Recurrence-Driven Summations in Automated Deduction

Visa Nummelin[1], Jasmin Blanchette[1,2], and Sander R. Dahmen[1]

[1] Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
{visa.nummelin,j.c.blanchette,s.r.dahmen}@vu.nl
[2] Max-Planck-Institut für Informatik, Saarland Informatics Campus, Saarbrücken, Germany
jasmin.blanchette@mpi-inf.mpg.de

Abstract. Many problems in mathematics and computer science involve summations. We present a procedure that automatically proves equalities involving finite summations, inspired by the theory of holonomic sequences. The procedure is designed to be interleaved with the activities of a higher-order automatic theorem prover. It performs an induction and automatically solves the induction step, leaving the base cases to the theorem prover. We show how to integrate the procedure with the superposition calculus.

1 Introduction

Finite summations—that is, summations $\sum_{i=m}^{n} t_i$ over finitely many terms t_i—are ubiquitous in mathematics and computer science, but they are poorly supported by automatic theorem provers. One reason is that summations are higher-order, whereas most theorem provers are first-order.

In recent years, we have seen the rise of higher-order provers, such as cvc5 [1], Leo-III [18], Vampire [3], veriT [1], and Zipperposition [2]. With these provers, a summation $\sum_{i=m}^{n} t_i$ can be represented as the λ-term $\mathsf{sum}\ m\ n\ (\lambda i.\ t_i)$; the traditional $\sum$ syntax can be seen as syntactic sugar. However, despite the use of heuristics [19, Sect. 4], higher-order provers are ill-equipped to reason inductively. A simple problem such as proving the equation $\sum_{i=0}^{\mathsf{n}} i = \mathsf{n}(\mathsf{n}+1)/2$ is a formidable challenge for them, even if we supply them with axioms characterizing $+$, $\cdot$, $/$, and $\sum$ together with an induction principle.

In this paper, we introduce a procedure for proving such equations in a higher-order prover. The procedure is triggered by a proof goal of the form $k\sum s + t = u$, possibly with some side conditions (Sect. 2). In a refutational prover, the equation would be negated, as $k\sum s + t \neq u$, and would correspond to the negated conjecture, a problem axiom, or some clause derived by the prover.

Our procedure translates facts about summations to linear recurrences. These recurrences have almost the same form as multivariate holonomic sequences [21], which strongly inspired this work. Each recurrence is associated with a multivariate sequence—a sequence with one or more indices.

The procedure has three steps:

1. *Initialization* (Sect. 3): Heuristically choose terms in the goal to generalize and perform induction on. Among the problem axioms, select those of a suitable form as initial recurrences for the procedure.
2. *Propagation* (Sect. 4): From the initial recurrences, compute recurrences corresponding to the goal. For $+$, $\cdot$, and $\sum$ expressions occurring in the goal, recurrences are computed from the recurrences of their direct subexpressions.
3. *Induction* (Sect. 5): If the final recurrences for the goal involve only the goal and no other sequences, use them for induction. If they make the difference of successive values of $k\sum s + t - u$ constant, this establishes the induction step. Then reduce the goal $k\sum s + t = u$ to a set of base cases and give these to the prover.

Propagation and induction apply holonomic techniques almost as a black box. Initialization connects them to the overall proof search.

For example, to prove $\sum_{i=0}^{n} i = n(n+1)/2$, the procedure would transform the equation into recurrences and find out that the difference $\sum_{i=0}^{n} i - n(n+1)/2$ remains constant as n increases, thereby establishing the induction step. If that difference is constantly 0, we get $\sum_{i=0}^{n} i = n(n+1)/2$. It suffices to prove a number of base cases, which are left to the prover. This example is very simple, but the procedure scales up to more sophisticated problems (Sect. 6). An implementation is under way in the Zipperposition prover [2].

The procedure treats $\sum$ as an interpreted (built-in) symbol. Summation yields a value in a commutative group, or in a ring if multiplication is present. The commutative group or ring gives us $+$, $\cdot$, and $-$. All these are also interpreted, as are numerals. Integers, including indices, can multiply group elements.

Compared with Wilf–Zeilberger pairs [20] and other methods (Sect. 7), the main benefit of our procedure is that it is widely applicable and may help prove not only difficult summations in a restrictive form but also easier summations in a more general form, which is useful in a general-purpose theorem prover.

We refer to our technical report [17] for more details.

2 Inference Rule

Our procedure can be integrated into a theorem prover, where it takes the form of an inference rule that complements the prover's existing rules. We present such a rule for provers based on the superposition calculus. Our technical report discusses a similar integration with satisfiability modulo theories (SMT) and tableaux.

Superposition is a calculus for classical first-order logic. It reasons about formulas in conjunctive normal form, or clauses, that contain the equality predicate. It has been extended to higher-order logic, as combinatory superposition [4] and λ-superposition [2]. Zipperposition, based on λ-superposition, has won in the higher-order division of CASC in 2020, 2021, and 2022.

Superposition is a refutational calculus: Starting from the negation of the conjecture taken as an axiom, it tries to derive a contradiction—the empty, false

clause $\bot$. This is achieved by saturation: systematically deriving consequences of existing clauses, aiming to eventually derive $\bot$.

Superposition consists of inference rules that generate new clauses based on the presence of suitable premises in the clause set. The premises of a rule are shown above a horizontal bar; the derived conclusion is shown below the bar. We extend superposition with a rule that interfaces with our summation procedure:

$$\frac{C_1 \;\cdots\; C_l \quad C \vee t \neq u}{D \vee C \vee \bigvee_{\vec{b} \in B} \{\vec{s} \mapsto \vec{b}\}(t \neq u)}\ \textsc{Summation}$$

The following side conditions apply:

- t is an expression that can be brought into the general form $k\sum_{i=m}^{n} t' + t''$ using the group properties (e.g., t could be $3 - \sum_{i=0}^{n} n^i$), or using the ring properties if t is known to be in a ring;
- the procedure performs an induction on the generalized subterms $\vec{s}$ of t and u (Sect. 3)—these $\vec{s}$ may not be subterms of each other;
- the procedure succeeds at proving the induction step based on initial recurrences derived from the premises $C_1, \ldots, C_l$ (Sect. 3) and propagation of these recurrences (Sect. 4);
- the procedure identifies B as the finite set of base cases of the induction, where each case is a vector $\vec{b}$ of terms of the same length as $\vec{s}$ (Sect. 5); and
- the subclause D captures side conditions determined by the procedure.

In the rule's conclusion, the notation $\{\vec{s} \mapsto \vec{b}\}(t \neq u)$ stands for the disequation $t \neq u$ in which every occurrence of a subterm s_i, where s_i is among the components of $\vec{s}$, is simultaneously replaced by the corresponding term b_i from $\vec{b}$. For example, $\{\mathsf{f\ a} \mapsto 0,\ \mathsf{b} \mapsto 1\}(\mathsf{g\ (f\ a)\ b}) = \mathsf{g}\ 0\ 1$. The intuition behind the rule is that the conclusion, which is about only a finite set of base cases, should be easier to refute than the rightmost premise. Sometimes the set B of base cases might even be empty, if the procedure needed no induction. As for the premises $C_1, \ldots, C_l$, they can contain useful information about $\vec{s}$, often about bounds.

Example 1. Consider the goal $\forall n.\ n \geq 0 \longrightarrow \sum_{i=0}^{n} i = n(n{+}1)/2$. Clausification produces two clauses: $\mathsf{n} \geq 0$ and $\sum_{i=0}^{\mathsf{n}} i \neq \mathsf{n}(\mathsf{n}{+}1)/2$ where n is a Skolem constant. The following inference applies our procedure to solve the induction step of the second premise:

$$\frac{\mathsf{n} \geq 0 \quad \sum_{i=0}^{\mathsf{n}} i \neq \mathsf{n}(\mathsf{n}+1)/2}{\sum_{i=0}^{0} i \neq 0(0+1)/2 \vee \sum_{i=0}^{1} i \neq 1(1+1)/2 \vee \sum_{i=0}^{2} i \neq 2(2+1)/2}\ \textsc{Summation}$$

The induction is on the subterm n, which is generalized to a variable. The procedure generates three base cases, for $\mathsf{n} \in \{0, 1, 2\}$. These are easy to reduce to falsity using superposition's simplification machinery and basic equations to expand finite sums. Built-in arithmetic procedures [9,10] can easily reason about the arithmetic expressions. At the end, the prover derives the empty clause $\bot$.

3 Initialization

The first step of our procedure is to recognize the special structure of recurrences. Variables on which we can perform induction appear as Skolem constants in the negated goal. Further opportunities for induction can be created by generalizing complex terms. Also as part of this step, we must choose which terms represent (multivariate) sequences and which clauses represent their recurrences.

Theory Detection. We require the necessary theory of summation to be predefined, either as a specially tagged subset of the input or as a theory that is hard-coded in the prover. Specifically, this means the inductive theory of integers, axioms for commutative groups (including multiplication by integers), and the definition of summation from 0 by $\sum_{n=0}^{-1} f_n = 0$ and $\sum_{n=0}^{m+1} f_n = \sum_{n=0}^{m} f_n + f_{m+1}$ even for negative $m \in \mathbb{Z}$. Other bounded intervals are then expressed as differences: $\sum_{n=k}^{m} f_n = \sum_{n=0}^{m} f_n - \sum_{n=0}^{k-1} f_n$.

Multiplication may be present or absent. We search for candidates that qualify as distributive binary operators from the negated goal. Often candidates would have the form $\mathsf{f}\ l\ r$ where l and r are left and right operands, respectively, but we generally allow the form $\mathsf{f}\ \vec{a}\ l\ \vec{b}\ r\ \vec{c}$ with a function symbol f and variable-free term tuples $\vec{a}, \vec{b}, \vec{c}$. This form is motivated by the encoding of types as parameters (e.g., $\mathsf{mul}\ \mathsf{real}\ l\ r$) and lifting of operations (e.g., lift $\mathsf{mul}_{\mathsf{real}}\ l\ r$) to quotients, products, etc. For each candidate, we can try to prove left and right distributivity by syntactically looking for that axiom or by running another instance of the prover. Detecting associativity, commutativity, and the unit element can also be useful.

Term Generalization. Term generalization transforms Skolem constants or complex terms to variables and then performs an induction on the variables. We propose a straightforward heuristic that generalizes each nonnumeral subterm s of type $\mathbb{Z}$ occurring in the negated goal such that s is variable-free even after applying the generalization heuristics to its subterms. For example, in the following variable-free integer terms, the underlined subterms would be generalized: $\underline{\mathsf{a}}$, 123, $\underline{\mathsf{f}\ 0\ 2}$, $2\mathsf{f}\ (\underline{\mathsf{g}\ (-1)})\ (-3\underline{\mathsf{a}})$, $\mathsf{f}\ 1\ (\mathsf{g}\ (\underline{\mathsf{a}} + 1))$.

Let $\vec{s}$ be the subterms chosen for generalization. Then based on the negated goal $C \vee t \neq u$ (as in the SUMMATION rule), generalization sets up the goal

$$\forall \vec{n} \in N.\ \{\vec{s} \mapsto \vec{n}\}(t - u) = 0$$

where $N \subseteq \mathbb{Z}^k$ collects the bounds of $\vec{s}$ (often $N = \mathbb{N}^k$). We accept conditional proofs of this goal.

The generalization makes it possible to use induction to prove that the goal sequence term $\{\vec{s} \mapsto \vec{n}\}(t - u)$—a function of $\vec{n}$—vanishes on N. We try to prove the generalized goal assuming $\neg C$ and some extra conditions E such as base cases of induction. Then instantiating $\vec{n} := \vec{s}$, we conclude $C \vee \neg E \vee t = u$. This together with the negated goal $C \vee t \neq u$ implies a conclusion of the form $C \vee \neg E$ for the SUMMATION rule. Note that C is not generalized.

The set N embodies knowledge about $\vec{s}$ that we find among existing clauses $C_1, \ldots, C_l$ and the condition $\neg C$. The free variables of $\neg C$ are interpreted as constants, and they can also occur in $\vec{s}$. For example, assume that $\vec{n} = n$ and $\vec{s} = \mathsf{f}\ s'$ and that the generalized goal contains the factorial $n!$. Its recurrence must be in a conditional clause—e.g., $(m+1)! = (m+1)m! \vee 0 \not\leq m$. To use this for $n!$, we need $n \geq 0$, which we can ensure using N if we find a bounding clause $\mathsf{f}\ s' \geq 0$ or its generalization such as $\mathsf{f}\ m \geq 0$ where m is a free variable. The more we know about $\vec{s}$, the more recurrences we can get. At the same time, N must allow induction, so we keep it convex by considering only coordinatewise bounds of $\vec{s}$.

Choice of Initial Recurrences. Semantically, the recurrences we look for are multivariate heterogeneous linear finite-fixed-step equations with polynomial coefficients. An archetypical example is

$$(n^2+1)f_{n+2,m+1} + mf_{n+1,m} = nmf_{n,m+1} - 2h_{n,m} + (m-n)h_{n,m-1}$$

Here, the sequences f, h are bivariate, and the sequence indices are all of the form $n+k$ or $m+k$ for numerals $k \in \mathbb{Z}$, amounting to finite fixed steps.

As sketched in Sect. 1, we must select some of the problem axioms as initial recurrences for the procedure. This is accomplished as follows. Let there be an edge between two axioms of the form $C \vee s = t$ (where C may be empty) if they both contain a top-level occurrence of the same sequence g (i.e., an occurrence of g that is not nested inside an uninterpreted function symbol). The axioms then form a graph. We take as initial recurrences the connected component of the generalized goal.

By a sequence g we mean the $\mathsf{f}\ \vec{a}$ part of a term of the form $\mathsf{f}\ \vec{a}\ \vec{n}$ where f is an uninterpreted function symbol, $\vec{a}$ is a tuple of variable-free terms, and $\vec{n}$ is a tuple of integer variables or affine (i.e., linear term + constant term) combinations of them. The tuples $\vec{a}$ and $\vec{n}$ may in general be interleaved, but $\vec{n}$ must be nonempty.

In other contexts, an analogous step is known as lemma filtering or premise selection [5, Sect. 2]. Bloat from irrelevant facts is less of an issue in the context of our procedure because it can use only linear recurrences. Beyond this, our simple heuristic does nothing to avoid bloat.

What should we do about conditions such as C in $C \vee \mathsf{f}\ \vec{a}\ \vec{n} = t$? In an implementation, there would be at least three ways to cope with them: We could forbid them and work only with unit equations such as $\mathsf{f}\ \vec{a}\ \vec{n} = t$. We could collect them and put them in the D component of the Summation rule's conclusion. Or we could attempt to prove them when the initial recurrences are selected. In our ongoing implementation, we chose the first option.

4 Propagation

Sequences defined by recurrences with polynomial coefficients and finitely many base case values are holonomic. They are closed under $+$, $\cdot$, $\sum$, and affine substitutions, which makes their equality decidable [21]. From linear recurrences

of arbitrary sequence terms $f_{n,m}$ and $g_{n,m}$, we can recursively derive recurrences for the complex sequence terms $f_{n,m} \pm g_{n,m}$, $f_{n,m} \cdot g_{n,m}$, $\sum_{n=0}^{m} f_{n,m}$, and $f_{an+bm,cn+dm}$ where a, b, c, d are numerals and n, m are variables. The derivation will be achieved by algorithms corresponding to the closure properties of holonomic sequences. We refer to this as *propagation* of the recurrences.

Given a negated goal $C \vee t \neq u$ to refute, generalization turns $t - u$ into a multivariate sequence, and we want to find linear recurrences for the sequence that can be used to inductively prove $t - u = 0$. Starting from the recurrences found in initialization, we propagate them from the bottom up along the tree structure of the term $t - u$. In each propagation, the fine syntactic structure of the terms is ignored and deduction is based only on noncommutative Gröbner basis completion. We start with a brief overview of this technique in our setting.

Gröbner Bases of Recurrence Operators. A (generalized) Gröbner basis is a certain well-behaved generating set of a left ideal of (possibly noncommutative) polynomials. Equivalently, we will view it as a system of polynomial equations that is complete for rewriting. Given a polynomial equation $P = 0$, for every monomial M we get a rewrite rule as follows. Decompose MP as $MP = L + R$ where L is the leading monomial of MP w.r.t. a fixed monomial ordering times its coefficient. Then $L = -R$ gives rise to a rewrite rule $L \to -R$. A system of equations is complete for rewriting if every one of its consequences can be proved via rewriting by these rules.

Example 2. The system $\{ab^2 = a + b, a^2b = a + 1\}$ does not prove its consequence $a^2 = b$ by rewriting. (Instead, subtract the equations multiplied by a or b.) In the other direction, the system's Gröbner basis $\{a^2 = b, b^2 = a + 1\}$ does give rewrite proofs $ab^2 \xrightarrow[b^2=a+1]{} a^2 + a \xrightarrow[a^2=b]{} b + a$ and $a^2b \xrightarrow[a^2=b]{} b^2 \xrightarrow[b^2=a+1]{} a + 1$.

Instead of polynomials, we have finite step recurrences with polynomial coefficients. The finite steps are represented by new indeterminates interpreted to shift sequences. With the coefficients, these create noncommuting polynomials.

Definition 3. *Operator polynomials* are a $\mathbb{Z}$-algebra with composition as product (meaning closed under addition, composition, and integer multiplication) spanned by the multiplier and shift operators:

- The *multiplier* operator M_j of index j multiplies a group-valued multivariate sequence f by the variable n_j of index j: $(M_j f)_{\vec{n}} = n_j f_{\vec{n}}$.
- The *shift* operator $S_j = \{n_j \mapsto n_j + 1\}$ of index j increments the variable n_j of index j of a multivariate sequence f by one: $(S_j f)_{\vec{n}} = f_{\{n_j \mapsto n_j + 1\}\vec{n}}$.

A theory of Gröbner bases exists for various polynomial algebras [13], but a simple sufficient requirement is that all indeterminates X, Y commute up to lower-order terms: $XY - YX \in \mathbb{Z}X + \mathbb{Z}Y + \mathbb{Z}$. The natural choice of taking all multiplier and shift operators as indeterminates satisfies this requirement. Indeed, for any sequence f, we have $(S_j M_j f)_{\vec{n}} = (S_j(\vec{m} \mapsto m_j f_{\vec{m}}))_{\vec{n}} = (n_j + 1)(S_j f)_{\vec{n}} = ((M_j S_j + S_j) f)_{\vec{n}}$ and all other pairs of multipliers and shifts commute exactly:

$$S_i M_j = M_j S_i + \delta_{i,j} S_j \qquad S_i S_j = S_j S_i \qquad M_i M_j = M_j M_i \tag{1}$$

for all i and j, where $\delta_{i,j}$ equals 1 if $i = j$ and 0 otherwise.

A single operator polynomial P_1 perfectly encodes a linear homogeneous recurrence $0 = P_1 g$ of a (syntaxwise atomic) sequence term g. However, we allow any heterogeneous recurrence of the form $0 = \vec{P} \bullet \vec{f} := P_1 f_1 + \cdots + P_k f_k$ where $\vec{f} = (f_1, \ldots, f_k)$ is an arbitrary tuple of different sequence terms. These lead to Gröbner basis theory of modules [14, Sect. 6.4]. The recurrence $0 = \vec{P} \bullet \vec{f}$ is encoded by the *polynomial vector* $\vec{P}$ whose component P_j we will conveniently index as $P_j f_j$ by a sequence term f_j instead of number j so that the order of $\vec{f}$ will not matter. Moreover, we often focus on a chosen main component, say $P_m f_m$, and write the recurrence in the form $0 = P_m f_m + e$ where the *excess terms* $e = \vec{P} \bullet \vec{f} - P_m f_m$ contain all the f_j components except the sequence term f_m. The precise choice of f_m becomes relevant only near the end (Definition 16).

When we consider a formal polynomial algebra (necessary to computing Gröbner bases), we will usually mean vector-valued polynomials with integer coefficients and indeterminates $M_1, M_2, \ldots, S_1, S_2, \ldots$ satisfying (1). Exceptionally, when we propagate to substitution, we consider compositions of shifts formally as further individual indeterminates, as explained above Procedure 10. Apart from this exceptional setting, we fix a choice of monomials as follows.

Definition 4. In our setting, a *monomial* is an expression of the form $M_1^{a_1} \cdots M_k^{a_k} S_1^{b_1} \cdots S_k^{b_k} f$ for a sequence term f and numeral exponents $a_j, b_j \in \mathbb{N}$.

The above monomial form is a polynomial vector with only its f component being nonzero. Due to the (non)commutation relations (1), polynomial vectors can be written as sums of monomials times their integer coefficients. This makes working with these noncommutative polynomials similar to working with commutative ones. A major difference is that monomials are not closed under multiplication by indeterminates, as illustrated by $S_1 \cdot M_1 f = M_1 S_1 f + S_1 f$ for a sequence term f. This complicates the definition of monomial order below, which in turn defines how to interpret a polynomial equation as a rewrite rule.

Definition 5. A *leading monomial* $\operatorname{lead} P$ of a polynomial vector $P \neq 0$ is the largest monomial in P. This is w.r.t. a *monomial order* $\preccurlyeq$, meaning a well-founded total order on monomials such that if $A \preccurlyeq B$, then $\operatorname{lead} XA \preccurlyeq \operatorname{lead} XB$ for all indeterminates X (both multipliers and shifts).

Buchberger's algorithm to compute Gröbner bases (also in a noncommutative vector-valued context) is compatible with the framework of saturation. It repeatedly derives from polynomial recurrences $P = 0$ and $R = 0$ a new equation $AP - BR = 0$ where coefficient–indeterminate products A, B make the leading monomials of AP and BR cancel. Whereas most general unifiers suffice in superposition, here the least divisible A, B suffice and do not even depend on the noncommutation relations. Unlike many saturation procedures, the completion into a Gröbner basis always terminates. The standard termination proof amounts to applying noetherianity of commutative polynomials over $\mathbb{Z}$ or Dickson's lemma [13].

Definition 6. Let X be a set of indeterminates and sequence terms. An *X-elimination order* is a monomial order such that monomials containing factors from X are larger than other monomials.

Our default choice for an X-elimination order is to compare the total degree over X where sequence terms, like indeterminates, contribute 1 to the degree. Then we break ties by comparing the total degree over the complement of X, and then the tuples of exponents lexicographically, considering smaller exponents as larger [8, Chapter 2 §2].

Procedure 7. *Eliminating* indeterminates and/or sequence terms X from a set R of recurrences means computing a Gröbner basis G of R w.r.t. an X-elimination order and then discarding all elements from G whose polynomial vector contains a subterm from X.

While in principle any Gröbner basis would suffice for elimination, our default choice is to compute the reduced Gröbner basis (i.e., the fully simplified one). This and the choice of elimination order may have a large impact because all propagations will be based on elimination. Next, we adapt them from the four closure properties of holonomic sequences by carrying excess terms along.

Propagation to Addition. Let us start with addition.

Procedure 8. Let f and g be sequence terms, and let h be the formal name of their addition $f + g$. The associated recurrences F of f and G of g are propagated to those of h by eliminating f and g from $F \cup G \cup \{h = f + g\}$. (By Procedure 7, this amounts to computing a Gröbner basis for these equations and then discarding the equations containing f or g.)

Actually, the same propagation technique works if $f + g$ is replaced by any expression in the general recurrence format $\vec{P} \bullet \vec{l}$ (a dot product of operator polynomials $\vec{P}$ and sequence terms $\vec{l}$). The key is that the defining equation $h = \vec{P} \bullet \vec{l}$ is again a linear recurrence.

Example 9. Consider the goal $\sum_{j=0}^{n} a_j = g_n + a_0$ given $g_0 = 0$ and $g_{n+2} = g_n + a_{n+1} + a_{n+2}$ for all $n \in \mathbb{N}$. The defining recurrence of g can be written using the operator polynomials as $S_1^2 g = g + S_1 a + S_1^2 a$. Let us also make explicit the defining recurrence of the sum $f_n := \sum_{j=0}^{n} a_j$—namely, $S_1 f = f + S_1 a$. We must prove that $h_n := g_n + a_0 - f_n$ is 0. To achieve this, we propagate recurrences to h using the elimination procedure described above (Procedure 7) and the total-degree-

based (f, g)-elimination order with $f \prec g$. Leading monomials are shown in bold:

$$\begin{array}{lll}
 & 0 = \boldsymbol{S_1^2 g} - g - S_1 a - S_1^2 a & \text{recurrence of } g \\
-S_1^2 & 0 = \boldsymbol{g} + a_0 - f - h & \text{definition of } h \\
\hline
 & 0 = -g - S_1 a - S_1^2 a - a_0 + \boldsymbol{S_1^2 f} + S_1^2 h & \\
-S_1 & 0 = \boldsymbol{S_1 f} - f - S_1 a & \text{recurrence of } f \\
\hline
 & 0 = -g - S_1 a - a_0 + S_1^2 h + \boldsymbol{S_1 f} & \\
- & 0 = \boldsymbol{S_1 f} - f - S_1 a & \text{recurrence of } f \\
\hline
 & 0 = -\boldsymbol{g} - a_0 + S_1^2 h + f & \\
+ & 0 = \boldsymbol{g} + a_0 - f - h & \text{definition of } h \\
\hline
 & 0 = \boldsymbol{S_1^2 h} - h &
\end{array}$$

In this example, $h_{n+2} - h_n = 0$ is the only recurrence that does not contain f and g, so we discard the rest of the Gröbner basis calculation. Since $h_{n+2} - h_n = 0$ contains only the sequence h, we can use it to prove the induction step (of size 2) of $\forall n.\ h_n = 0$. We are then left with the two base cases $h_0 = 0$ and $h_1 = 0$, which the SUMMATION inference would include in its conclusion without auxiliary symbols as $\sum_{j=0}^{0} a_j \neq g_0 + a_0 \vee \sum_{j=0}^{1} a_j \neq g_1 + a_0$.

Propagation to Substitution. Consider a numeral matrix $a = [a_{kj}]_{kj} \in \mathbb{Z}^{d \times D}$ and a vector $\vec{b} \in \mathbb{Z}^d$. They characterize an affine substitution $\sigma = \{\vec{n} \mapsto a\vec{n} + \vec{b}\}$ $= \{n_k \mapsto \sum_{j=1}^{D} a_{kj} n_j + b_k \mid 1 \leq k \leq d\}$. As an operator on sequences, σ performs an affine change of variables: $(\sigma f)_{\vec{n}} = f_{a\vec{n}+\vec{b}}$. We restrict our substitutions to affine ones to stay within the propagation framework.

Clearly, any recurrence $Pf = 0$ of f implies $\sigma P f = 0$. Moreover, if $\sigma P = P'\sigma$, then $P'\sigma f = 0$ gives a recurrence of σf. Finding such an operator polynomial P' for a general P can be reduced to finding an operator polynomial P'_X satisfying $\sigma X = P'_X \sigma$ for every indeterminant X. In other words, we must push all indeterminants X leftwards. For multipliers, we have $\sigma(M_1, \ldots, M_d) = (a(M_1, \ldots, M_D) + \vec{b})\sigma$. In contrast, shifts are easily pushed only rightwards—namely, $S_j \sigma = \sigma S_1^{a_{1j}} \cdots S_d^{a_{dj}}$. Consequently, the recurrences of f must be first expressed in terms of the composite shifts $\mathbb{S}_j := S_1^{a_{1j}} \cdots S_d^{a_{dj}}$. As operators, these satisfy the (non)commutation relations

$$\mathbb{S}_j M_k = (M_k + a_{kj})\mathbb{S}_j \qquad \mathbb{S}_i \mathbb{S}_j = \mathbb{S}_j \mathbb{S}_i \qquad \mathbb{S}_i S_j = S_j \mathbb{S}_i \tag{2}$$

This makes the $\mathbb{S}_j$'s suitable as indeterminates in Gröbner basis computations.

Accordingly, for propagation to substitution, we enlarge our formal polynomial algebra to also contain the indeterminates $\mathbb{S}_1, \mathbb{S}_2, \ldots$ satisfying the relations (2), while also keeping (1). We note that *as operators*, the indeterminates further satisfy (essentially by definition) for all sequences f the relations

$$(\mathbb{S}_j \textstyle\prod_{k:\, a_{kj}<0} S_k^{|a_{kj}|}) f = (\prod_{k:\, a_{kj}>0} S_k^{a_{kj}}) f \quad \text{for } j \in \{1, \ldots, D\} \tag{3}$$

While we could have defined our polynomial algebra to incorporate these relations as well, we chose to *not* do this, thereby creating a distinction between the

semantic operators and the syntactic polynomial algebra. Instead, we will add these relations to the system of recurrence equations of which we compute the Gröbner basis. Finally, we extend our notion of *monomial* from Definition 4 to mean any expression of the form $M_1^{x_1} \cdots M_d^{x_d} S_1^{y_1} \cdots S_d^{y_d} \mathbb{S}_1^{z_1} \cdots \mathbb{S}_D^{z_D} f$ where the exponents $x_j, y_j, z_j \in \mathbb{N}$ are numerals and f is a sequence term.

Procedure 10. Recurrences of a sequence term f are propagated to its affine substitution $(\sigma f)_{\vec{n}} = f_{a\vec{n}+\vec{b}}$ as follows, using the extended polynomial algebra described above. Eliminate each S_k from the system of polynomial equations containing both the recurrences of f and the relations (3). Every resulting recurrence $P(\vec{M}, \vec{\mathbb{S}})f + e = 0$ implies a recurrence $P(a\vec{M} + \vec{b}, \vec{S})\sigma f + \sigma e = 0$ of σf where we have collected the indeterminates into vectors and where e are excess terms that do not contain f.

Example 11. Consider the formula $\sum_{n_1=0}^{n_2} \binom{n_1}{n_2-n_1} = F_{n_2+1}$ where the Fibonacci numbers are defined by $F_1 = F_2 = 1$ and $(S_1^2 - S_1 - 1)F = 0$. For the binomial coefficient $\binom{\cdot}{n_1,n_2} = \binom{n_1}{n_2} = \frac{n_1!}{n_2!(n_1-n_2)!}$, the recurrence from Pascal's triangle reads in variable-free notation as $(S_1S_2 - S_2 - 1)\binom{\cdot}{\cdot} = 0$ and extends $\binom{n_1}{n_2}$ from $0 \le n_2 \le n_1$ to all integers n_2 and natural numbers n_1. Moreover, we have $\binom{n_1}{n_2} = \frac{n_1}{n_2}\binom{n_1-1}{n_2-1}$ or equivalently $((M_2+1)S_1S_2 - M_1 - 1)\binom{\cdot}{\cdot} = 0$. We want to propagate these to the substitution $\sigma = \{n_1 \mapsto n_1, n_2 \mapsto n_2 - n_1\}$. We have $S_1\sigma = \sigma S_1 S_2^{-1}$ and $S_2\sigma = \sigma S_2$. So we introduce for $S_1S_2^{-1}$ and S_2 the new indeterminates $\mathbb{S}_1$ and $\mathbb{S}_2$ whose characterizing recurrences (3) read

$$(\mathbb{S}_1S_2 - S_1)\binom{\cdot}{\cdot} = 0 \quad \text{(i)} \qquad\qquad (\mathbb{S}_2 - S_2)\binom{\cdot}{\cdot} = 0 \quad \text{(ii)}$$

Next, we eliminate S_1, S_2 in favor of $\mathbb{S}_1, \mathbb{S}_2$. Here, (ii) immediately rewrites every S_2 to $\mathbb{S}_2$ and then (i) becomes $(-S_1 + \mathbb{S}_1\mathbb{S}_2)\binom{\cdot}{\cdot} = 0$, which rewrites every S_1. The remaining steps to complete a Gröbner basis w.r.t. some total-degree order are irrelevant for what we want to illustrate. We factor the result for readability:

$$\begin{aligned}
(-S_1 + \mathbb{S}_1\mathbb{S}_2)\tbinom{\cdot}{\cdot} &= 0 & (-S_2 + \mathbb{S}_2)\tbinom{\cdot}{\cdot} &= 0\\
(\mathbb{S}_1\mathbb{S}_2^2 - \mathbb{S}_2 - 1)\tbinom{\cdot}{\cdot} &= 0 & ((M_2+1)\mathbb{S}_2 - M_1 + M_2)\tbinom{\cdot}{\cdot} &= 0\\
((M_1+1)\mathbb{S}_1\mathbb{S}_2 - (M_1 - M_2 + 2)\mathbb{S}_1 - M_1 - 1)\tbinom{\cdot}{\cdot} &= 0\\
((M_1 - M_2 + 1)(M_1 - M_2 + 2)\mathbb{S}_1 - M_1M_2 - M_2)\tbinom{\cdot}{\cdot} &= 0
\end{aligned}$$

Now σ maps the lowest four recurrences to recurrences of $f_{n_1,n_2} = \binom{n_1}{n_2-n_1}$ below:

$$\begin{aligned}
(S_1S_2^2 - S_2 - 1)f &= 0 & ((M_2 - M_1 + 1)S_2 - 2M_1 + M_2)f &= 0\\
((M_1+1)S_1S_2 - (2M_1 - M_2 + 2)S_1 - M_1 - 1)f &= 0\\
((2M_1 - M_2 + 1)(2M_1 - M_2 + 2)S_1 - (M_1+1)(M_2 - M_1))f &= 0
\end{aligned}$$

The next step would be to propagate to the summation. We postpone it to Example 14, after we have presented the required theory.

Propagation to Product. Let $\cdot$ be ring multiplication or more generally a group bihomomorphism. Consider sequence terms f and g such that their pointwise product $fg = f \cdot g$ is defined. If f and g depend on disjoint sets of variables, recurrences of fg are essentially a union of recurrences of f and g. Namely, let $Pf + e = 0$ be any recurrence of f where P is an operator polynomial on the variables of f and the excess terms e do not contain f. Then $P(fg) + eg = 0$ because g is effectively a constant to P, and similarly for recurrences of g. With the help of this special case, propagation to product can be reduced to propagation to substitution, as explained below.

Procedure 12. Let f and g be sequence terms parameterized by the variables $\vec{n} = (n_j)_{j=1}^k$. Let $\vec{m} = (n_{j+k})_{j=1}^k$ be a tuple of fresh variables. The recurrences of f and g are propagated to their pointwise product fg in two steps. First, the recurrences of the variable-disjoint product $f_{\vec{n}} g_{\vec{m}}$ are those of $f_{\vec{n}}$ and $g_{\vec{m}}$ multiplied by $g_{\vec{m}}$ or $f_{\vec{n}}$ respectively. Then the recurrences of $f_{\vec{n}} g_{\vec{n}} = \{\vec{m} \mapsto \vec{n}\}(f_{\vec{n}} g_{\vec{m}})$ are found by propagating to substitution according to Procedure 10.

Propagation to Summation. We finally consider as the last type of propagation the summations $\sum_{n_1=0}^{n_2} f_{\vec{n}}$. Without loss of generality, we assume that the variables are numbered so that the sum acts on the first two. Similarly to above, we consider the consequence $\sum_{n_1=0}^{n_2} P f_{\vec{n}} + \sum_{n_1=0}^{n_2} e_{\vec{n}} = 0$ of a recurrence $Pf + e = 0$ of the sequence term f where P is an operator polynomial and e are excess terms. We want to find an operator polynomial P' such that $\sum_{n_1=0}^{n_2} P$ becomes $P' \sum_{n_1=0}^{n_2}$ up to excess terms. Like for substitutions, finding such a P' for P can be reduced to finding an operator polynomial P'_X satisfying $\sum_{n_1=0}^{n_2} X = P'_X \sum_{n_1=0}^{n_2}$ up to excess terms for every indeterminant X. The result will be a recurrence $P' \sum_{n_1=0}^{n_2} f_{\vec{n}} + e' = 0$ of $\sum_{n_1=0}^{n_2} f_{\vec{n}}$.

Procedure 13. Recurrences of a sequence term f are propagated to its sum $\sum_{n_1=0}^{n_2} f_{\vec{n}}$ as follows. First, eliminate multipliers M_1 from all recurrences of f. Every resulting recurrence $Pf + e = 0$ implies $\sum_{n_1=0}^{n_2} P f_{\vec{n}} + \sum_{n_1=0}^{n_2} e_{\vec{n}} = 0$. Here, P is an operator polynomial that does not contain M_1, and the excess terms e do not contain f. Next, each of these recurrences is rewritten into the form $P' \sum_{n_1=0}^{n_2} f_{\vec{n}} + E_0 + E_{n_2} + \sum_{n_1=0}^{n_2} e_{\vec{n}} = 0$ where P' is an operator polynomial and the E_m's are part of excess terms built by applying operator polynomials and the substitution $\{n_1 \mapsto m\}$ to f. This is achieved by commuting $\sum_{n_1=0}^{n_2}$ with indeterminates other than S_1 and S_2. These two indeterminates are instead handled by the formulas

$$\sum_{n_1=0}^{n_2} S_1 g_{\vec{n}} = \sum_{n_1=0}^{n_2} g_{\vec{n}} + \{n_1 \mapsto n_2 + 1\} g_{\vec{n}} - \{n_1 \mapsto 0\} g_{\vec{n}}$$
$$\sum_{n_1=0}^{n_2} S_2 g_{\vec{n}} = S_2 \sum_{n_1=0}^{n_2} g_{\vec{n}} - S_2 \{n_1 \mapsto n_2\} g_{\vec{n}}$$

Example 14. Let us continue the proof of $\sum_{n_1=0}^{n_2} \binom{n_1}{n_2-n_1} = F_{n_2+1}$ from Example 11. There we found for the summand $f_{n_1,n_2} = \binom{n_1}{n_2-n_1}$ a recurrence $(S_1 S_2^2 - S_2 - 1) f = 0$. It is actually the only recurrence after eliminating M_1 as a first step

of propagation to summation. Next, we set S_1 to 1 using a telescoping identity:

$$\textstyle\sum_{n_1=0}^{n_2} S_1 S_2^2 f = \sum_{n_1=0}^{n_2} S_2^2 f + \{n_1 \mapsto n_2\} S_1 S_2^2 f - \{n_1 \mapsto 0\} S_2^2 f$$

Then we push leftwards the remaining shifts S_2:

$$\begin{aligned}\textstyle\sum_{n_1=0}^{n_2}(S_2^2 - S_2)f &= S_2 \textstyle\sum_{n_1=0}^{n_2}(S_2 - 1)f - S_2\{n_1 \mapsto n_2\}(S_2 - 1)f \\ &= (S_2^2 - S_2)\textstyle\sum_{n_1=0}^{n_2} f - S_2^2\{n_1 \mapsto n_2\}f - S_2\{n_1 \mapsto n_2\}(S_2 - 1)f\end{aligned}$$

Hence, in total we have

$$\begin{array}{llll}
\multicolumn{4}{l}{\textstyle\sum_{n_1=0}^{n_2}(S_1 S_2^2 - S_2 - 1)f - (S_2^2 - S_2 - 1)\sum_{n_1=0}^{n_2} f} \\
= \{n_1 \mapsto n_2\} S_1 S_2^2 f & - \{n_1 \mapsto 0\} S_2^2 f & - S_2^2\{n_1 \mapsto n_2\}f & - S_2\{n_1 \mapsto n_2\}(S_2 - 1)f \\
= \binom{n_2+1}{1} & - \binom{0}{n_2+2} & - \binom{n_2+2}{0} & - \binom{n_2+1}{1} + \binom{n_2+1}{0} \\
= \cancel{\binom{n_2+1}{1}} & - 0 & - 1 & - \cancel{\binom{n_2+1}{1}} + 1 \qquad = 0
\end{array}$$

Since $(S_1 S_2^2 - S_2 - 1)f = 0$, we have $(S_2^2 - S_2 - 1)\sum_{n_1=0}^{n_2} f = 0$. Now this is the same recurrence that the shifted Fibonacci numbers F_{n_2+1} satisfy and hence the final propagation to difference gives $(S_2^2 - S_2 - 1)(\sum_{n_1=0}^{n_2} f - F_{n_2+1}) = 0$. This proves an induction step of size 2 and leaves two base cases that can be discharged by a theorem prover.

Iteration on Excess Terms. Let g be the term from the negated goal to be proved to be 0. After propagating along the structure of g, we end up with recurrences of the form $Pg = e$ where P is an operator polynomial and the excess terms e do not contain g. In the rare case where e is not syntactically 0, then $Pg = e$ cannot immediately be used for a proof by induction. A solution is to iterate a full series of propagations with e in place of g to find $P_2 e = e_2$ and conclude $P_2 P g = P_2 e = e_2$, then repeat as long as necessary. So in general, we propagate to e_{j-1} to find $P_j e_{j-1} = e_j$, conclude $P_j \cdots P_2 P g = e_j$, and repeat until we encounter an $e_k = 0$.

We will impose an order on the sequence terms to accomplish three things. First, we get a clean definition of which terms in a recurrence are excess. Second, well-foundedness of the order will guarantee termination of the iteration of full propagations to excess terms. Third, the iterations can be interleaved with basic normalizations such as $\{n_2 \mapsto 2n_1\} M_2 \{n_3 \mapsto 3n_2\} f \to 2M_1\{n_3 \mapsto 6n_1\} f$.

The sequence terms can be viewed as terms over the first-order alphabet with binary function symbol $\cdot$, unary substitution σ and sum $\sum_j^a$ symbols, and a nullary symbol for each multiplier indeterminate and each sequence term with uninterpreted head. Here, σ ranges over affine substitutions, a ranges over affine variable sums (i.e., $a = \vec{b} \bullet \vec{n} + c$ for some numeral vector $\vec{b}$ and numeral c), and the index j encodes the variable bound in the sum. Addition ($+$) is excluded because as $\cdot$, σ, and $\sum_j^a$ are all distributive, we normalize recurrences to the form $c_1 t_1 + \cdots + c_k t_k = 0$ where $c_j \in \mathbb{Z} \setminus \{0\}$ are numeric coefficients and t_j are distinct sequence terms in the above addition-free alphabet.

Definition 15. The *spine* of a sequence term f, denoted by $\operatorname{spine} f$, is the sequence term obtained by erasing operator polynomial indeterminates from it. This means fully reducing the sequence term by the rewrite rules $M_j t, tM_j \to t$, $\{\vec{n} \mapsto b\vec{n} + \vec{c}\}t \to \{\vec{n} \mapsto b\vec{n}\}t$, and $\sum_j^{\vec{c}\bullet\vec{n}+d} t \to \sum_j^{\vec{c}\bullet\vec{n}} t$ where M_j is a multiplier symbol, b is a numeral matrix, $\vec{c}$ is a numeral vector, and d is a numeral.

We can easily describe how each propagation step changes the spines of the involved sequence terms. Propagation to the addition $f + g$ produces only spines e_f and e_g in the resulting recurrences, where e_f denotes a spine of a term from a recurrence of f and analogously for e_g. Moreover, propagation to the substitution σf produces σe_f, propagation to the product fg produces $(\operatorname{spine} f)e_g$ and $e_f(\operatorname{spine} g)$, and propagation to the summation $\sum_{n_1=0}^{n_2} f$ produces $\{n_1 \mapsto 0\}$ $(\operatorname{spine} f)$, $\{n_1 \mapsto n_2\}(\operatorname{spine} f)$, and $\sum_{n_1=0}^{n_2} e_f$, where e_f and e_g are as above.

Since term orders are compatible with contexts, comparing spines using a term order will almost automatically keep excess terms smaller than the main terms over propagation steps. Given how spines change under propagation, it suffices to additionally require that the term order $>$ satisfy $\sum_j^a t > \sigma t$ for substitutions σ and sequence terms t. Below we consider a more specific term order, which will be able to orient simplification rules as well.

Definition 16. Fix any transfinite Knuth–Bendix order [15] with exactly three different weights $W\sum_{n=0}^{a} > W(\cdot) > 3W\sigma > 0$ and all argument coefficients set to 2. Moreover, nullary multiplier symbols must have equal weights, and substitutions with fewer bindings must have lower precedence than those with more bindings. The *excess* (*partial*) *order* on the sequence terms is obtained by comparing the spines of terms using this fixed order. *Excess terms* of a recurrence are all its nonmaximal sequence terms w.r.t. the excess order.

The weights for the excess order are arranged to be compatible with normalization, which pushes substitutions to leaves and pulls summations towards the root of the term tree. The resulting normal form is simply the typical way of writing sequence terms without operator notation. It is also the normal form of the rewrite system consisting of the applicable associativity and/or commutativity rules of $\cdot$ as well as the following rules:

$$\begin{aligned}
1t, t1, \{\}t &\to t & (\sigma \cup \{n_j \mapsto a\})u &\to \sigma u \\
(\sigma \cup \{n_j \mapsto a\})M_j &\to a & \sigma\sigma' t &\to (\sigma \circ \sigma')t \\
\sigma(ts) &\to \sigma t \cdot \sigma s & \sigma\textstyle\sum_j^a t &\to \sum_j^{\sigma a} \sigma t \\
s \cdot \textstyle\sum_j^a t &\to \sum_j^a st & (\textstyle\sum_j^a s) \cdot t &\to \sum_j^a st \\
\multicolumn{4}{c}{\textstyle\sum_j^c t \to \{n_j \mapsto 0\}t + \cdots + \{n_j \mapsto c\}t} \\
\multicolumn{4}{c}{\textstyle\sum_j^d t \to -\{n_j \mapsto -1\}t - \cdots - \{n_j \mapsto d+1\}t}
\end{aligned}$$

where s, t, u are sequence terms, u does not contain the variable n_j, a is an affine variable sum, M_j is a multiplier symbol, σ, σ' are affine substitutions, the numeral c is nonnegative, and the numeral d is negative.

These rules produce additions, which must be interpreted as follows. For any rule above of the general form $t_0 \to c_1 t_1 + \cdots + c_k t_k$, the actual rewrite that takes place on the entire recurrence is $f[t_0] + R = 0 \to c_1 f[t_1] + \cdots + c_k f[t_k] + R = 0$ where the terms c_j are numeral coefficients, the terms $f[t_j]$ are sequence terms with a distinguished subterm t_j, and R is the sum of the remaining terms in the recurrence. This is analogous to how monomial reductions lift to polynomials in Gröbner basis theory.

To conclude termination, it suffices to prove that t_0 dominates each of $t_1, \ldots, t_k$ individually. The proof is in our technical report. It makes apparent our choices of weights and argument coefficients for the transfinite Knuth–Bendix order.

5 Induction

After propagation, we consider all recurrences $Pg = 0$ of the goal sequence term g to be proved to be 0. In exceptionally fortunate cases, the operator polynomial P is ± 1 and we are unconditionally done because muliplication by the ± 1 maps is invertible in any group. This naturally happens when the objective is to prove a recurrence that this method derives as a substep anyway. Otherwise, we apply induction and leave as conditions the base cases as well as invertibility of the multiplication maps associated with leading monomials' coefficients.

A common case is that variables range over natural numbers and we have a final recurrence with, ignoring multiplier indeterminates, leading monomial $S_1^{b_1} \cdots S_k^{b_k} g$ w.r.t. any monomial order. Then the values $\bigcup_{j=1}^{k} \{\vec{n} \in \mathbb{N}^k \mid n_j < b_j\}$ suffice for the base cases. As a union of stacked hyperplanes that is infinite unless $k \leq 1$, but it corresponds to only $\sum_{j=1}^{k} b_j$ one-variable substitutions $\{n_j \mapsto a\}$ for $1 \leq j \leq k$ and $0 \leq a < b_j$. If our eager generalization produced variables that do not participate in their induction (i.e., their b_j's are 0), they are replaced back to their original values.

If there is more than one applicable final recurrence, we take the intersection of their base value sets w.r.t. the same monomial order. To see that it works, consider any point outside the intersection. It is a nonbase point w.r.t. some final recurrence and hence the induction step can be taken by the recurrence.

To represent the intersection as substitutions, we distribute it over the hyperplane stack unions. This results in a union of hyperline stacks of the form $N(J, \vec{b}) := \{\vec{n} \in \mathbb{N}^k \mid n_j < b_j \text{ for all } j \in J\}$ where $\vec{b}$ and $J \subseteq \{1, \ldots, k\}$ vary. One such stack is represented by $\prod_{j \in J} b_j$ substitutions $\{n_j \mapsto a_j \mid j \in J\}$ enumerating choices $0 \leq a_j < b_j$. Unfortunately, distribution duplicates some base cases. To compensate, if $I \subseteq J$ and $\vec{b} \geq \vec{c}$ pointwise, then $N(I, \vec{b}) \supseteq N(J, \vec{c})$, so that $N(J, \vec{c})$ can be removed in favor of $N(I, \vec{b})$.

If a variable $n \in \mathbb{Z}$ is unbounded, we perform two inductions on the rays: $0 \leq n$ and $n < b$ if b base cases are needed. The backward induction on $n < b$ can be transformed into an induction on $\mathbb{N}$ by the change of variables $n \mapsto b - 1 - n$.

6 Examples

Since our procedure is based on the theory of holonomic sequences, it can prove the induction step of formulas from that theory, such as the binomial formula and Vandermonde's convolution:

$$(a+b)^h = \sum_{n=0}^{h} \binom{h}{n} a^n b^{h-n} \qquad \binom{a+b}{h} = \sum_{n=0}^{h} \binom{a}{n}\binom{b}{h-n}$$

See also Example 11. Heterogeneous recurrences enable proving elementary general sequence formulas, such as Example 9 and the following:

$$\sum_{n=0}^{h} f_{h-n} = \sum_{n=0}^{h} f_n \qquad \sum_{n=0}^{h}\sum_{k=0}^{n} f_{n,k} = \sum_{k=0}^{h}\sum_{n=k}^{h} f_{n,k}$$

If we ignore the holonomic base case requirements, the procedure is more widely applicable but, because it is not a decision procedure, success is harder to predict [7]. We can for example prove the induction steps of Abel's binomial formula and of some Stirling number identities:

$$(a+b)^h = \sum_{n=0}^{h} \binom{h}{n} a(a-n)^{n-1}(b+n)^{h-n} \qquad h^k/h! = \sum_{n=0}^{h} \left\{ {k \atop n} \right\} /(h-n)!$$

Here, the Stirling numbers of the second kind $\left\{ {k \atop n} \right\}$ are one of many special non-holonomic sequences that frequently arise in combinatorics.

As further demonstration, we apply our procedure to the last formula. For convenience, we will use the name of a variable also to denote its multiplier operator. Moreover, we will use the uppercase version of the name of a variable to denote its shift operator. The defining recurrence of the Stirling numbers then reads $(KN - (n+1)N - 1)\left\{ {k \atop n} \right\} = 0$ for $k, n \geq 0$, where K and N denote the shift operators for the variables k and n, the leftmost n denotes the multiplier for the variable n, and the rightmost n is the variable itself. This recurrence is complemented by the initial values $\left\{ {0 \atop 0} \right\} = 1$ and $\left\{ {n \atop 0} \right\} = \left\{ {0 \atop n} \right\} = 0$ if $n \neq 0$.

Starting from the right, the inverse $m!^{-1}$ of the factorial satisfies the recurrence $(mM + M - 1)m!^{-1} = 0$ that holds for all $m \in \mathbb{Z}$ by extension. This must be found in the initialization step because there is no propagation to division. Propagation to the substitution $\{m \mapsto h - n\}$ then gives the following recurrences, factored for clarity:

$$((h-n+1)H - 1)(h-n)!^{-1} = 0 \qquad (N - h + n)(h-n)!^{-1} = 0$$

To propagate to product, we consider $\left\{ {k_1 \atop n_1} \right\}$ and $(h_2 - n_2)!^{-1}$ with variables renamed apart. We must propagate to the substitution $\{n_j \mapsto n,\ h_j \mapsto h,\ k_j \mapsto k \mid j \in \{1,2\}\}$ the recurrences of $\left\{ {k_1 \atop n_1} \right\}(h_2 - n_2)!^{-1}$ given by the following five operator polynomials:

$$K_1 N_1 - (n_1+1)N_1 - 1,\ H_1 - 1 \qquad \text{from } \left\{ {k_1 \atop n_1} \right\}$$
$$(h_2 - n_2 + 1)H_2 - 1,\ N_2 - h_2 + n_2,\ K_2 - 1 \qquad \text{from } (h_2 - n_2)!^{-1}$$

We added here the trivial recurrences given by $H_1 - 1$ and $K_2 - 1$ implied by the independence from h_1 and k_2. Among the defining recurrences (3) of the

compound shift indeterminates N, H, K, the recurrence $H_1H_2 - H$ simplifies to $H_2 - H$ by $H_1 - 1$ and $K_1K_2 - K$ to $K_1 - K$ by $K_2 - 1$. (In other words, the factorwise renaming of already disjoint variables h and k amounts to renaming in the entire product.) The third compound shift recurrence, $N_1N_2 - N$, simplifies to $(h_2 - n_2)N_1 - N$ by $N_2 - h_2 + n_2$. The part of the Gröbner basis with only compound shifts is then straightforwardly finished with the result $\{KN - (n_1 + 1)N - h_2 + n_2,\ (h_2 - n_2 + 1)H - 1\}$. Hence this propagation step yields

$$(KN - (n+1)N - h + n)\frac{\{{k \atop n}\}}{(h-n)!} = 0 \qquad ((h - n + 1)H - 1)\frac{\{{k \atop n}\}}{(h-n)!} = 0$$

To sum over n, we first eliminate n from the previous two recurrences and conclude $(H(K - h) + (N - 1)(KH - (h + 1)H + 1))(\{{k \atop n}\}/(h - n)!) = 0$. The sum has natural boundaries, meaning that the summand vanishes outside them. This guarantees that there will be no excess terms, which we also tediously discover when pulling out the indeterminates:

$$\begin{aligned}
\sum_{n=0}^{h} (N-1)\overbrace{(KH - (h+1)H + 1)}^{P}\frac{\{{k \atop n}\}}{(h-n)!} &= P\Big(\frac{\{{k \atop h+1}\}}{(-1)!} - \frac{\{{k \atop 0}\}}{h!}\Big) \\
&= -\{{k+1 \atop 0}\}/(h+1)! + \{{k \atop 0}\}((h+1)H - 1)h!^{-1} = 0 \\
\sum_{n=0}^{h} H(K-h)\frac{\{{k \atop n}\}}{(h-n)!} - H(K-h)\sum_{n=0}^{h}\frac{\{{k \atop n}\}}{(h-n)!} &= -(K-h)\frac{\{{k \atop h+1}\}}{(-1)!} = 0
\end{aligned}$$

Here, we have extended the inverse of the factorial by its recurrence to have $(-1)!^{-1} = 0$. So we obtain a recurrence $H(K - h)\sum_{n=0}^{h}\{{k \atop n}\}/(h - n)! = 0$ for the left-hand side of our goal. For the right-hand side, we unproblematically obtain $(K - h)(h^k/h!) = 0$. Hence $H(K - h)$ zeros out the difference $h^k/h! - \sum_{n=0}^{h}\{{k \atop n}\}/(h - n)!$.

The largest shift HK of the operator $H(K - h)$ determines the two sets of base cases $h = 0$ and $k = 0$ sufficient for induction. Unlike in the holonomic setting, the base cases are not necessarily easier than the original formula. Hence we leave the base cases to the prover. Here, the first case $h = 0$ is easy because the sum becomes a single term. On the other hand, when $k = 0$ the sum retains its variable length of $h + 1$, but the nonholonomic factor $\{{k \atop n}\}$ simplifies to its initial value $\{{0 \atop n}\}$. With these initial values, only the term with $n = 0$ remains, while generally we might have to repeat Summation.

7 Related Work

Holonomic sequences are closely related to our work. Unlike our approach, which allows infinitely many base cases as long as they are finitely representable (Sect. 5), they are limited to a finite number of base cases. If we relax this limitation, we obtain the homogeneous version of our propagation procedure (i.e., without excess terms), whose theory Chyzak, Kauers, and Salvy laid out [7].

Their rules to predict the number (technically, the Hilbert dimension) of recurrences apply to our setting by ignoring the excess terms. Heterogeneity is discussed with Gröbner bases under the name of module bases [6,11,16]. Its integration into propagations made elementary identities about general sequences automatically provable, which may be of interest for general-purpose theorem provers.

Hypergeometric sums are practically common holonomic sequences that have much faster algorithms available. Gosper's indefinite summation [12] can be applied to compute Wilf–Zeilberger pairs [20], which come with compact proof certificates for definite sum identities. These fast methods admit generalizations to the full holonomic setting. See Koutschan's thesis [14] for an overview.

8 Conclusion

We presented a procedure for proving equations involving summations within an automatic higher-order theorem prover. The procedure is inspired by holonomic sequences and partly generalizes them. It expresses the problem as recurrences and systematically derives new recurrences from existing ones. In case of success, it establishes the induction step of a proof by induction, leaving the base cases to the prover.

As future work, we want to continue implementing the procedure in Zipperposition [2]. We hope that the subsequent practical experiments help us to settle how side conditions of initial recurrences ought to be handled.

Acknowledgment. We thank Pascal Fontaine for his ideas on how to integrate our procedure into SMT and tableaux. We thank Anne Baanen, Pascal Fontaine, and Mark Summerfield for suggesting textual improvements.

Nummelin and Blanchette's research has received funding from the Netherlands Organization for Scientific Research (NWO) under the Vidi program (project No. 016.Vidi.189.037, Lean Forward). Dahmen's research has received funding from the Netherlands Organization for Scientific Research (NWO) under the Vidi program (project No. 639.032.613, New Diophantine Directions).

References

1. Barbosa, H., Reynolds, A., Ouraoui, D.E., Tinelli, C., Barrett, C.W.: Extending SMT solvers to higher-order logic. In: Fontaine, P. (ed.) CADE-27. LNCS, vol. 11716, pp. 35–54. Springer (2019)
2. Bentkamp, A., Blanchette, J., Tourret, S., Vukmirović, P.: Superposition for full higher-order logic. In: Platzer, A., Sutcliffe, G. (eds.) CADE-28. LNCS, vol. 12699, pp. 396–412. Springer (2021)
3. Bhayat, A., Reger, G.: A combinator-based superposition calculus for higher-order logic. In: Peltier, N., Sofronie-Stokkermans, V. (eds.) IJCAR 2020 (1). LNCS, vol. 12166, pp. 278–296. Springer (2020)

4. Bhayat, A., Reger, G.: A combinator-based superposition calculus for higher-order logic. In: Peltier, N., Sofronie-Stokkermans, V. (eds.) IJCAR 2020, Part I. LNCS, vol. 12166, pp. 278–296. Springer (2020)
5. Blanchette, J.C., Kaliszyk, C., Paulson, L.C., Urban, J.: Hammering towards QED. J. Formaliz. Reason. **9**(1), 101–148 (2016)
6. Bueso, J., Gómez-Torrecillas, J., Verschoren, A.: Gröbner bases for modules, pp. 169–202. Springer Netherlands, Dordrecht (2003)
7. Chyzak, F., Kauers, M., Salvy, B.: A non-holonomic systems approach to special function identities. In: Johnson, J.R., Park, H., Kaltofen, E. (eds.) Symbolic and Algebraic Computation, International Symposium, ISSAC 2009, Seoul, Republic of Korea, July 29-31, 2009, Proceedings. pp. 111–118. ACM (2009)
8. Cox, D.A., Little, J., O'Shea, D.: Ideals, varieties, and algorithms; An introduction to computational algebraic geometry and commutative algebra. Undergraduate Texts in Mathematics, Springer, Cham, fourth edn. (2015)
9. Cruanes, S.: Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond. Ph.D. thesis, École polytechnique (2015)
10. Dragan, I., Korovin, K., Kovács, L., Voronkov, A.: Bound propagation for arithmetic reasoning in Vampire. In: Bjørner, N.S., Negru, V., Ida, T., Jebelean, T., Petcu, D., Watt, S.M., Zaharie, D. (eds.) SYNASC 2013. pp. 169–176. IEEE Computer Society (2013)
11. Fajardo, W., Gallego, C., Lezama, O., Reyes, A., Suárez, H., Venegas, H.: Gröbner Bases of Modules, pp. 261–286. Springer International Publishing, Cham (2020)
12. Gosper, R.W.: Decision procedure for indefinite hypergeometric summation. Proc. Natl. Acad. Sci. USA **75**(1), 40–42 (1978)
13. Kandri-Rody, A., Weispfenning, V.: Non-commutative gröbner bases in algebras of solvable type. J. Symb. Comput. **9**(1), 1–26 (1990)
14. Koutschan, C.: Advanced applications of the holonomic systems approach. ACM Communications in Computer Algebra **43**(3/4), 119 (2009)
15. Ludwig, M., Waldmann, U.: An extension of the knuth-bendix ordering with lpo-like properties. In: Dershowitz, N., Voronkov, A. (eds.) Logic for Programming, Artificial Intelligence, and Reasoning, 14th International Conference, LPAR 2007, Yerevan, Armenia, October 15-19, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4790, pp. 348–362. Springer (2007)
16. Maletzky, A., Immler, F.: Gröbner bases of modules and faugère's f_4 algorithm in isabelle/hol. CoRR **abs/1805.00304** (2018)
17. Nummelin, V., Blanchette, J., Dahmen, S.: Automated deduction with recurrence-driven summations (technical report). Technical report (2023), https://lean-forward.github.io/pubs/sums_report.pdf
18. Steen, A., Benzmüller, C.: Extensional higher-order paramodulation in Leo-III. J. Autom. Reason. **65**(6), 775–807 (2021)
19. Vukmirović, P., Bentkamp, A., Blanchette, J., Cruanes, S., Nummelin, V., Tourret, S.: Making higher-order superposition work. In: Platzer, A., Sutcliffe, G. (eds.) CADE-28. LNCS, vol. 12699, pp. 415–432. Springer (2021)
20. Wilf, H.S., Zeilberger, D.: Rational functions certify combinatorial identities. Journal of the American Mathematical Society **3**(1), 147–158 (1990)
21. Zeilberger, D.: A holonomic systems approach to special functions identities. Journal of Computational and Applied Mathematics **32**(3), 321–368 (1990)