Interactions of Proof Assistants and Mathematics,
International Summer School, Regensburg

PART 2: FORMALISATION OF ADDITIVE COMBINATORICS IN ISABELLE/HOL

22/9/2023

UNIVERSITY OF
CAMBRIDGE

Isabelle

Angeliki Koutsoukou-Argyraki

Royal Holloway, University of London, UK
and
University of Cambridge, UK

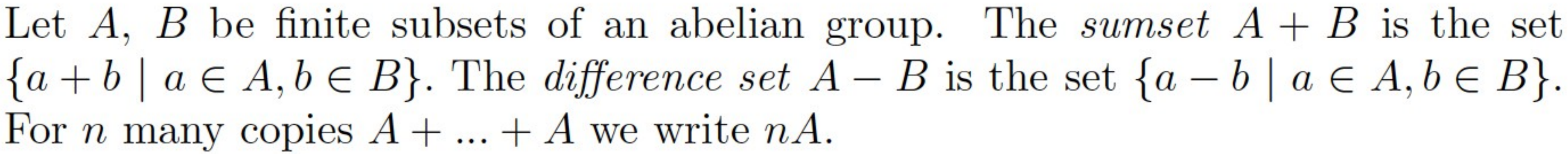
erc

European Research Council
Established by the European Commission

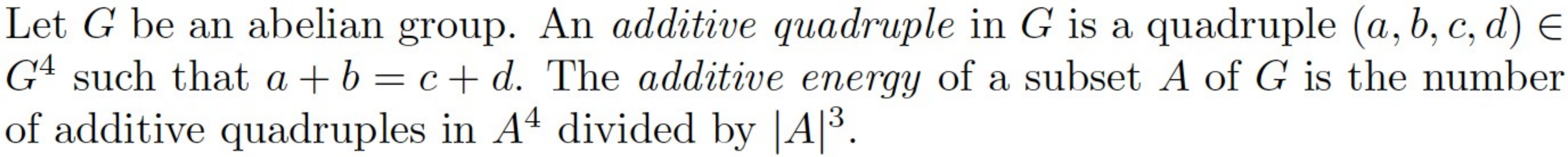
In mid-2022 I initiated a line of work to formalise material in additive combinatorics, on the structure of sumsets of finite subsets of abelian groups.

(See my invited talk in the proceedings of the 14th Conference on Interactive Theorem Proving (ITP 2023)
DOI: 10.4230/LIPIcs.ITP.2023.1)

Basic definitions:



Let A, B be finite subsets of an abelian group. The *sumset* $A+B$ is the set $\{a+b \mid a \in A, b \in B\}$. The *difference set* $A-B$ is the set $\{a-b \mid a \in A, b \in B\}$. For n many copies $A+...+A$ we write nA.



Let G be an abelian group. An *additive quadruple* in G is a quadruple $(a,b,c,d) \in G^4$ such that $a+b=c+d$. The *additive energy* of a subset A of G is the number of additive quadruples in A^4 divided by $|A|^3$.

My motivation:

* Case studies to explore limits of formalisation: material combining different areas

* Advanced material, relatively recent research results but very simple prerequisites (some of which had to be built by us)

* Using source material from the Cambridge Mathematical Tripos

* Working with mathematics students too

* Contribution to the Isabelle libraries of formalised mathematics (AFP)

My motivation:

* Personal interest in this area of mathematics:

material provides tools for the study of the active research area of arithmetic progressions in integers.

Note: we have already formalised Roth’s important Theorem on Arithmetic Progressions.

[Suggested by Lawrence Paulson]

Used the experimental abstract algebra library by Clemens Ballarin (“A Case Study in Basic Algebra”, 2019) available on the AFP, instead of the standard algebra library. Makes heavy use of locales and follows Jacobson’s “Basic Algebra” book.

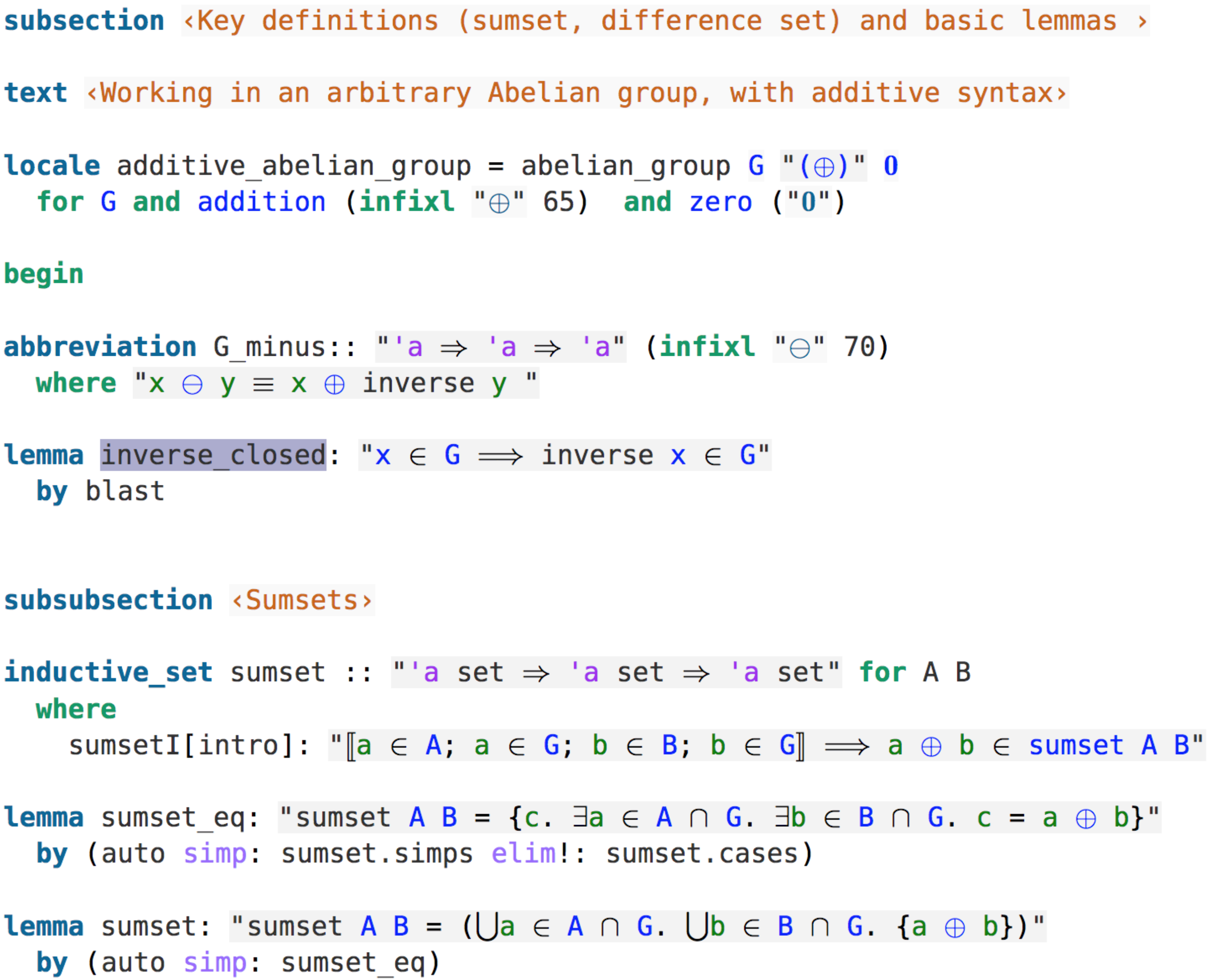
AFP entries produced during the 2022 project:

- The Plünnecke-Ruzsa Inequality (A. K.-A. & Lawrence C. Paulson, 2022).
- Khovanskii’s Theorem (A. K.-A. & Lawrence C. Paulson, 2022).
- The Balog-Szemerédi-Gowers Theorem (A. K.-A., Mantas Bakšys and Chelsea Edmonds, 2022).

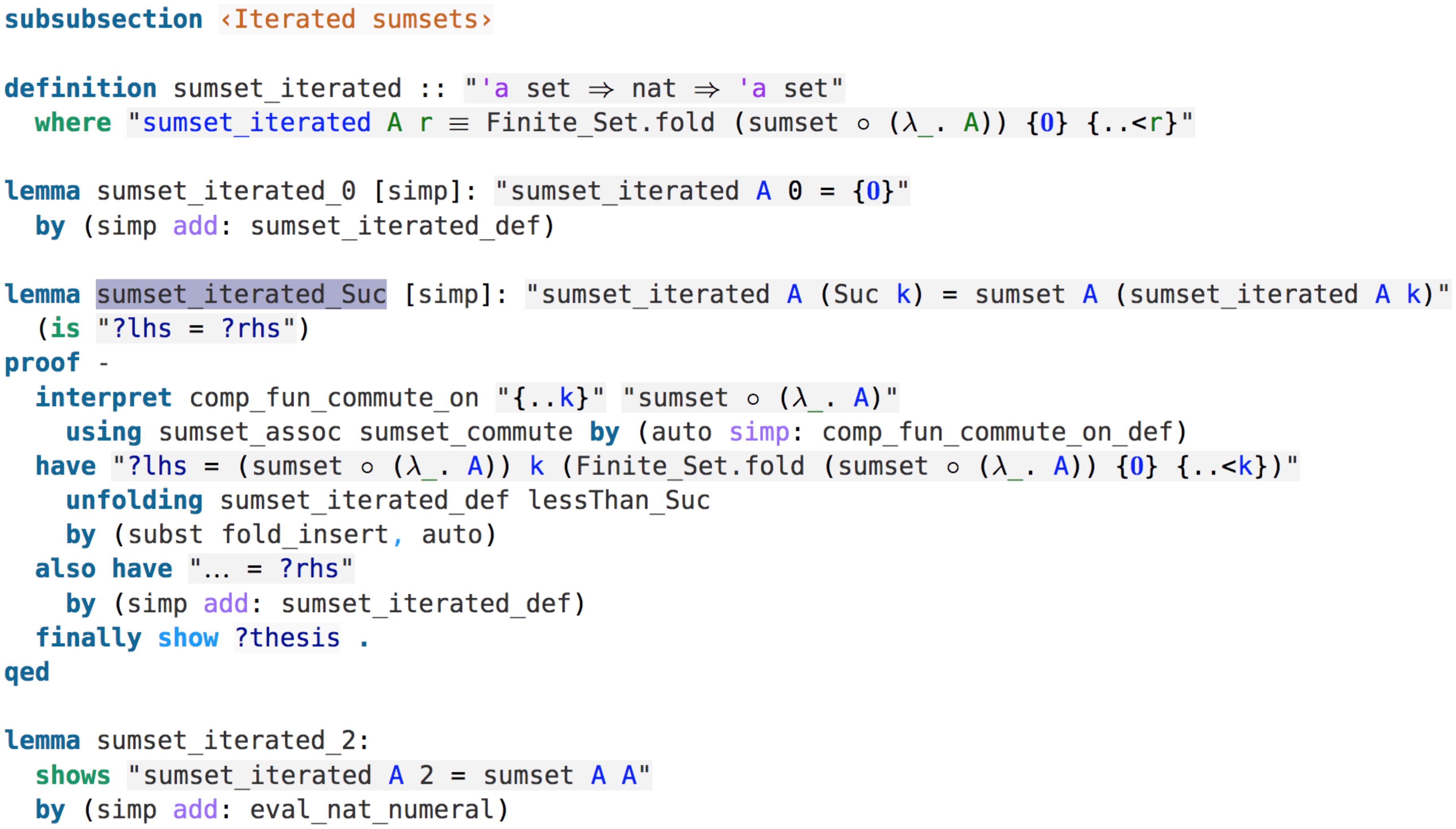
(Source: Introduction to Additive Combinatorics, Course notes for Part III of the Cambridge Mathematical Tripos by W.T. Gowers (2022)).

- Kneser’s Theorem and the Cauchy-Davenport Theorem (Mantas Bakšys & A. K.-A., 2022)

(Source: DeVos, M. (2014). A Short Proof of Kneser’s Addition Theorem for Abelian Groups. In: Nathanson, M. (eds) Combinatorial and Additive Number Theory. Springer Proceedings in Mathematics & Statistics, vol 101.)

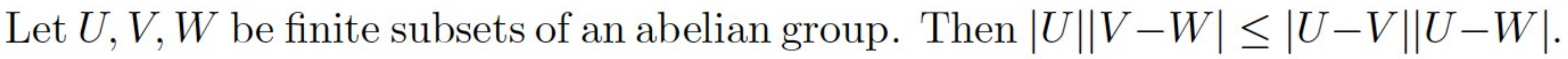


```
subsection ‹Key definitions (sumset, difference set) and basic lemmas ›

text ‹Working in an arbitrary Abelian group, with additive syntax›

locale additive_abelian_group = abelian_group G "(⊕)" 0
  for G and addition (infixl "⊕" 65)  and zero ("0")

begin

abbreviation G_minus:: "'a ⇒ 'a ⇒ 'a" (infixl "⊖" 70)
  where "x ⊖ y ≡ x ⊕ inverse y "

lemma inverse_closed: "x ∈ G ⟹ inverse x ∈ G"
  by blast


subsubsection ‹Sumsets›

inductive_set sumset :: "'a set ⇒ 'a set ⇒ 'a set" for A B
  where
    sumsetI[intro]: "⟦a ∈ A; a ∈ G; b ∈ B; b ∈ G⟧ ⟹ a ⊕ b ∈ sumset A B"

lemma sumset_eq: "sumset A B = {c. ∃a ∈ A ∩ G. ∃b ∈ B ∩ G. c = a ⊕ b}"
  by (auto simp: sumset.simps elim!: sumset.cases)

lemma sumset: "sumset A B = (⋃a ∈ A ∩ G. ⋃b ∈ B ∩ G. {a ⊕ b})"
  by (auto simp: sumset_eq)
```

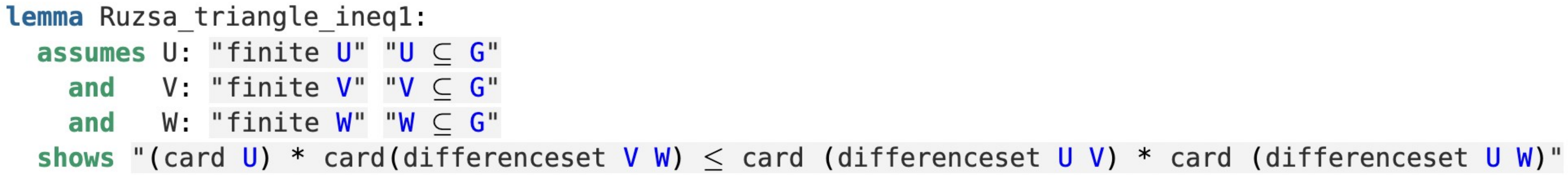


```
subsubsection ‹Iterated sumsets›

definition sumset_iterated :: "'a set ⇒ nat ⇒ 'a set"
  where "sumset_iterated A r ≡ Finite_Set.fold (sumset ∘ (λ_. A)) {0} {..<r}"

lemma sumset_iterated_0 [simp]: "sumset_iterated A 0 = {0}"
  by (simp add: sumset_iterated_def)

lemma sumset_iterated_Suc [simp]: "sumset_iterated A (Suc k) = sumset A (sumset_iterated A k)"
  (is "?lhs = ?rhs")
proof -
  interpret comp_fun_commute_on "{..k}" "sumset ∘ (λ_. A)"
    using sumset_assoc sumset_commute by (auto simp: comp_fun_commute_on_def)
  have "?lhs = (sumset ∘ (λ_. A)) k (Finite_Set.fold (sumset ∘ (λ_. A)) {0} {..<k})"
    unfolding sumset_iterated_def lessThan_Suc
    by (subst fold_insert, auto)
  also have "… = ?rhs"
    by (simp add: sumset_iterated_def)
  finally show ?thesis .
qed

lemma sumset_iterated_2:
  shows "sumset_iterated A 2 = sumset A A"
  by (simp add: eval_nat_numeral)
```

Ruzsa

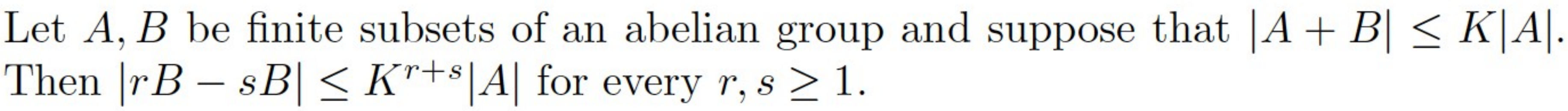


Let U, V, W be finite subsets of an abelian group. Then $|U||V - W| \leq |U - V||U - W|$.

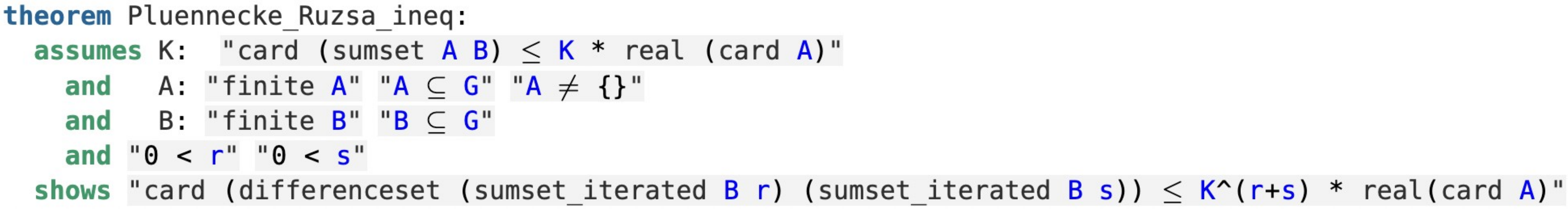


```
lemma Ruzsa_triangle_ineq1:
  assumes U: "finite U" "U ⊆ G"
    and   V: "finite V" "V ⊆ G"
    and   W: "finite W" "W ⊆ G"
  shows "(card U) * card(differenceset V W) ≤ card (differenceset U V) * card (differenceset U W)"
```

Plünnecke-Ruzsa

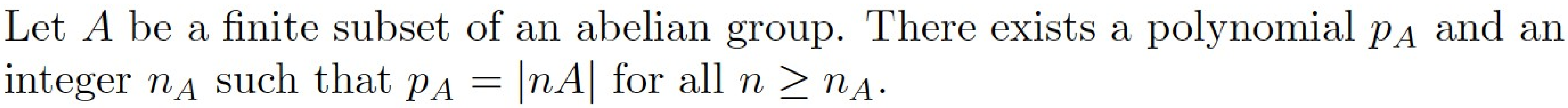


Let A, B be finite subsets of an abelian group and suppose that $|A + B| \leq K|A|$. Then $|rB - sB| \leq K^{r+s}|A|$ for every $r, s \geq 1$.

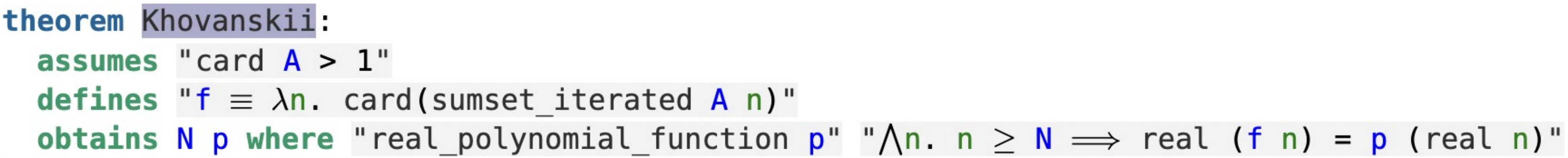


```
theorem Pluennecke_Ruzsa_ineq:
  assumes K:  "card (sumset A B) ≤ K * real (card A)"
    and   A: "finite A" "A ⊆ G" "A ≠ {}"
    and   B: "finite B" "B ⊆ G"
    and "0 < r" "0 < s"
  shows "card (differenceset (sumset_iterated B r) (sumset_iterated B s)) ≤ K^(r+s) * real(card A)"
```

Khovanskii

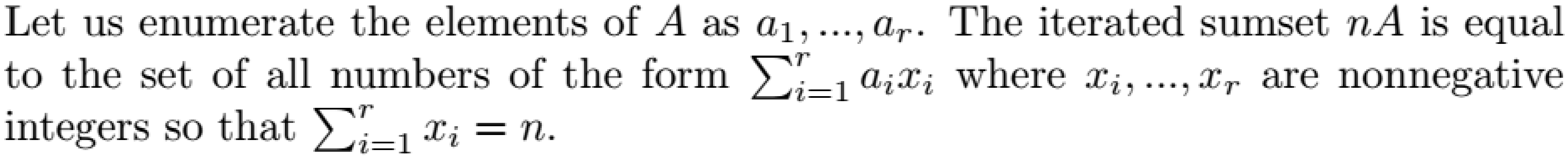


Let A be a finite subset of an abelian group. There exists a polynomial p_A and an integer n_A such that $p_A = |nA|$ for all $n \geq n_A$.



```
theorem Khovanskii:
  assumes "card A > 1"
  defines "f ≡ λn. card(sumset_iterated A n)"
  obtains N p where "real_polynomial_function p" "⋀n. n ≥ N ⟹ real (f n) = p (real n)"
```

Khovanskii

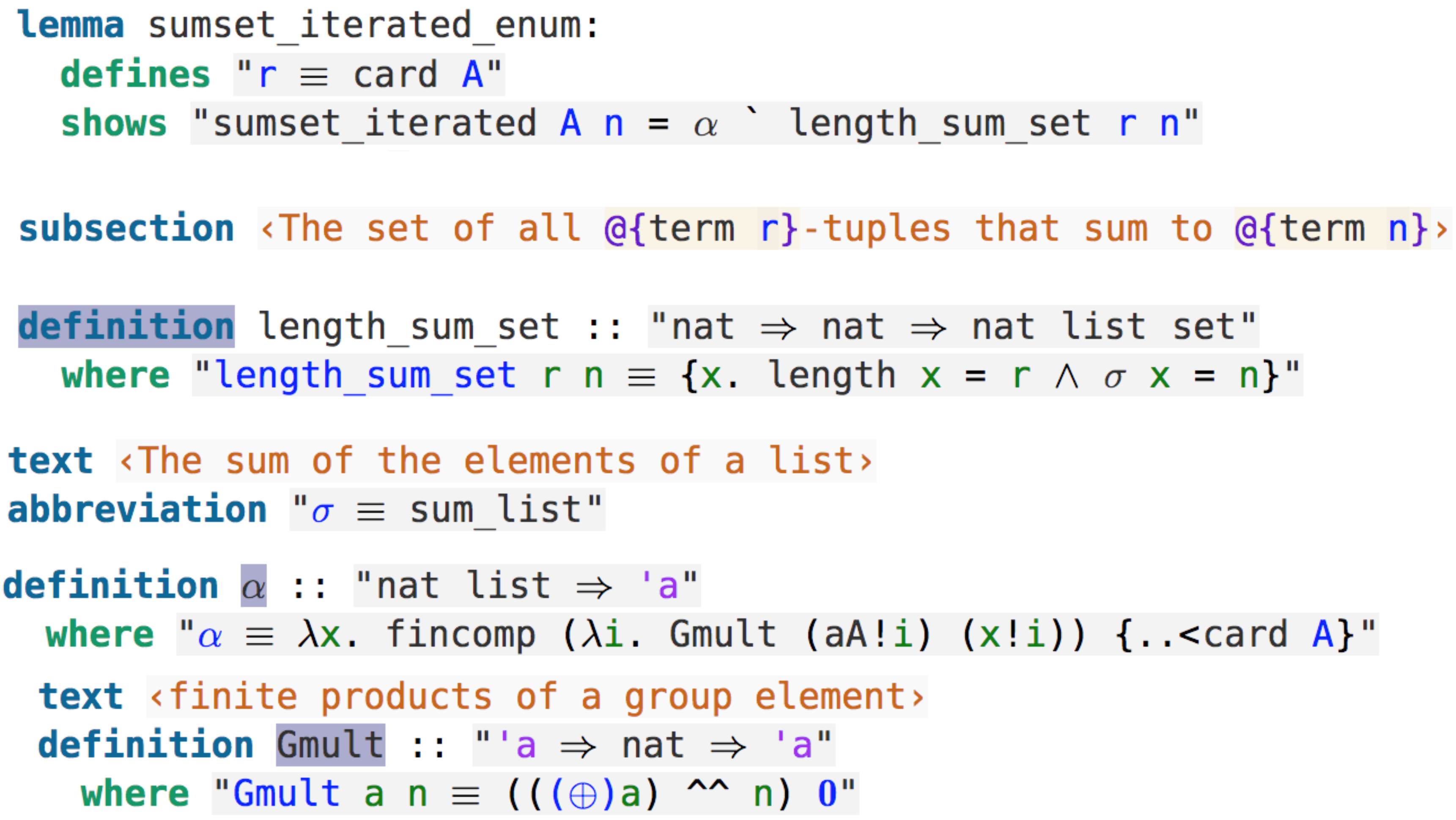


Let us enumerate the elements of A as $a_1, ..., a_r$. The iterated sumset nA is equal to the set of all numbers of the form $\sum_{i=1}^{r} a_i x_i$ where $x_i, ..., x_r$ are nonnegative integers so that $\sum_{i=1}^{r} x_i = n$.

We treat tuples as lists.

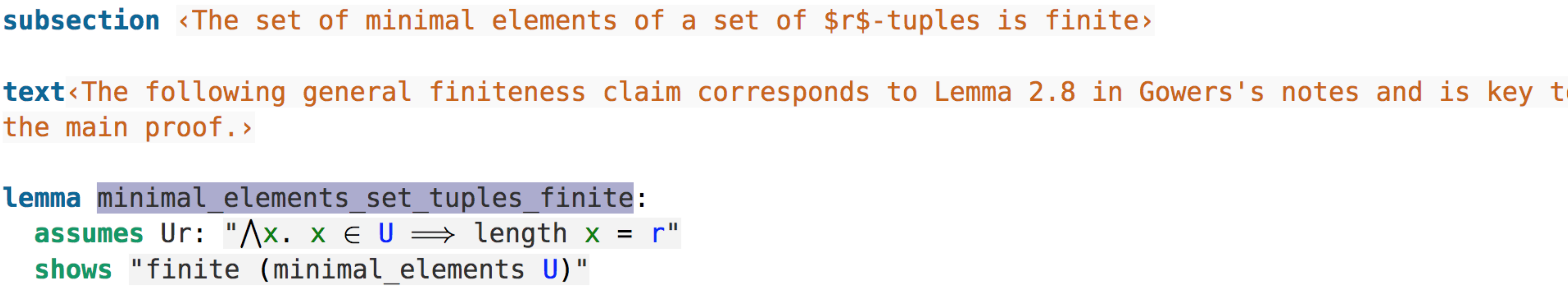
Required a development of a Product Operator for Commutative Monoids theory (finite products in group theory) which was largely based on HOL/Algebra/FiniteProduct.thy.

Khovanskii

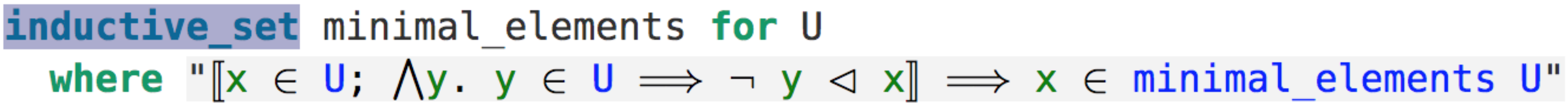


```
lemma sumset_iterated_enum:
  defines "r ≡ card A"
  shows "sumset_iterated A n = α ` length_sum_set r n"

subsection ‹The set of all @{term r}-tuples that sum to @{term n}›

definition length_sum_set :: "nat ⇒ nat ⇒ nat list set"
  where "length_sum_set r n ≡ {x. length x = r ∧ σ x = n}"

text ‹The sum of the elements of a list›
abbreviation "σ ≡ sum_list"

definition α :: "nat list ⇒ 'a"
  where "α ≡ λx. fincomp (λi. Gmult (aA!i) (x!i)) {..<card A}"

 text ‹finite products of a group element›
 definition Gmult :: "'a ⇒ nat ⇒ 'a"
   where "Gmult a n ≡ (((⊕)a) ^^ n) 𝟬"
```

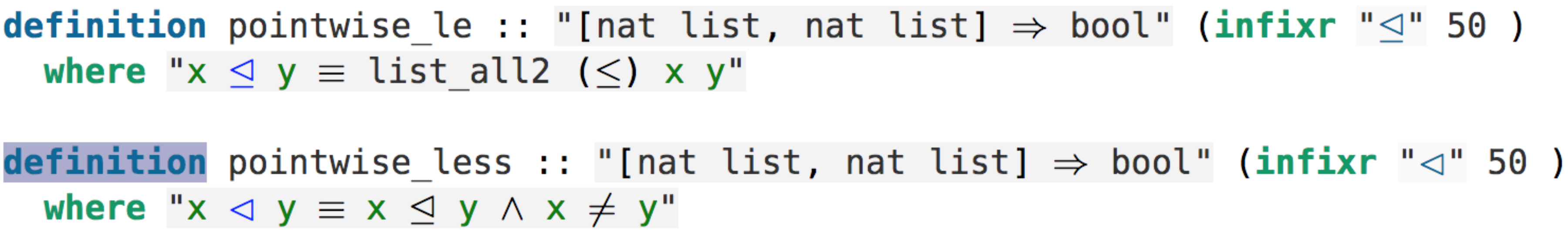
Khovanskii



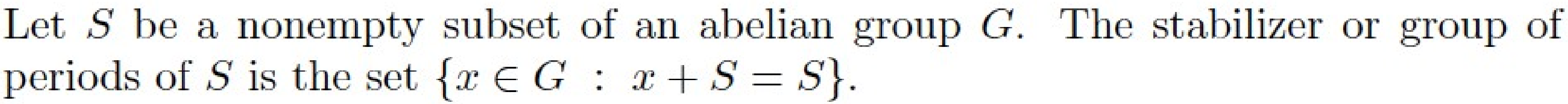
```
subsection ‹The set of minimal elements of a set of $r$-tuples is finite›

text‹The following general finiteness claim corresponds to Lemma 2.8 in Gowers's notes and is key t
the main proof.›

lemma minimal_elements_set_tuples_finite:
  assumes Ur: "⋀x. x ∈ U ⟹ length x = r"
  shows "finite (minimal_elements U)"
```



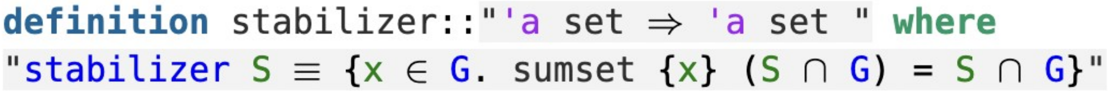
```
inductive_set minimal_elements for U
  where "⟦x ∈ U; ⋀y. y ∈ U ⟹ ¬ y ◁ x⟧ ⟹ x ∈ minimal_elements U"
```



```
definition pointwise_le :: "[nat list, nat list] ⇒ bool" (infixr "⊴" 50 )
  where "x ⊴ y ≡ list_all2 (≤) x y"

definition pointwise_less :: "[nat list, nat list] ⇒ bool" (infixr "◁" 50 )
  where "x ◁ y ≡ x ⊴ y ∧ x ≠ y"
```

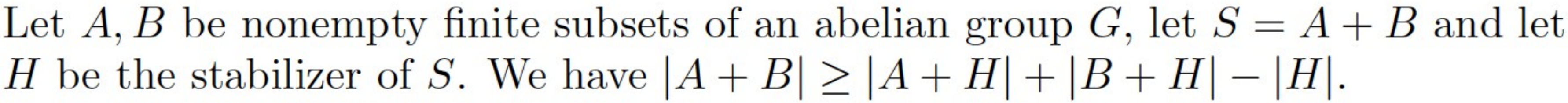


Let S be a nonempty subset of an abelian group G. The stabilizer or group of periods of S is the set $\{x \in G \ : \ x + S = S\}$.

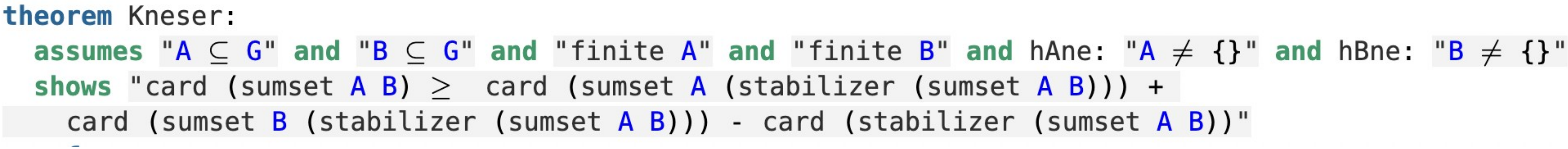


```
definition stabilizer::"'a set ⇒ 'a set " where
"stabilizer S ≡ {x ∈ G. sumset {x} (S ∩ G) = S ∩ G}"
```

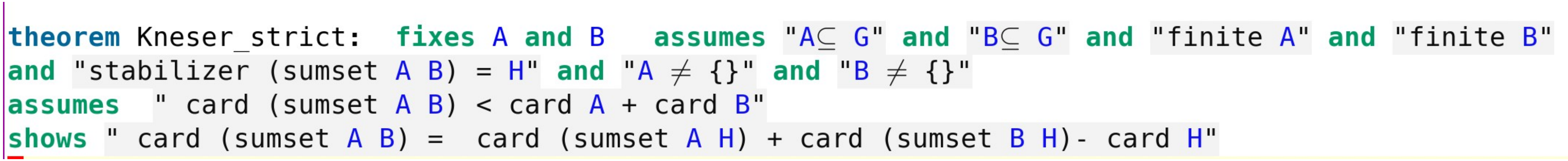
Kneser



Let A, B be nonempty finite subsets of an abelian group G, let $S = A + B$ and let H be the stabilizer of S. We have $|A + B| \geq |A + H| + |B + H| - |H|$.



```
theorem Kneser:
  assumes "A ⊆ G" and "B ⊆ G" and "finite A" and "finite B" and hAne: "A ≠ {}" and hBne: "B ≠ {}"
  shows "card (sumset A B) ≥  card (sumset A (stabilizer (sumset A B))) +
    card (sumset B (stabilizer (sumset A B))) - card (stabilizer (sumset A B))"
```



```
theorem Kneser_strict:  fixes A and B   assumes "A⊆ G" and "B⊆ G" and "finite A" and "finite B"
and "stabilizer (sumset A B) = H" and "A ≠ {}" and "B ≠ {}"
assumes  " card (sumset A B) < card A + card B"
shows " card (sumset A B) =  card (sumset A H) + card (sumset B H)- card H"
```

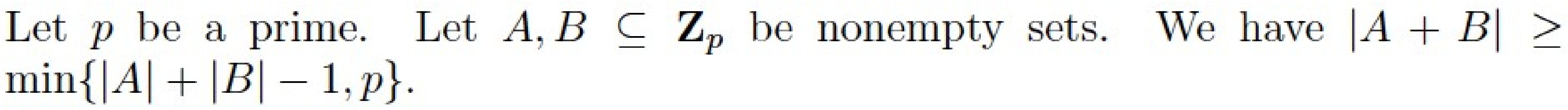
Kneser

Induction on the cardinality of a finite set in an abelian group with the induction hypothesis applied to a finite set (of smaller cardinality) now in a quotient group of the original abelian group.

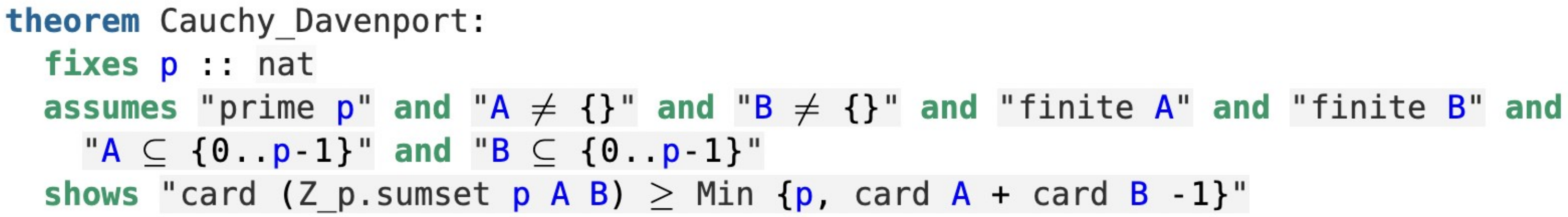
Issue in formalisation: the quotient group and the original abelian group have carrier sets of different types. Our induction argument needs to generalise the types of the carrier sets of the abelian groups we are considering.

Workaround solution: force the quotient group to live over the same type as the original abelian group by using the coset representatives as the group elements, and push all of the relevant information through this isomorphism. (thanks to Manuel Eberl via Zulip)

Cauchy-Davenport



Let p be a prime. Let $A, B \subseteq \mathbf{Z}_p$ be nonempty sets. We have $|A + B| \geq \min\{|A| + |B| - 1, p\}$.



```
theorem Cauchy_Davenport:
  fixes p :: nat
  assumes "prime p" and "A ≠ {}" and "B ≠ {}" and "finite A" and "finite B" and
    "A ⊆ {0..p-1}" and "B ⊆ {0..p-1}"
  shows "card (Z_p.sumset p A B) ≥ Min {p, card A + card B -1}"
```

Note:

Mantas Bakšys and Yaël Dillies later went on to formalise Kneser’s Theorem and the Cauchy-Davenport Theorem in Lean too.
(Joint paper in progress to compare the formalisations).

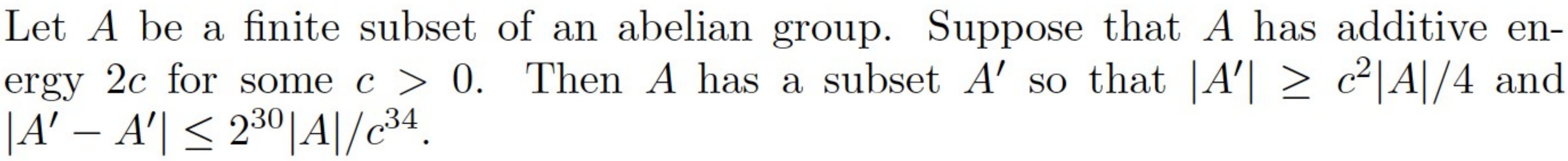
* A formalisation of the Balog-Szemerédi-Gowers Theorem in Isabelle/HOL (A. K.-A., Mantas Bakšys & Chelsea Edmonds, in CPP '23: 12th ACM SIGPLAN, International Conference on Certified Programs and Proofs).

A profound result in additive combinatorics which played a central role in Gowers's proof deriving the first effective bounds for Szemerédi's Theorem on arithmetic progressions.

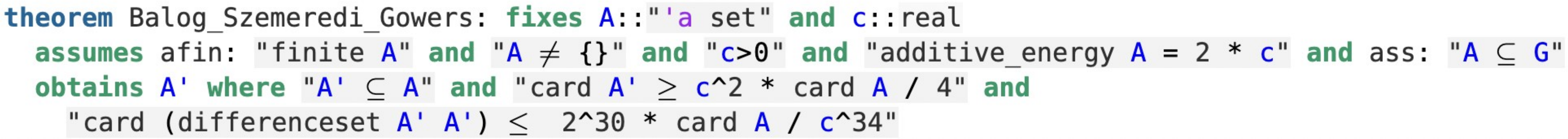
Balog & Szemerédi (1994): Every finite subset of an abelian group of given additive energy must contain a large subset whose sumset is small.

Gowers (2001): New proof with better bounds on the cardinalities.

Balog-Szemerédi-Gowers:



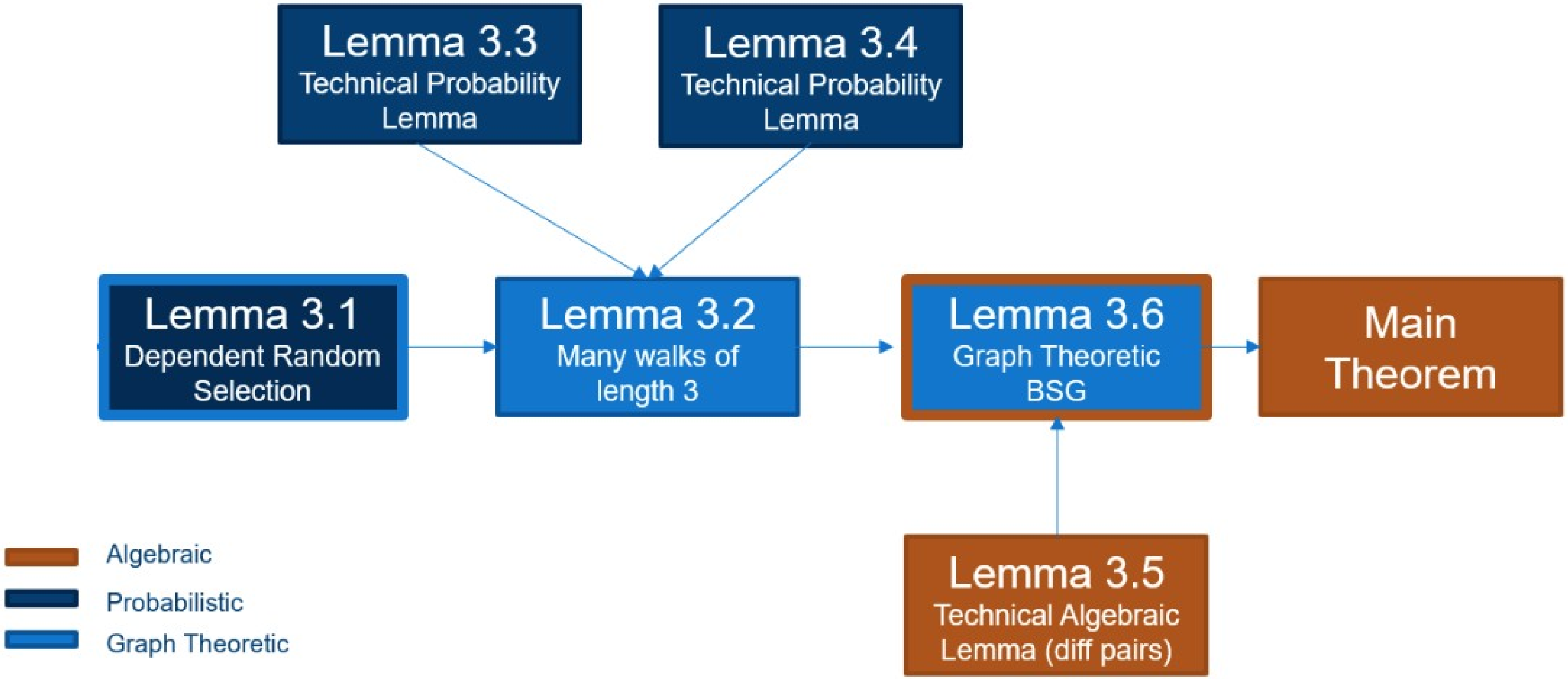
Let A be a finite subset of an abelian group. Suppose that A has additive energy $2c$ for some $c > 0$. Then A has a subset A' so that $|A'| \geq c^2|A|/4$ and $|A' - A'| \leq 2^{30}|A|/c^{34}$.



```
theorem Balog_Szemeredi_Gowers: fixes A::"'a set" and c::real
  assumes afin: "finite A" and "A ≠ {}" and "c>0" and "additive_energy A = 2 * c" and ass: "A ⊆ G"
  obtains A' where "A' ⊆ A" and "card A' ≥ c^2 * card A / 4" and
    "card (differenceset A' A') ≤  2^30 * card A / c^34"
```

(Analogous version for sumsets).

The proof involves a fascinating interplay between graph theory, probability theory, additive combinatorics: expressed via an implementation of locales, Isabelle’s module system.
Made use of a new, general undirected graph theory library by Edmonds.



Lemma 3.3
Technical Probability Lemma

Lemma 3.4
Technical Probability Lemma

Lemma 3.1
Dependent Random Selection

Lemma 3.2
Many walks of length 3

Lemma 3.6
Graph Theoretic BSG

Main Theorem

Lemma 3.5
Technical Algebraic Lemma (diff pairs)

🟫 Algebraic

⬛ Probabilistic

🟦 Graph Theoretic

Preliminary Graph theoretic definitions

Definition (Density)

Given a bipartite graph G with finite vertex sets X, Y and edge set $E \subseteq X \times Y$ define the *density* of G as $\delta(G) = |E|/|X||Y|$

Definition (Neighbourhood)

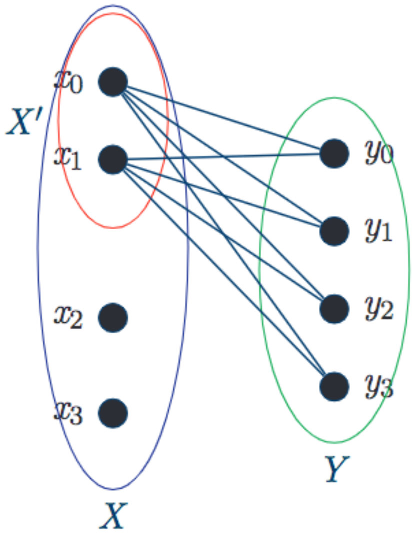
Given a graph $G = (V, E)$ we define the *neighbourhood* of $x \in V$ to be $N(x) = \{y \in V | xy \in E\}$

Definition (Codegree)

Given a graph $G = (V, E)$ we define the codegree of $x, y \in V$ as the number of paths of length two for x to y in G or formally *codegree*$(x, y) = |N(x) \cap N(y)|$

The Dependent Random Selection Method

Motivation : We want to find a large subset with better properties, based on the structure of the original set



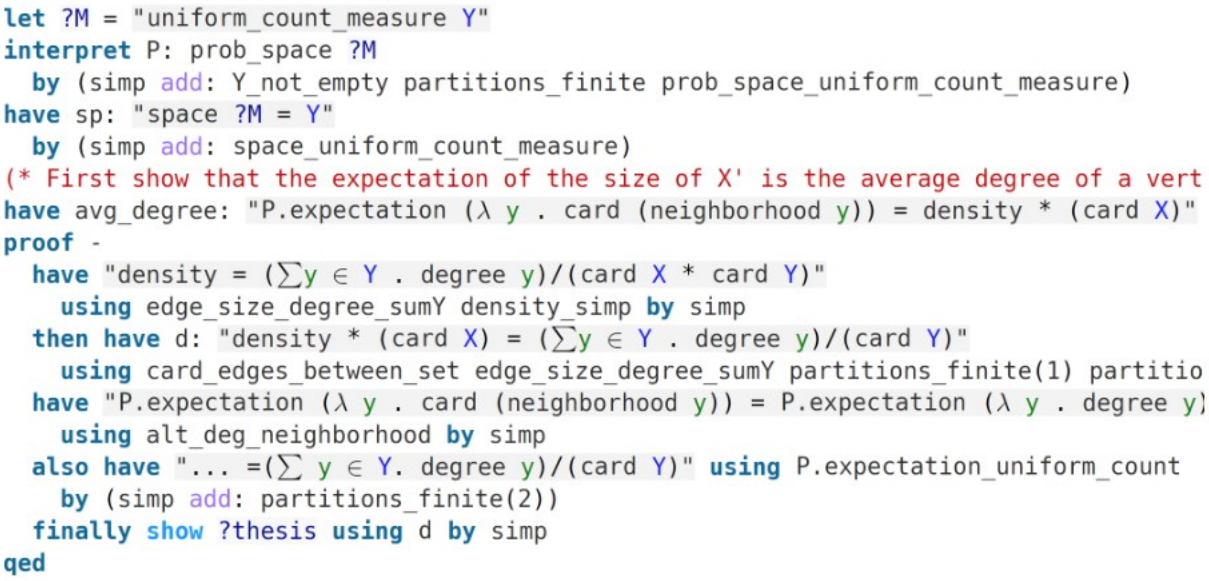
Lemma (3.1)

Given a bipartite graph $G = (X \cup Y, E)$ of density $\delta > 0$, for every $c > 0$ we can find $X' \subseteq X$ such that $|X'| \geq \delta|X|/\sqrt{2}$ and the proportion of pairs $(x, x') \in (X')^2$ with codegree$(x, x') < c|Y|$ is at most $2c/\delta^2$

The Dependent Random Selection Method

Proof Sketch.

Instead of defining X' randomly, define X' by picking $y \in Y$ at random, and let $X' = N(y)$. Now determine properties of X', e.g. the expected size of X' is average degree of $y \in Y$. □



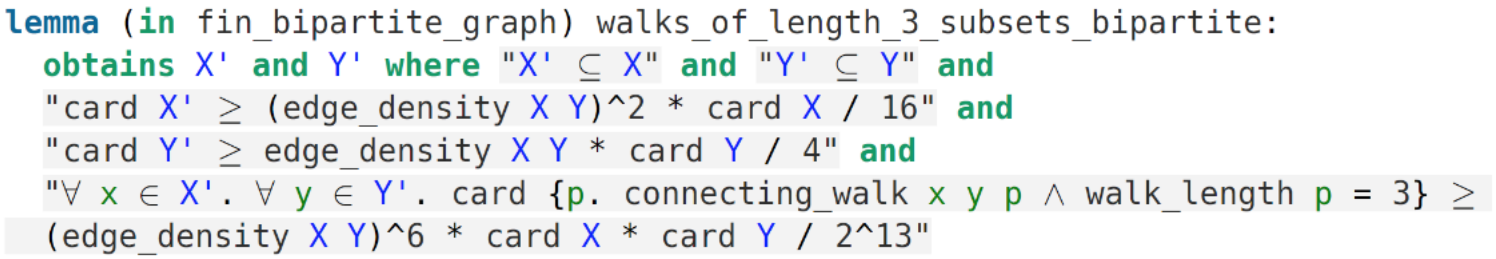
```
let ?M = "uniform_count_measure Y"
interpret P: prob_space ?M
  by (simp add: Y_not_empty partitions_finite prob_space_uniform_count_measure)
have sp: "space ?M = Y"
  by (simp add: space_uniform_count_measure)
(* First show that the expectation of the size of X' is the average degree of a vert
have avg_degree: "P.expectation (λ y . card (neighborhood y)) = density * (card X)"
proof -
  have "density = (∑y ∈ Y . degree y)/(card X * card Y)"
    using edge_size_degree_sumY density_simp by simp
  then have d: "density * (card X) = (∑y ∈ Y . degree y)/(card Y)"
    using card_edges_between_set edge_size_degree_sumY partitions_finite(1) partitio
  have "P.expectation (λ y . card (neighborhood y)) = P.expectation (λ y . degree y)
    using alt_deg_neighborhood by simp
  also have "... =(∑ y ∈ Y. degree y)/(card Y)" using P.expectation_uniform_count
    by (simp add: partitions_finite(2))
  finally show ?thesis using d by simp
qed
```

Lemma 3.2

There are many paths of length 3 between vertices in subsets

Lemma (3.2)

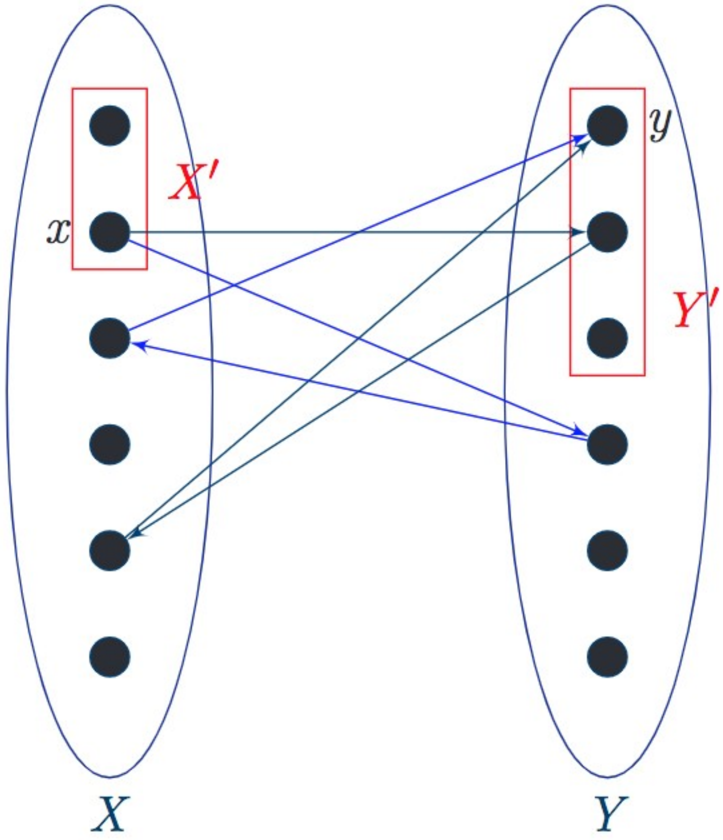
Let G be a bipartite graph with finite vertex sets X and Y and density δ. Then there are subsets $X' \subseteq X$ and $Y' \subseteq Y$ with $|X'| \geq \delta^2|X|/16$ and $|Y'| \geq \delta|Y|/4$ such that for every $x \in X'$ and $y \in Y'$ the number of paths of length 3 between x and y in G is at least $\delta^6|X||Y|/2^{13}$.



```
lemma (in fin_bipartite_graph) walks_of_length_3_subsets_bipartite:
  obtains X' and Y' where "X' ⊆ X" and "Y' ⊆ Y" and
  "card X' ≥ (edge_density X Y)^2 * card X / 16" and
  "card Y' ≥ edge_density X Y * card Y / 4" and
  "∀ x ∈ X'. ∀ y ∈ Y'. card {p. connecting_walk x y p ∧ walk_length p = 3} ≥
  (edge_density X Y)^6 * card X * card Y / 2^13"
```

Lemma 3.2

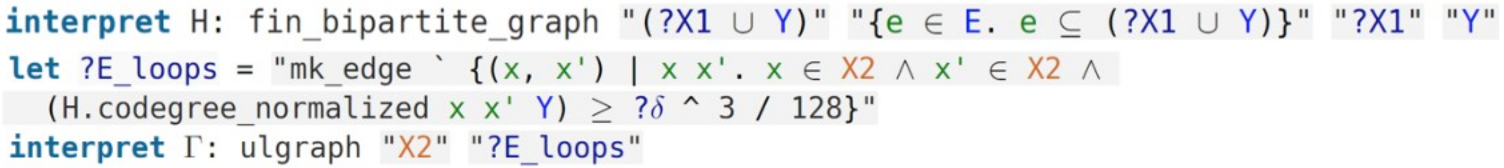
There are many paths of length 3 between vertices in subsets



"Transporting" Information across proofs

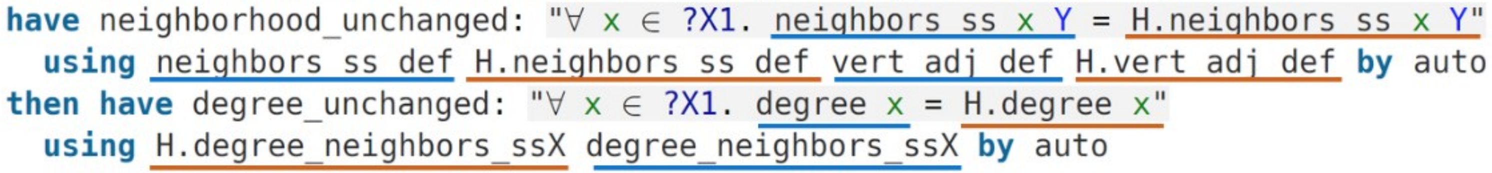
Lemma 3.2 involves many different probability spaces ($X2 \subset X$)

... And several graph constructs



```
interpret H: fin_bipartite_graph "(?X1 ∪ Y)" "{e ∈ E. e ⊆ (?X1 ∪ Y)}" "?X1" "Y"
let ?E_loops = "mk_edge ` {(x, x') | x x'. x ∈ X2 ∧ x' ∈ X2 ∧
  (H.codegree_normalized x x' Y) ≥ ?δ ^ 3 / 128}"
interpret Γ: ulgraph "X2" "?E_loops"
```

We can transport information easily using locale definitions



```
have neighborhood_unchanged: "∀ x ∈ ?X1. neighbors_ss x Y = H.neighbors_ss x Y"
  using neighbors_ss_def H.neighbors_ss_def vert_adj_def H.vert_adj_def by auto
then have degree_unchanged: "∀ x ∈ ?X1. degree x = H.degree x"
  using H.degree_neighbors_ssX degree_neighbors_ssX by auto
```

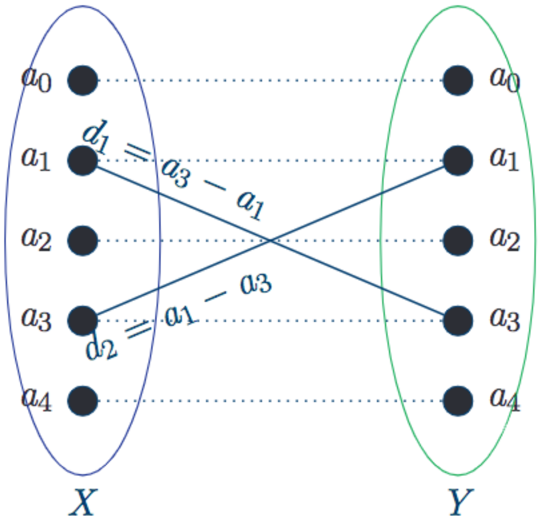
Turning additive structure into graphs

Definition (θ-popular difference)

Given an abelian group G and a finite subset A, define $d \in G$ to be a θ-popular difference if $|\{(a, b) \in A^2 | a - b = d\}| \geq \theta |A|$

Idea: Finding large subsets B, C of $A \subseteq G$ that contain many unique paths of length 3 between them along *popular differences* can give us upper bounds on the size of $|C - B|$.

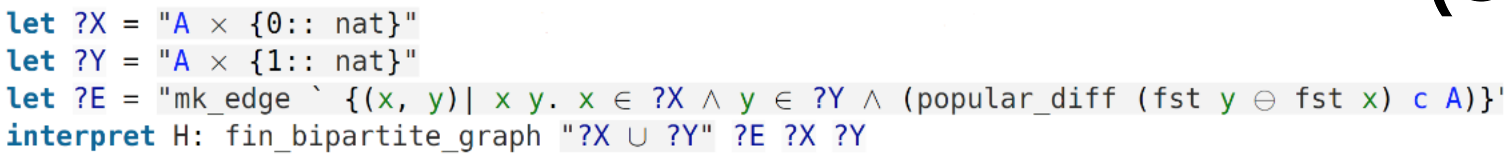
The formal construction



Graph Construct

- Vertices: Let $V = X \cup Y$ where X and Y are copies of $A \subseteq G$
- Edges: $a_i a_j \in E$ if and only if $a_j - a_i$ is c-popular. **(This "glues" algebra and graph theory)**
- Lemma 3.2 then gives large subsets with paths of length 3.

Working in the additive abelian group context:



```
let ?X = "A × {0:: nat}"
let ?Y = "A × {1:: nat}"
let ?E = "mk_edge ` {(x, y)| x y. x ∈ ?X ∧ y ∈ ?Y ∧ (popular_diff (fst y ⊖ fst x) c A)}'
interpret H: fin_bipartite_graph "?X ∪ ?Y" ?E ?X ?Y
```

: need for a new graph theory library with suf. abstract types (Ch. Edmonds)

Lemma 3.5

To successfully use the introduced auxiliary graph by applying Lemma 3.2, we will need to prove that it is reasonably dense, i.e. that for *many* pairs $(a_i, a_j) \in A^2$, $a_i - a_j$ is c-popular.

Lemma (3.5)

Let A be a finite subset of an abelian group G. Suppose A has additive energy $2c$. Then the number of c-popular differences $d \in A - A$ is at least $c|A|$.

Application of Lemma 3.2 to additive combinatorics

Lemma (3.6)

Let A be a finite subset of an abelian group G. Suppose A has additive energy $2c$ for $c > 0$. Then A has subsets B and C with $|B| \geq c^4|A|/16$ and $|C| \geq c^2|A|/4$ such that $|C - B| \leq 2^{13}c^{-15}|A|$.

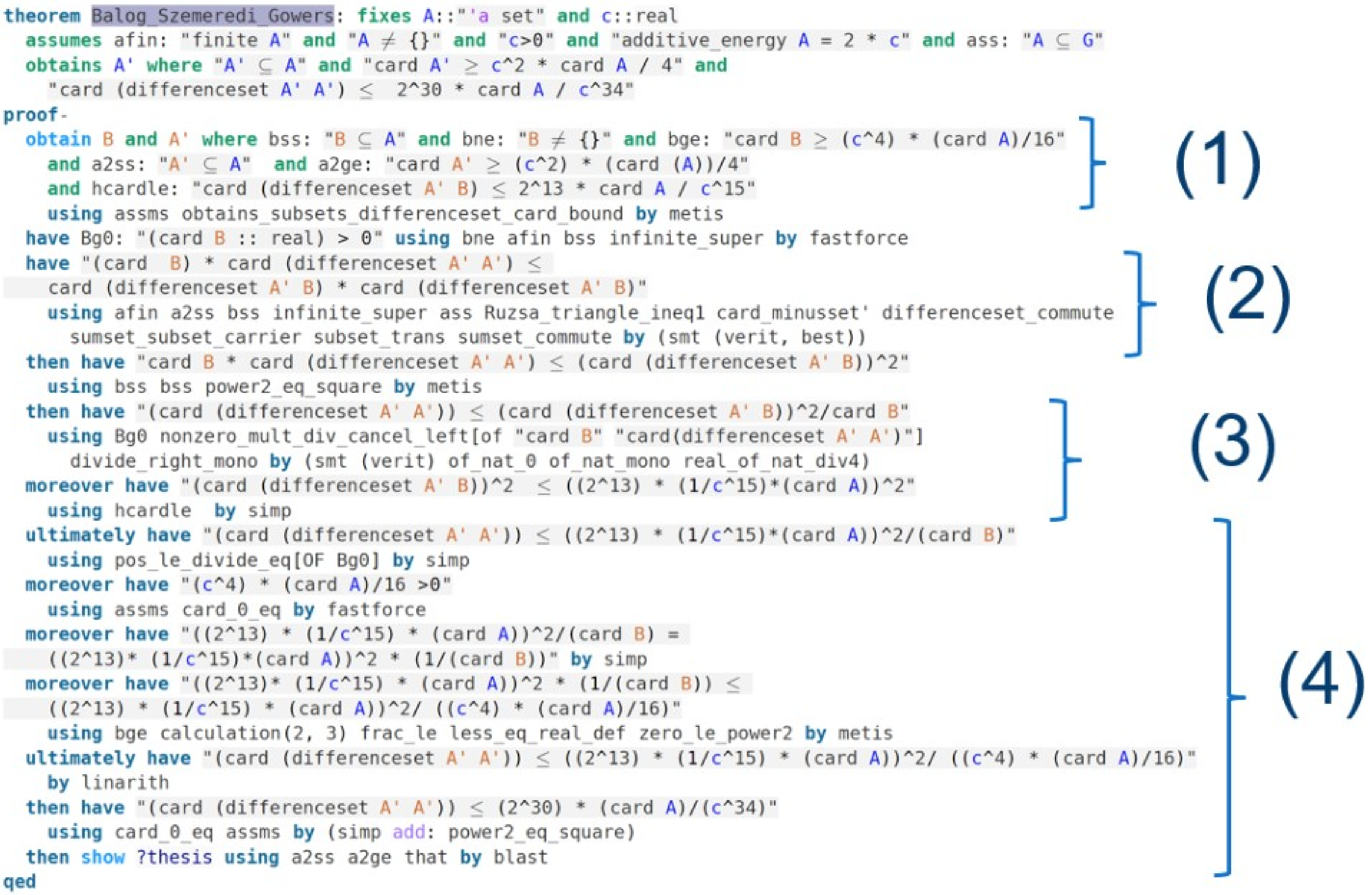
Proof Sketch:

- Create an auxiliary bipartite graph using copies of A and c-popular differences.
- Apply Lemma 3.2 to the auxiliary graph to find $\Omega(|A|^2)$ paths of length 3 between pairs $(x, y) \in B \times C$, i.e. pairs $(z, w) \in A^2$ such that $x - z, z - w, w - y$ are all c-popular.

Application of Lemma 3.2 to additive combinatorics

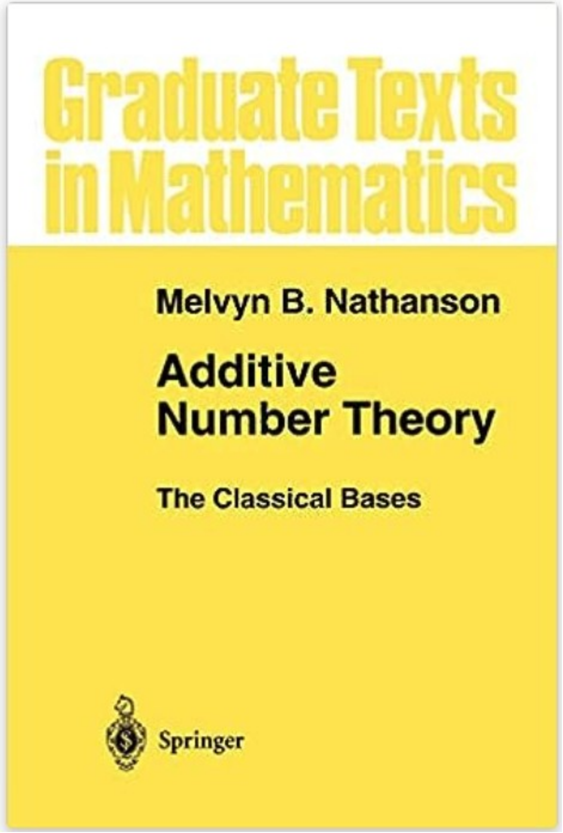
Proof sketch (continued):

- For each such path $xzwy$ in the auxiliary graph, we can find a set of $\Omega(|A|^3)$ sextuples $(p, q, r, s, t, u) \in A^6$ such that $p - q = x - z$, $r - s = z - w$, $t - u = w - y$
- Each such set of sextuples is disjoint for distinct $(z, w) \in A^2$ and (further) distinct $x - y$. Hence we have found distinct subjects of size $\Omega(|A|^5)$ in A^6 for each $x - y \in B - C$, so $|B - C| = O(|A|)$.



```
theorem Balog_Szemeredi_Gowers: fixes A::"'a set" and c::real
  assumes afin: "finite A" and "A ≠ {}" and "c>0" and "additive_energy A = 2 * c" and ass: "A ⊆ G"
  obtains A' where "A' ⊆ A" and "card A' ≥ c^2 * card A / 4" and
    "card (differenceset A' A') ≤  2^30 * card A / c^34"
proof-
  obtain B and A' where bss: "B ⊆ A" and bne: "B ≠ {}" and bge: "card B ≥ (c^4) * (card A)/16"
    and a2ss: "A' ⊆ A"  and a2ge: "card A' ≥ (c^2) * (card (A))/4"
    and hcardle: "card (differenceset A' B) ≤ 2^13 * card A / c^15"
    using assms obtains_subsets_differenceset_card_bound by metis
  have Bg0: "(card B :: real) > 0" using bne afin bss infinite_super by fastforce
  have "(card  B) * card (differenceset A' A') ≤
    card (differenceset A' B) * card (differenceset A' B)"
    using afin a2ss bss infinite_super ass Ruzsa_triangle_ineq1 card_minusset' differenceset_commute
      sumset_subset_carrier subset_trans sumset_commute by (smt (verit, best))
  then have "card B * card (differenceset A' A') ≤ (card (differenceset A' B))^2"
    using bss bss power2_eq_square by metis
  then have "(card (differenceset A' A')) ≤ (card (differenceset A' B))^2/card B"
    using Bg0 nonzero_mult_div_cancel_left[of "card B" "card(differenceset A' A')"]
      divide_right_mono by (smt (verit) of_nat_0 of_nat_mono real_of_nat_div4)
  moreover have "(card (differenceset A' B))^2  ≤ ((2^13) * (1/c^15)*(card A))^2"
    using hcardle  by simp
  ultimately have "(card (differenceset A' A')) ≤ ((2^13) * (1/c^15)*(card A))^2/(card B)"
    using pos_le_divide_eq[OF Bg0] by simp
  moreover have "(c^4) * (card A)/16 >0"
    using assms card_0_eq by fastforce
  moreover have "((2^13) * (1/c^15) * (card A))^2/(card B) =
    ((2^13)* (1/c^15)*(card A))^2 * (1/(card B))" by simp
  moreover have "((2^13)* (1/c^15) * (card A))^2 * (1/(card B)) ≤
    ((2^13) * (1/c^15) * (card A))^2/ ((c^4) * (card A)/16)"
    using bge calculation(2, 3) frac_le less_eq_real_def zero_le_power2 by metis
  ultimately have "(card (differenceset A' A')) ≤ ((2^13) * (1/c^15) * (card A))^2/ ((c^4) * (card A)/16)"
    by linarith
  then have "(card (differenceset A' A')) ≤ (2^30) * (card A)/(c^34)"
    using card_0_eq assms by (simp add: power2_eq_square)
  then show ?thesis using a2ss a2ge that by blast
qed
```

Other recent formalisation work in additive number theory by my students at Cambridge, following Nathanson's book



Graduate Texts in Mathematics

Melvyn B. Nathanson

Additive Number Theory

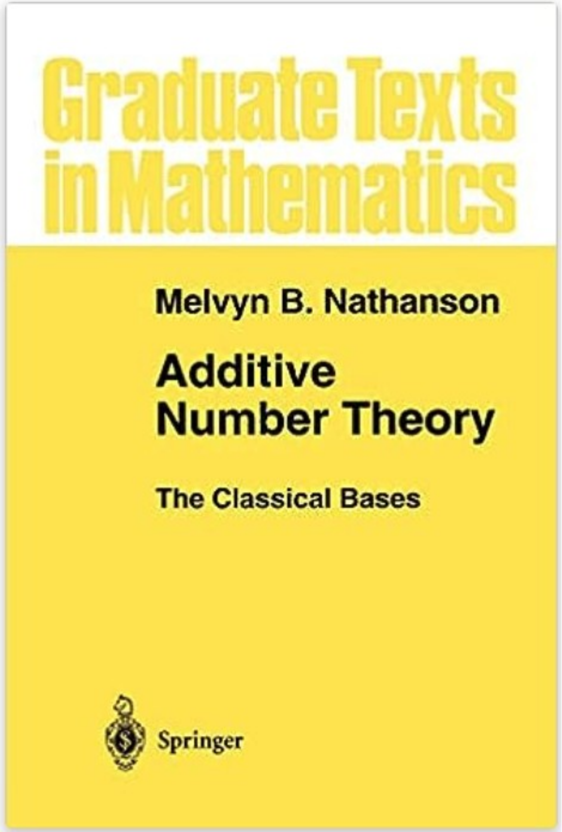
The Classical Bases

Springer

- Ryan Shao: Part III (i.e. MPhil) Project in Advanced Computer Science: "Formalisation of an Upper Bound for the Easier Waring's Problem in Isabelle", 2022-2023 (Honours Pass with Distinction).

- Jamie Chen: Part II (i.e. 3rd year) "Formalising the Wieferich–Kempner Theorem in Isabelle/HOL", 2022-2023.

...just completed (summer 2023):



Graduate Texts in Mathematics

Melvyn B. Nathanson

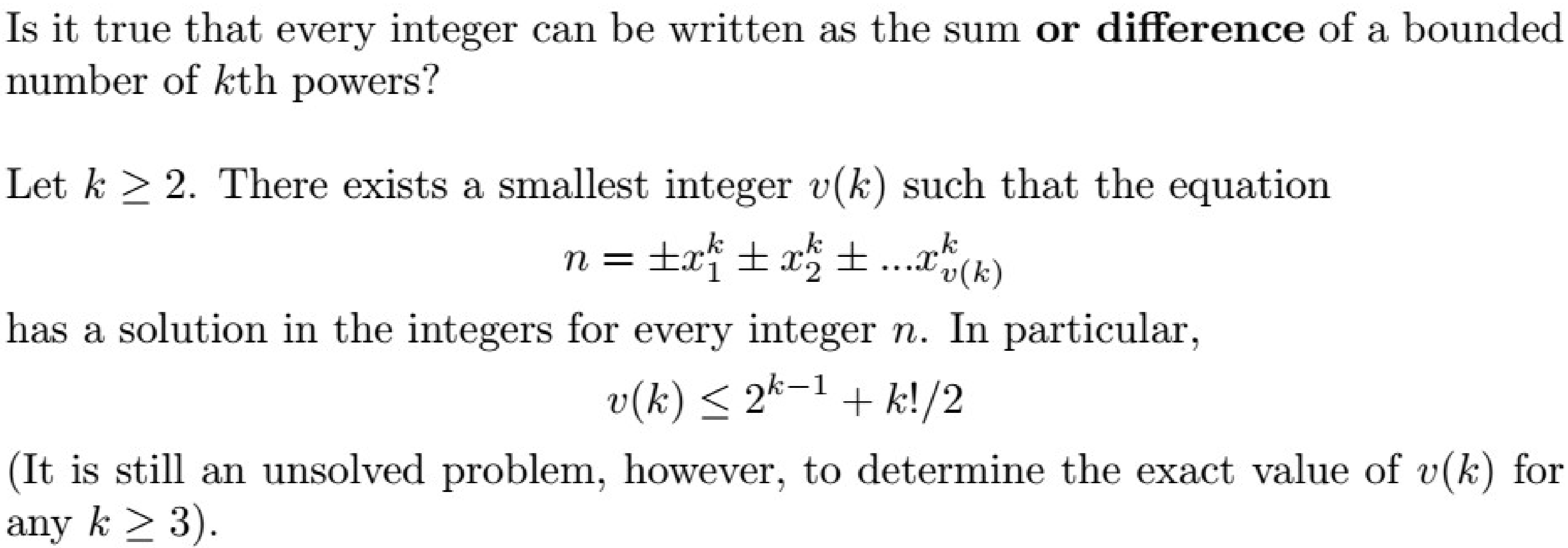
Additive Number Theory

The Classical Bases

Springer

- Kevin Lee and Zhengkun (Chris) Ye, Cambridge Mathematics students (8-week internships with the support of the Cambridge Mathematics Placement (CMP) programme) formalised (versions of) the Polygonal Number Theorem in Isabelle/HOL.

Easier Waring’s Problem



Is it true that every integer can be written as the sum **or difference** of a bounded number of kth powers?

Let $k \geq 2$. There exists a smallest integer $v(k)$ such that the equation

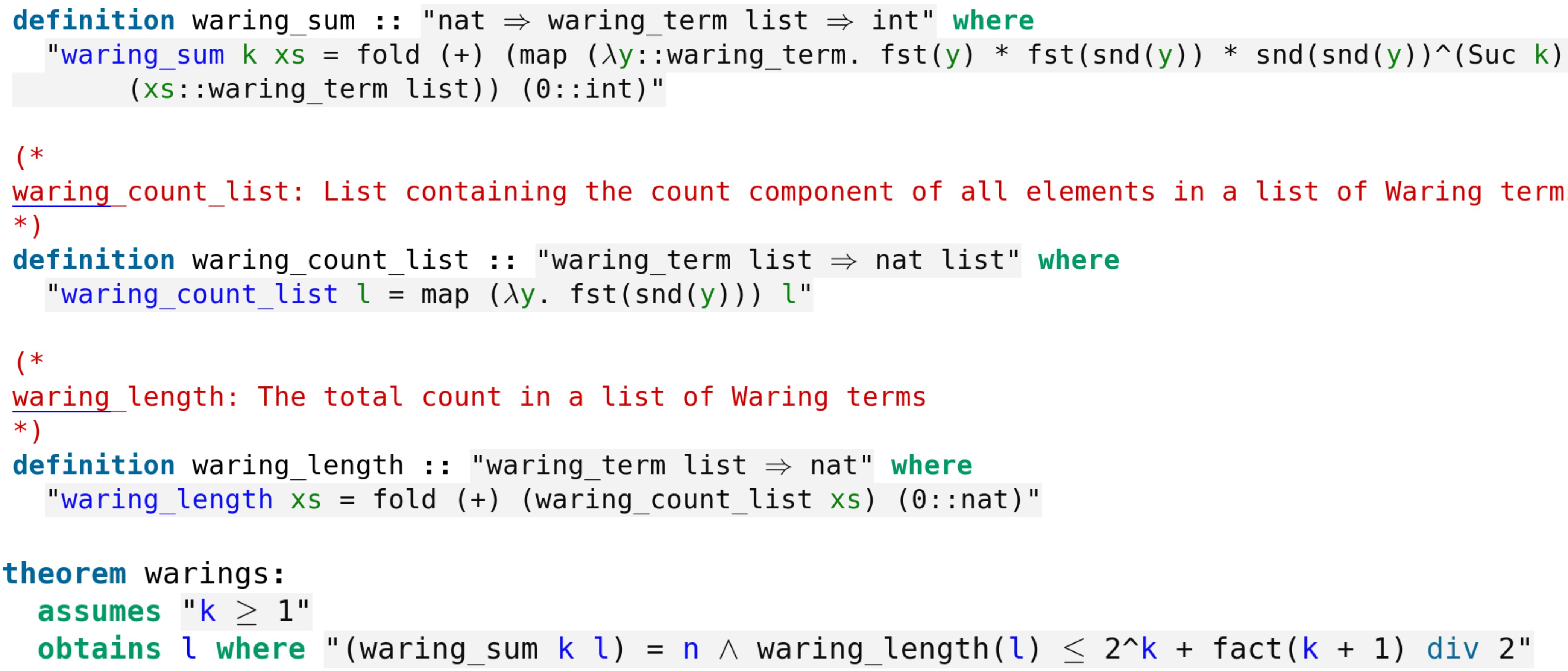
$$n = \pm x_1^k \pm x_2^k \pm ...x_{v(k)}^k$$

has a solution in the integers for every integer n. In particular,

$$v(k) \leq 2^{k-1} + k!/2$$

(It is still an unsolved problem, however, to determine the exact value of $v(k)$ for any $k \geq 3$).

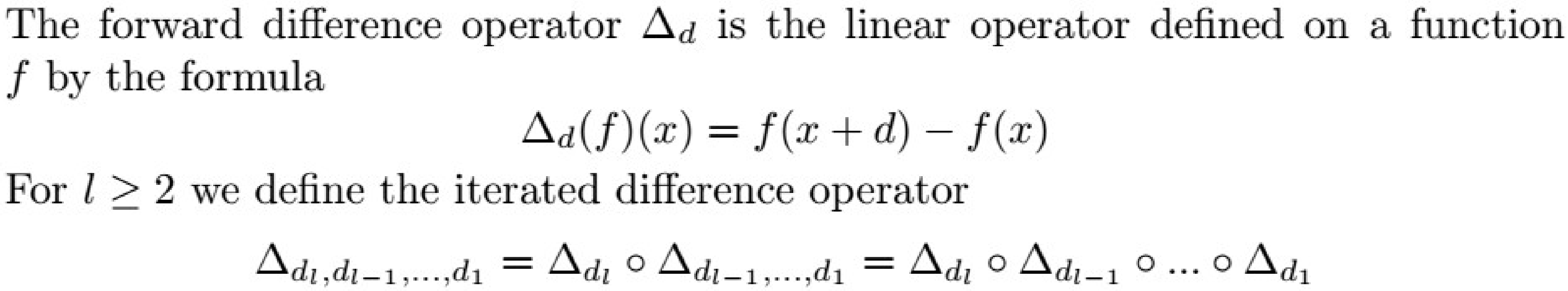
Easier Waring’s Problem



```
definition waring_sum :: "nat ⇒ waring_term list ⇒ int" where
  "waring_sum k xs = fold (+) (map (λy::waring_term. fst(y) * fst(snd(y)) * snd(snd(y))^(Suc k)
        (xs::waring_term list)) (0::int)"

(*
waring_count_list: List containing the count component of all elements in a list of Waring term
*)
definition waring_count_list :: "waring_term list ⇒ nat list" where
  "waring_count_list l = map (λy. fst(snd(y))) l"

(*
waring_length: The total count in a list of Waring terms
*)
definition waring_length :: "waring_term list ⇒ nat" where
  "waring_length xs = fold (+) (waring_count_list xs) (0::nat)"

theorem warings:
  assumes "k ≥ 1"
  obtains l where "(waring_sum k l) = n ∧ waring_length(l) ≤ 2^k + fact(k + 1) div 2"
```

(about 1400 lines of code)

Easier Waring’s Problem



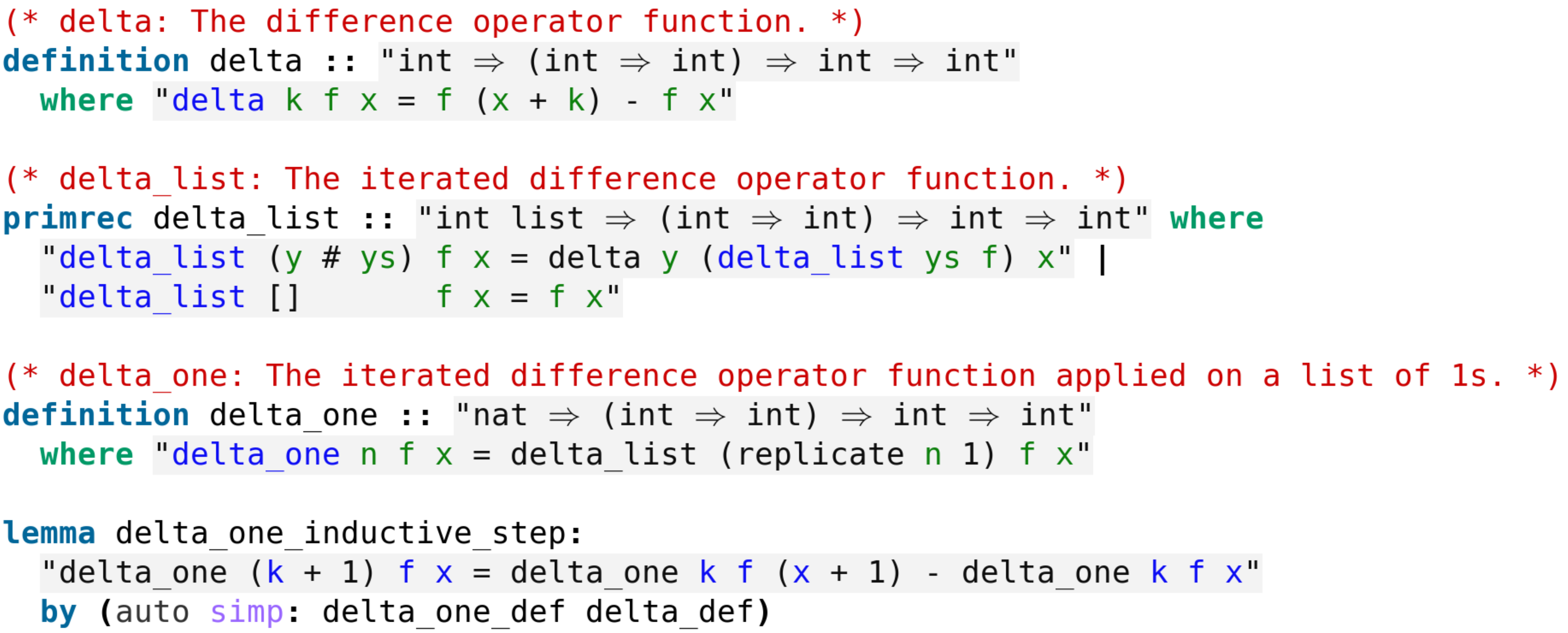
The forward difference operator Δ_d is the linear operator defined on a function f by the formula

$$\Delta_d(f)(x) = f(x+d) - f(x)$$

For $l \geq 2$ we define the iterated difference operator

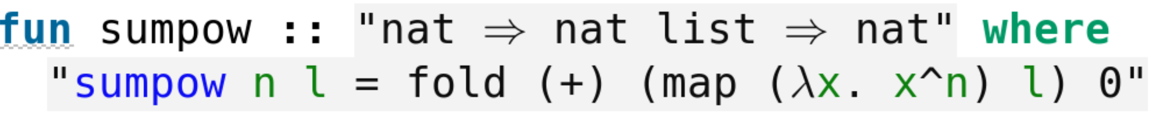
$$\Delta_{d_l, d_{l-1}, \ldots, d_1} = \Delta_{d_l} \circ \Delta_{d_{l-1}, \ldots, d_1} = \Delta_{d_l} \circ \Delta_{d_{l-1}} \circ \ldots \circ \Delta_{d_1}$$

Easier Waring’s Problem

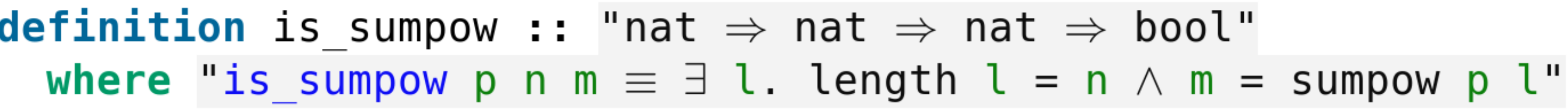


```
(* delta: The difference operator function. *)
definition delta :: "int ⇒ (int ⇒ int) ⇒ int ⇒ int"
  where "delta k f x = f (x + k) - f x"

(* delta_list: The iterated difference operator function. *)
primrec delta_list :: "int list ⇒ (int ⇒ int) ⇒ int ⇒ int" where
  "delta_list (y # ys) f x = delta y (delta_list ys f) x" |
  "delta_list []       f x = f x"

(* delta_one: The iterated difference operator function applied on a list of 1s. *)
definition delta_one :: "nat ⇒ (int ⇒ int) ⇒ int ⇒ int"
  where "delta_one n f x = delta_list (replicate n 1) f x"

lemma delta_one_inductive_step:
  "delta_one (k + 1) f x = delta_one k f (x + 1) - delta_one k f x"
  by (auto simp: delta_one_def delta_def)
```

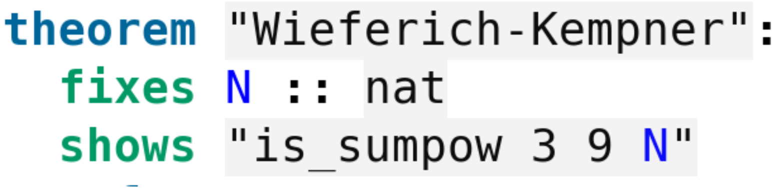
The Wieferich-Kempner Theorem



```
fun sumpow :: "nat ⇒ nat list ⇒ nat" where
  "sumpow n l = fold (+) (map (λx. x^n) l) 0"
```



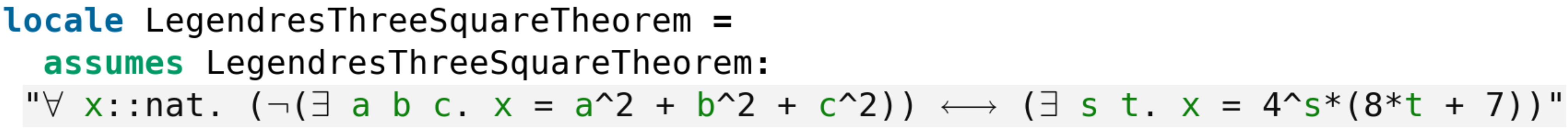
```
definition is_sumpow :: "nat ⇒ nat ⇒ nat ⇒ bool"
  where "is_sumpow p n m ≡ ∃ l. length l = n ∧ m = sumpow p l"
```



```
theorem "Wieferich-Kempner":
  fixes N :: nat
  shows "is_sumpow 3 9 N"
```

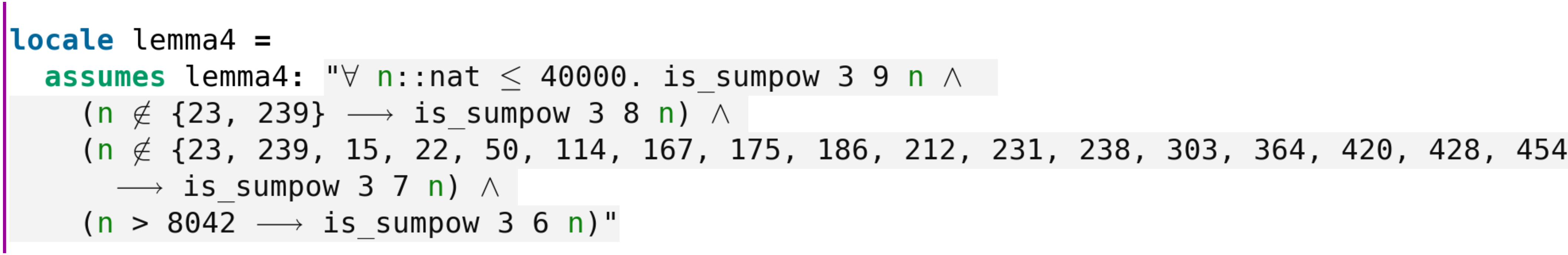
(about 1100 lines of code
BUT...)

The Wieferich-Kempner Theorem



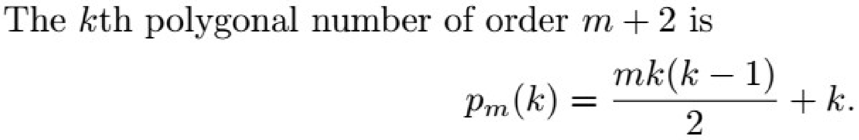
```
locale LegendresThreeSquareTheorem =
  assumes LegendresThreeSquareTheorem:
 "∀ x::nat. (¬(∃ a b c. x = a^2 + b^2 + c^2)) ⟷ (∃ s t. x = 4^s*(8*t + 7))"
```

Both the Wieferich–Kempner Theorem and the Polygonal Number Theorem build on Legendre’s Three Squares Theorem, which has been recently formalised in Isabelle by Anton Danilkin and Loïc Chevalier!



```
locale lemma4 =
  assumes lemma4: "∀ n::nat ≤ 40000. is_sumpow 3 9 n ∧
    (n ∉ {23, 239} ⟶ is_sumpow 3 8 n) ∧
    (n ∉ {23, 239, 15, 22, 50, 114, 167, 175, 186, 212, 231, 238, 303, 364, 420, 428, 454
      ⟶ is_sumpow 3 7 n) ∧
    (n > 8042 ⟶ is_sumpow 3 6 n)"
```

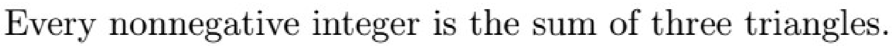
The Polygonal Number Theorem



The kth polygonal number of order $m+2$ is

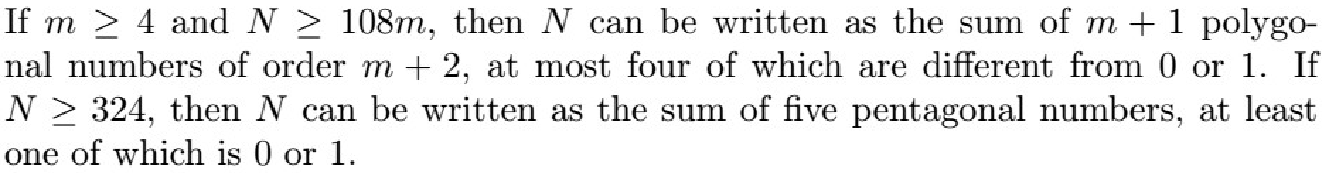
$$p_m(k) = \frac{mk(k-1)}{2} + k.$$

Gauss



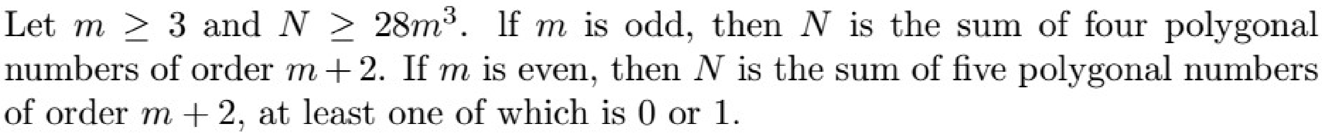
Every nonnegative integer is the sum of three triangles.

Cauchy



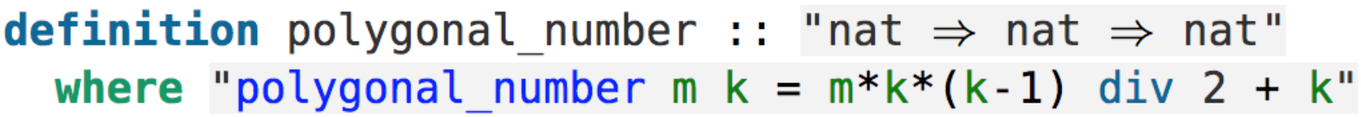
If $m \geq 4$ and $N \geq 108m$, then N can be written as the sum of $m+1$ polygonal numbers of order $m+2$, at most four of which are different from 0 or 1. If $N \geq 324$, then N can be written as the sum of five pentagonal numbers, at least one of which is 0 or 1.

Legendre



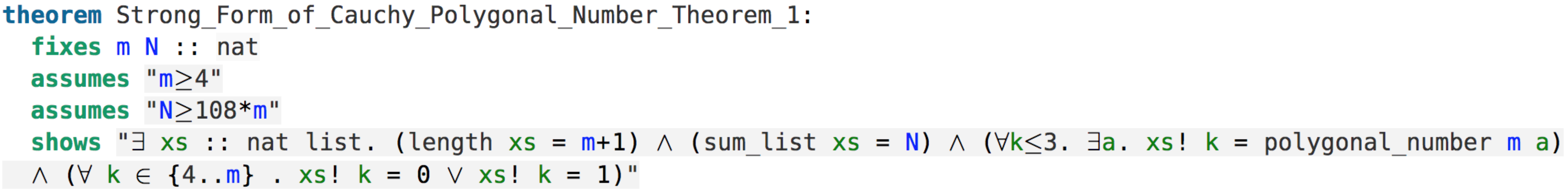
Let $m \geq 3$ and $N \geq 28m^3$. lf m is odd, then N is the sum of four polygonal numbers of order $m+2$. If m is even, then N is the sum of five polygonal numbers of order $m+2$, at least one of which is 0 or 1.

The Polygonal Number Theorem

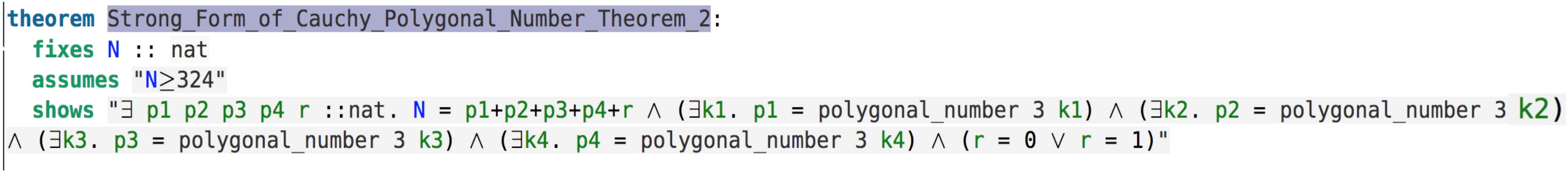


```
definition polygonal_number :: "nat ⇒ nat ⇒ nat"
  where "polygonal_number m k = m*k*(k-1) div 2 + k"
```

Cauchy



```
theorem Strong_Form_of_Cauchy_Polygonal_Number_Theorem_1:
  fixes m N :: nat
  assumes "m≥4"
  assumes "N≥108*m"
  shows "∃ xs :: nat list. (length xs = m+1) ∧ (sum_list xs = N) ∧ (∀k≤3. ∃a. xs! k = polygonal_number m a)
  ∧ (∀ k ∈ {4..m} . xs! k = 0 ∨ xs! k = 1)"
```



```
theorem Strong_Form_of_Cauchy_Polygonal_Number_Theorem_2:
  fixes N :: nat
  assumes "N≥324"
  shows "∃ p1 p2 p3 p4 r ::nat. N = p1+p2+p3+p4+r ∧ (∃k1. p1 = polygonal_number 3 k1) ∧ (∃k2. p2 = polygonal_number 3 k2)
∧ (∃k3. p3 = polygonal_number 3 k3) ∧ (∃k4. p4 = polygonal_number 3 k4) ∧ (r = 0 ∨ r = 1)"
```

Conclusion: Lessons learned

* Formalisation goals accomplished

* Still yet to encounter any material impossible to formalise in simple type theory

* Advanced mathematics within reach

* Locales can be very useful (to capture interaction between different mathematical areas and to “cheat” by including unformalised material as assumptions)

* The formalisation process can reveal the need for a higher level of abstraction in prerequisites (e.g. new graph theory library by Chelsea Edmonds)

Conclusion: Lessons learned

* Sledgehammer’s automation is practical and efficient

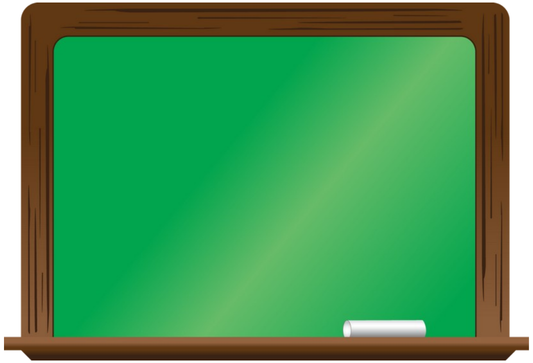
* Students can learn Isabelle very fast and formalise advanced material successfully

* Collaborative work, filling in library gaps

* We still need: better automation, efficient organisation and management of libraries (definitions, elementary properties and basics, advanced results)

* Our libraries can grow increasingly fast!

Thank you



+

=