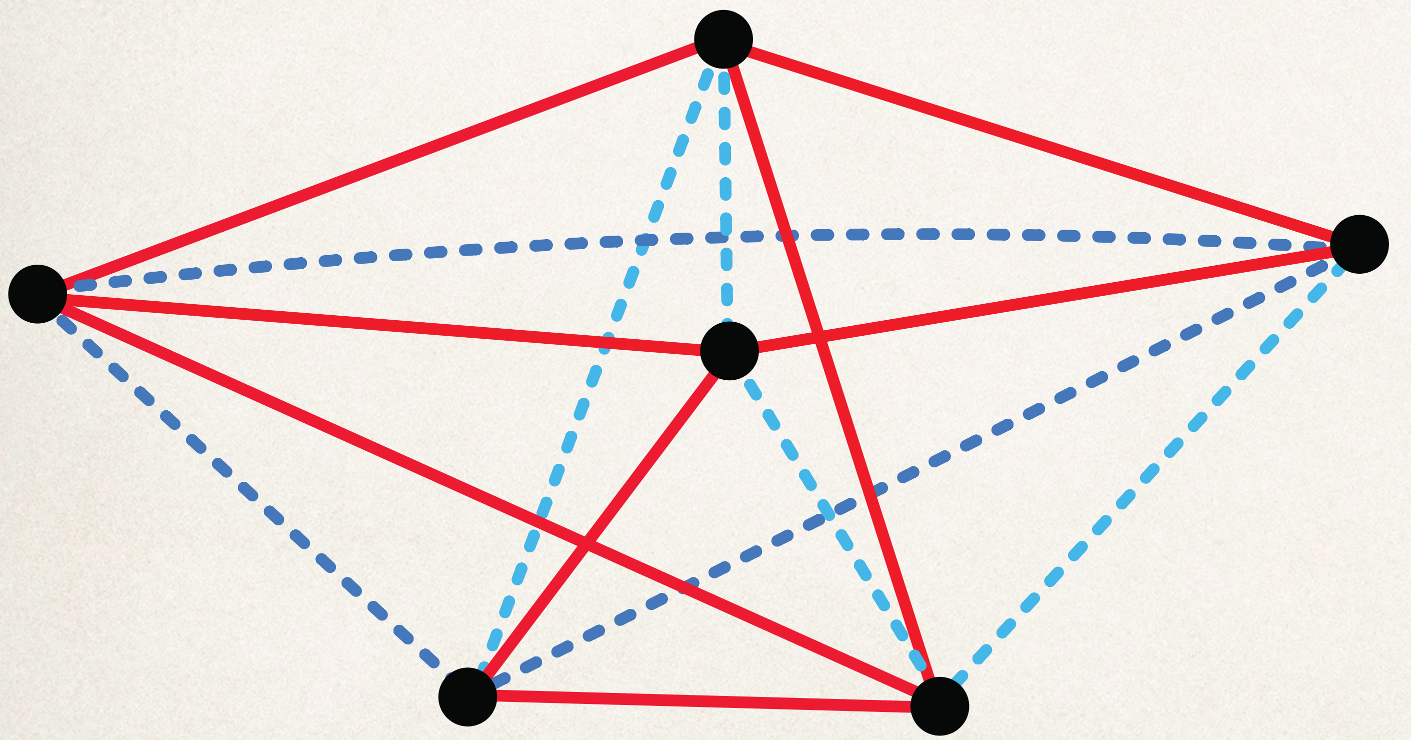
```
lemma "x ∈ { 0 .. 1 :: real } ⟶ x² ≤ x"
  by (approximation 30 splitting: x=1 taylor: x = 3)
```

# But is there an application?

# Ramsey's Theorem

# “Party Problem” version (2-sets)

For all $m$ and $n$ there exists a number $R(m, n)$ such that every graph with at least $R(m, n)$ vertices contains a *clique* of size $m$ or an *anti-clique* of size $n$



**Or**: a **complete** graph with edges coloured red / blue contains a *red clique* of size $m$ or a *blue clique* of size $n$

$$R(3,3) = 6$$

# Other forms (for $r$-sets, $r \neq 2$)

$r = 1$: Ramsey's theorem is just the *pigeonhole principle*

$r > 2$: hypergraph form, with unimaginably large *Ramsey numbers*

The $r = 2$ case can also be generalised with *transfinite ordinals or cardinals*

# Ramsey numbers

$R(3,3) = 6$

$R(4,4) = 18$

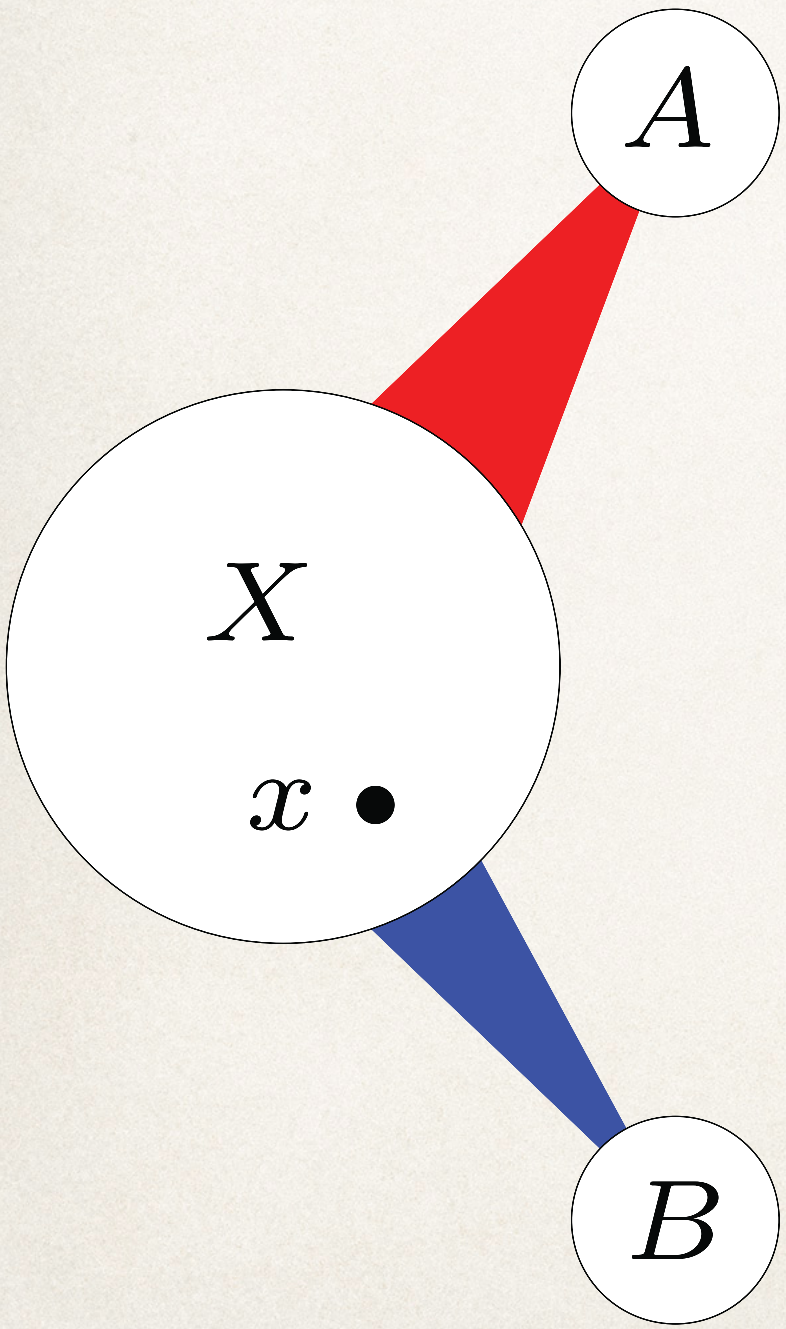
$43 \leq R(5,5) \leq 46$

Erdős (with Szekeres for the upper bound) proved

$$\sqrt{2}^k \leq R(k,k) \leq \binom{2k-2}{k-1} < 4^k$$

**A new result** replaces 4 by $4 - \epsilon$, an exponential improvement

# "Algorithm" to prove the $4^k$ bound




*At start*: put all vertices in $X$; set $A = B = \{\}$

$$X \to N_R(x) \cap X \qquad A \to A \cup \{x\}$$

if $x$ has more red neighbours than blue in $X$

$$X \to N_B(x) \cap X \qquad B \to B \cup \{x\}$$

otherwise

Builds a red clique in $A$, a blue clique in $B$

- At each step, choose the vertex $x$ arbitrarily
  - … the set $X$ loses up to half its vertices
  - … there are only red edges to $A$, blue edges to $B$
- If $|X| \geq 2^{k+\ell}$ then iteration finally yields a clique: either $|A| \geq k$ or $|B| \geq \ell$
- In the "diagonal" case $k = \ell$, the upper bound is $4^k$

*Could a more sophisticated algorithm do better?*

# A New Paper on Ramsey's Theorem

# AN EXPONENTIAL IMPROVEMENT FOR DIAGONAL RAMSEY

MARCELO CAMPOS, SIMON GRIFFITHS, ROBERT MORRIS, AND JULIAN SAHASRABUDHE

ABSTRACT. The Ramsey number $R(k)$ is the minimum $n \in \mathbb{N}$ such that every red-blue colouring of the edges of the complete graph $K_n$ on $n$ vertices contains a monochromatic copy of $K_k$. We prove that

$$R(k) \leqslant (4 - \varepsilon)^k$$

for some constant $\varepsilon > 0$. This is the first exponential improvement over the upper bound of Erdős and Szekeres, proved in 1935.

First formalised, in Lean, by Bhavik Mehta:
before the referees had completed their reviews!

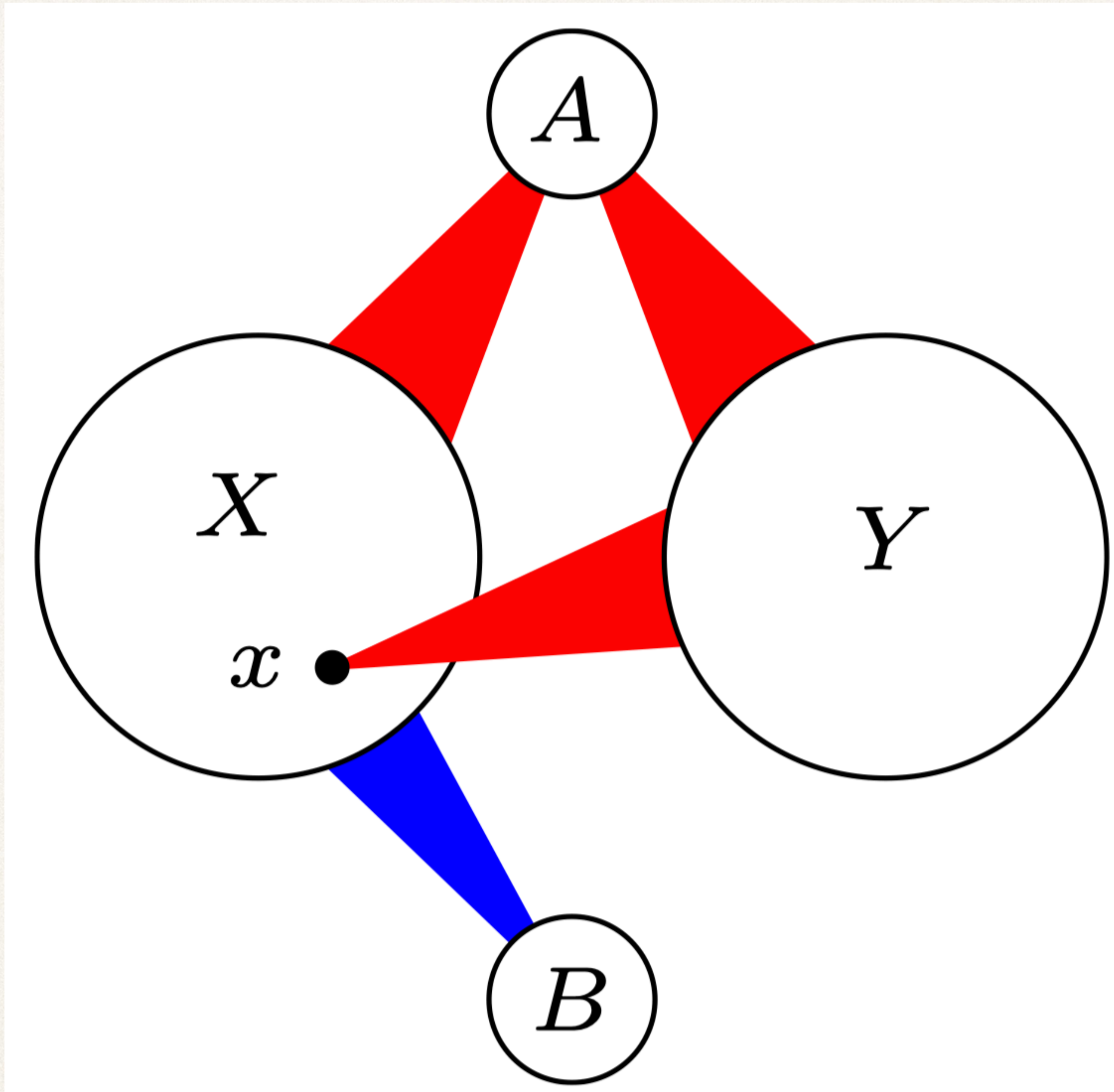
# What's the mathematics like?

- A more complicated "book algorithm"
- A string of technical lemmas describing its behaviour
  - Numerous estimates with finite sums / products
  - Numeric parameters; high-precision calculations
  - **Lots and lots of limit arguments**

*And it's 57 pages*

# The variables and their constraints

- Integers $\ell \leqslant k$ and a complete $n$-graph
- Edge colouring with no red $k$-clique, no blue $\ell$-clique
- Sets of vertices $X$, $Y$, $A$, $B$, the latter two initially empty
- All edges between $A$ and $A$, $X$, $Y$ are red
- All edges between $B$ and $B$, $X$ are blue


A
X
x
Y
B

# Some mathematical preliminaries

Standard definitions for undirected graphs

As $X$ and $Y$ evolve, need to maintain a sufficient *red density*

$$p = \frac{e_R(X, Y)}{|X||Y|}$$

Algorithm tries to build a large red clique in $A$

# The main execution steps

- *Degree regularisation*: remove from $X$ all vertices with "few" red neighbours in $Y$
- *Big blue step*: If there exist $R(k, \lceil \ell^{2/3} \rceil)$ vertices in $X$ with "lots" of blue neighbours in $X$, move a block of them into $B$ while leaving just their blue neighbours in $X$
- *Red* and *density-boost* steps: an element of $X$ with "few" blue neighbours in $X$ is moved into $A$ or into $B$, according to the red density of the resulting $X$ and $Y$

# A red or density-boost step

$$X \to N_R(x) \cap X \qquad Y \to N_R(x) \cap Y \qquad A \to A \cup \{x\}$$

*versus*

$$X \to N_B(x) \cap X \qquad Y \to N_R(x) \cap Y \qquad B \to B \cup \{x\}$$

resembles the basic algorithm,
except that $x$ is carefully selected

# A Glimpse at the Proofs

# Defining the "book algorithm"

```
definition next_state :: "[real,nat,nat,'a config] ⇒ 'a config" where
  "next_state ≡ λμ l k (X,Y,A,B).
       if many_bluish μ l k X
       then let (S,T) = choose_blue_book μ (X,Y,A,B) in (T, Y, A, B∪S)
       else let x = choose_central_vx μ (X,Y,A,B) in
            if reddish k X Y (red_density X Y) x
            then (Neighbours Red x ∩ X, Neighbours Red x ∩ Y, insert x A, B)
            else (Neighbours Blue x ∩ X, Neighbours Red x ∩ Y, A, insert x B)"
```

```
primrec stepper :: "[real,nat,nat,nat] ⇒ 'a config" where
  "stepper μ l k 0 = (X0,Y0,{},{})"
| "stepper μ l k (Suc n) =
     (let (X,Y,A,B) = stepper μ l k n in
      if termination_condition l k X Y then (X,Y,A,B)
      else if even n then degree_reg k (X,Y,A,B) else next_state μ l k (X,Y,A,B))"
```

*Many routine properties easily proved*

# A proof in more detail: Lemma 4.1

**Lemma 4.1.** *Set* $b = \ell^{1/4}$. *If there are* $R(k, \ell^{2/3})$ *vertices* $x \in X$ *such that*

$$|N_B(x) \cap X| \geqslant \mu |X|, \tag{9}$$

*then* $X$ *contains either a red* $K_k$, *or a blue book* $(S, T)$ *with* $|S| \geqslant b$ *and* $|T| \geqslant \mu^{|S|}|X|/2$.

Four weeks, 354 lines and several buckets of sweat later…

```
lemma Blue_4_1:
  assumes "X⊆V" and manyb: "many_bluish X" and big: "Big_Blue_4_1 μ l"
  shows "∃S T. good_blue_book X (S,T) ∧ card S ≥ l powr (1/4)"
```

[The claim holds for **sufficiently large** $\ell$ and $k$]

# What did I do in those three weeks?

Proved the Erdős lower bound for Ramsey numbers, $2^{k/2} \leq R(k, k)$

Got to grips with neighbours, edge densities, convexity

figured out that most claims only **hold in the limit**

Formalised a second probabilistic proof

# First half of the proof

*Proof of Lemma 4.1.* Let $W \subset X$ be the set of vertices with blue degree at least $\mu|X|$, set $m = \ell^{2/3}$, and note that $|W| \geqslant R(k,m)$, so $W$ contains either a red $K_k$ or a blue $K_m$. In the former case we are done, so assume that $U \subset W$ is the vertex set of a blue $K_m$. Let $\sigma$ be the density of blue edges between $U$ and $X \setminus U$, and observe that

$$\sigma = \frac{e_B(U, X \setminus U)}{|U| \cdot |X \setminus U|} \geqslant \frac{\mu|X| - |U|}{|X| - |U|} \geqslant \mu - \frac{1}{k}, \tag{10}$$

since $|U| = m$ and $|X| \geqslant R(k,m)$, and each vertex of $U$ has at least $\mu|X|$ blue neighbours in $X$. Since $\mu > 0$ is constant, $b = \ell^{1/4}$ and $m = \ell^{2/3}$, it follows that $b \leqslant \sigma m/2$.

Inequalities frequently hold only in the limit

Bhavik changed this to 2

```
have "μ * (card X - card U) ≤ card (Blue ∩ all_edges_betw_un {u} (X-U)) + (1-μ) * m"
  if "u ∈ U" for u
proof -
  have NBU: "Neighbours Blue u ∩ U = U - {u}"
    using ‹clique U Blue› Red_Blue_all singleton_not_edge that
    by (force simp: Neighbours_def clique_def)
  then have NBX_split: "(Neighbours Blue u ∩ X) = (Neighbours Blue u ∩ (X-U)) ∪ (U - {u})"
    using ‹U ⊂ X› by blast
  moreover have "Neighbours Blue u ∩ (X-U) ∩ (U - {u}) = {}"
    by blast
  ultimately have "card(Neighbours Blue u ∩ X) = card(Neighbours Blue u ∩ (X-U)) + (m - Suc 0)"
    by (simp add: card_Un_disjoint finite_Neighbours ‹finite U› ‹card U = m› that)
  then have "μ * (card X) ≤ real (card (Neighbours Blue u ∩ (X-U))) + real (m - Suc 0)"
    using W_def ‹U ⊆ W› bluish_def that by force
  then have "μ * (card X - card U)
          ≤ card (Neighbours Blue u ∩ (X-U)) + real (m - Suc 0) - μ *card U"
    by (smt (verit) cardU_less_X nless_le of_nat_diff right_diff_distrib')
  then have *: "μ * (card X - card U) ≤ real (card (Neighbours Blue u ∩ (X-U))) + (1-μ)*m"
    using assms by (simp add: ‹card U = m› left_diff_distrib)
  have "inj_on (λx. {u,x}) (Neighbours Blue u ∩ X)"
    by (simp add: doubleton_eq_iff inj_on_def)
  moreover have "(λx. {u,x}) ` (Neighbours Blue u ∩ (X-U)) ⊆ Blue ∩ all_edges_betw_un {u} (X-U)"
    using Blue_E by (auto simp: Neighbours_def all_edges_betw_un_def)
  ultimately have "card (Neighbours Blue u ∩ (X-U)) ≤ card (Blue ∩ all_edges_betw_un {u} (X-U))"
    by (metis NBX_split Blue_eq card_image card_mono complete fin_edges finite_Diff finite_Int inj_(
  with * show ?thesis
    by auto
qed
```

# Second half of the proof

Let $S \subset U$ be a uniformly-chosen random subset of size $b$, and let $Z = |N_B(S) \cap (X \setminus U)|$ be the number of common blue neighbours of $S$ in $X \setminus U$. By convexity, we have

$$\mathbb{E}[Z] = \binom{m}{b}^{-1} \sum_{v \in X \setminus U} \binom{|N_B(v) \cap U|}{b} \geqslant \binom{m}{b}^{-1} \binom{\sigma m}{b} \cdot |X \setminus U|.$$

probabilistic argument

Now, by Fact 4.2, and recalling (10), and that $b = \ell^{1/4}$ and $m = \ell^{2/3}$, it follows that

$$\mathbb{E}[Z] \geqslant \sigma^b \exp\left(-\frac{b^2}{\sigma m}\right) \cdot |X \setminus U| \geqslant \frac{\mu^b}{2} \cdot |X|, \tag{11}$$

and hence there exists a blue clique $S \subset U$ of size $b$ with at least this many common blue neighbours in $X \setminus U$, as required. □

Probabilistic proofs – commonplace in combinatorics – were introduced by Erdős

```
define Ω where "Ω ≡ nsets U b"  —‹Choose a random subset of size @{term b}›
have cardΩ: "card Ω = m choose b"
  by (simp add: Ω_def ‹card U = m›)
then have finΩ: "finite Ω" and "Ω ≠ {}" and "card Ω > 0"
  using ‹b ≤ m› not_less by fastforce+
define M where "M ≡ uniform_count_measure Ω"
interpret P: prob_space M
  using M_def ‹b ≤ m› cardΩ finΩ prob_space_uniform_count_measure by force
have measure_eq: "measure M C = (if C ⊆ Ω then card C / card Ω else 0)" for C
  by (simp add: M_def finΩ measure_uniform_count_measure_if)

define Int_NB where "Int_NB ≡ λS. ⋂v∈S. Neighbours Blue v ∩ (X-U)"
have sum_card_NB: "(∑A∈Ω. card (⋂(Neighbours Blue ` A) ∩ Y))
                   = (∑v∈Y. card (Neighbours Blue v ∩ U) choose b)"
  if "finite Y" "Y ⊆ X-U" for Y
  using that
proof (induction Y)
  case (insert y Y)
  have *: "Ω ∩ {A. ∀x∈A. y ∈ Neighbours Blue x} = nsets (Neighbours Blue y ∩ U) b"
    "Ω ∩ - {A. ∀x∈A. y ∈ Neighbours Blue x} = Ω - nsets (Neighbours Blue y ∩ U) b"
    "[Neighbours Blue y ∩ U]⇗b⇖ ⊆ Ω"
    using insert.prems by (auto simp: Ω_def nsets_def in_Neighbours_iff insert_commute)
  then show ?case
    using insert finΩ
    by (simp add: Int_insert_right sum_Suc sum.If_cases if_distrib [of card]
        sum.subset_diff flip: insert.IH)
qed auto
```

# Further stages of the proof

- Ensuring the red density between $X$, $Y$ is high enough
- Ensuring that $X$ and $Y$ aren't "used up" too quickly
- Exponential improvements away from the diagonal
- The main result, on the diagonal ($k = \ell$)

# Computer Algebra Aspects

# Formalising claims about limits

- Accumulate equalities required by each theorem, e.g.
  $\ell \geq (6/\mu)^{12/5}$ or $\dfrac{2}{\ell} \leq (\mu - 2/\ell)((5/4)^{1/\lceil \ell^{1/4} \rceil} - 1)$
- Check them out by plotting in Maple
- … then prove that they actually hold in the limit
- For the base cases, use the proof method **real_asymp**

# Limit claims either **local** to the theorem

```
let ?Big = "λl. m_of l ≥ 12  ∧  l ≥ (6/μ) powr (12/5)  ∧  l ≥ 15
              ∧ 1 ≤ 5/4 * exp (- ((b_of l)^2) / ((μ - 2/l) * m_of l))  ∧  μ > 2/l
              ∧ 2/l ≤ (μ - 2/l) * ((5/4) powr (1/b_of l) - 1)"
have big_enough_l: "∀∞l. ?Big l"
  unfolding m_of_def b_of_def using assms by (intro eventually_conj; real_asymp)
```

# Or **separate from** the theorem

```
definition "Big_X_7_6 ≡
   λμ l. Lemma_bblue_dboost_step_limit μ l ∧ Lemma_bblue_step_limit μ l ∧ Big_X_7_12 μ l
         ∧ (∀k. k≥l ⟶ Big_X_7_8 k ∧ 1 - 2 * eps k powr (1/4) > 0)"

lemma Big_X_7_6:
  assumes "0<μ" "μ<1"
  shows "∀∞l. Big_X_7_6 μ l"
  unfolding Big_X_7_6_def eventually_conj_iff all_imp_conj_distrib eps_def
  apply (simp add: bblue_dboost_step_limit Big_X_7_8 Big_X_7_12
        bblue_step_limit eventually_all_ge_at_top assms)
  by (intro  eventually_all_ge_at_top; real_asymp)
```

# Landau symbols in the proofs

Many formulas such as $|Y| \geqslant 2^{o(k)} p_0^{s+t} \cdot |Y_0|$

Quite a few different Landau symbol occurrences, but mostly $o(k)$

I preferred making these functions explicit

# Proving $\prod_{i \in \mathcal{D}} \frac{|X_i|}{|X_{i-1}|} = 2^{o(k)}$

```
definition "ok_fun_X_7_6 ≡
  λl k. ((1 + (real k + real l)) * ln (1 - 2 * eps k powr (1/4))
      - (k powr (3/4) + 7 * eps k powr (1/4) * k + 1) * (2 * ln k)) / ln 2"

lemma ok_fun_X_7_6: "ok_fun_X_7_6 l ∈ o(real)" for l
  unfolding eps_def ok_fun_X_7_6_def by real_asymp

lemma X_7_6:
  fixes l k
  assumes μ: "0<μ" "μ<1" and "Colours l k"
  assumes big: "Big_X_7_6 μ l"
  defines "X ≡ Xseq μ l k" and "𝒟 ≡ Step_class μ l k {dreg_step}"
  shows "(∏i∈𝒟. card(X(Suc i)) / card (X i)) ≥ 2 powr ok_fun_X_7_6 l k"
```

# A proof using exact calculations

Since $\delta = \min\{1/200, \gamma/20\}$, to deduce that $t \geqslant 2k/3$ it now suffices to check that[11]

$$\left(1 - \frac{1}{200\gamma}\right)\left(1 + \frac{1}{e(1-\gamma)}\right)^{-1} \geqslant \left(1 - \frac{1}{40}\right)\left(1 + \frac{5}{4e}\right)^{-1} > 0.667 > \frac{2}{3} \qquad (47)$$

for all $1/10 \leqslant \gamma \leqslant 1/5$, and that

```
define c where "c ≡ λx::real. 1 + 1 / (exp 1 * (1-x))"
define f47 where "f47 ≡ λx. (1 - 1/(200*x)) * inverse (c x)"
have "concave_on {1/10..1/5} f47" [46 lines]
moreover have "f47(1/10) > 0.667"
  unfolding f47_def c_def by (approximation 15)
moreover have "f47(1/5) > 0.667"
  unfolding f47_def c_def by (approximation 15)
ultimately have 47: "f47 x > 0.667" if "x ∈ {1/10..1/5}" for x
  using concave_on_ge_min that by fastforce
```

# Proving Lemma A.4

```
lemma A4:
  assumes "y ∈ {0.341..3/4}"
  shows "f2 (x_of y) y ≤ 2 - 1/2^11"
  unfolding f2_def f1_def x_of_def H_def
  using assms by (approximation 18 splitting: y = 13)
```

```
goal (1 subgoal):
 1. 3 * y / 5 + 5454 / 10 ^ 4 + y +
    (2 - (3 * y / 5 + 5454 / 10 ^ 4)) *
    (- (1 / (2 - (3 * y / 5 + 5454 / 10 ^ 4))) *
     log 2 (1 / (2 - (3 * y / 5 + 5454 / 10 ^ 4))) -
     (1 - 1 / (2 - (3 * y / 5 + 5454 / 10 ^ 4))) *
     log 2 (1 - 1 / (2 - (3 * y / 5 + 5454 / 10 ^ 4)))) -
    1 / (40 * ln 2) *
    ((1 - (3 * y / 5 + 5454 / 10 ^ 4)) / (2 - (3 * y / 5 + 5454 / 10 ^ 4)))
    ≤ 2 - 1 / 2 ^ 11
```

# Conclusions

- Some proofs really do need computer algebra or exact arithmetic
- The **approximation** and **real_asymp** proof methods are fast and powerful
- Differentiation by pure inference is easy, if a hack
- This proof is incredibly difficult

```
text ‹Main theorem 1.1: the exponent is approximately 3.9987›
theorem Main_1_1:
  obtains ε::real where "ε>0" "∀∞k. RN k k ≤ (4-ε)^k"
proof
  let ?ε = "0.00134::real"
  have "∀∞k. k>0 ∧ log 2 (RN k k) / k ≤ 2 - delta'"
    unfolding eventually_conj_iff using Aux_1_1 eventually_gt_at_top by blast
  then have "∀∞k. RN k k ≤ (2 powr (2-delta')) ^ k"
  proof (eventually_elim)
    case (elim k)
    then have "log 2 (RN k k) ≤ (2-delta') * k"
      by (meson of_nat_0_less_iff pos_divide_le_eq)
    then have "RN k k ≤ 2 powr ((2-delta') * k)"
      by (smt (verit, best) Transcendental.log_le_iff powr_ge_zero)
    then show "RN k k ≤ (2 powr (2-delta')) ^ k"
      by (simp add: mult.commute powr_power)
  qed
  moreover have "2 powr (2-delta') ≤ 4 - ?ε"
    unfolding delta'_def by (approximation 25)
  ultimately show "∀∞k. real (RN k k) ≤ (4-?ε) ^ k"
    by (smt (verit) power_mono powr_ge_zero eventually_mono)
qed auto
```

The whole development is 11 K lines and runs in under 9 minutes. Formalisation took 251 days.

Many thanks to Mantas Baksys, Manuel Eberl, Simon Griffiths, Fabian Immler, Bhavik Mehta and Andrew Thomason

(If you want to understand the actual proof, please see Bhavik's *Lean Together* talk on the **leanprover community** YouTube channel)