Recurrence-Driven Summations in Automated Deduction (Technical Report)

Visa Nummelin[1][0000-0001-7049-6847],
Jasmin Blanchette[1,2][0000-0002-8367-0936], and
Sander R. Dahmen[1][0000-0002-0014-0789]

[1] Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
{visa.nummelin,j.c.blanchette,s.r.dahmen}@vu.nl
[2] Ludwig-Maximilians-Universität München, Munich, Germany
jasmin.blanchette@lmu.de

Abstract. Many problems in mathematics and computer science involve summations. We present a procedure that automatically proves equations 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 superposition, satisfiability modulo theories, and tableaux.

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 [2], E [26], Leo-III [23], Vampire [3], veriT [2], and Zipperposition [25]. With these provers, a summation $\sum_{i=m}^{n} t_i$ can be represented as $\mathsf{sum}\ m\ n\ (\lambda i.\ t_i)$; the traditional $\sum$ syntax can be seen as syntactic sugar for a λ-term. But despite the use of heuristics [25, Sect. 4], higher-order provers are ill-equipped to reason inductively. A simple problem such as $\sum_{i=0}^{\mathsf{n}} i = \mathsf{n}(\mathsf{n}+1)/2$ is a formidable challenge for them, even if we include axioms for $+$, $\cdot$, $/$, and $\sum$ together with an induction principle.

In this report, 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 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 [28], which, while not being a prerequisite for reading this paper, strongly inspired our work. Each recurrence is associated with a multivariate sequence—a sequence with one or more indices. In this report, the word "sequence" generally means "multivariate sequence."

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 $k\sum s + t - u$ constantly 0, this establishes the induction step. Then reduce the goal to a set of base cases and give these to the prover.

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

For example, to prove $\sum_{i=0}^{\mathsf{n}} i = \mathsf{n}(\mathsf{n}+1)/2$, the procedure would transform the equation into recurrences and find out that the difference $\sum_{i=0}^{\mathsf{n}} i - \mathsf{n}(\mathsf{n}+1)/2$ remains constant as n increases, thereby establishing the induction step. If that difference is constantly 0, we get $\sum_{i=0}^{\mathsf{n}} i = \mathsf{n}(\mathsf{n}+1)/2$; in general, 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 [25].

The procedure treats $\sum$ as an interpreted (built-in) symbol. The summation expression evaluates to a value in a commutative group, or a ring if ring multiplication is present. The commutative group or ring gives us $+$, $\cdot$, and $-$. These are also interpreted, as are numerals. Integers, including indices, can multiply group elements. Based on the interpretation, we use the forms $t = u$ and $t - u = 0$ interchangeably.

Compared with Wilf–Zeilberger pairs [27] and other methods (Sect. 7), the main benefit of our procedure is that it goes beyond holonomic sequences and supports both uninterpreted functions and an infinite number of base cases. Our procedure 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. At the heart of our work is the novel combination of techniques from superposition and holonomic sequences, which is visible both in the prover integration (Sect. 2) and in the computation of so-called excess terms (Sect. 4).

2 Inference Rules

Our procedure can be integrated into various types of theorem provers, 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 and briefly discuss a similar integration with two other proof calculi: satisfiability modulo theories (SMT) and tableaux.

2.1 Superposition

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 can be viewed as a generalization of the Knuth–Bendix completion procedure [17] to clausal first-order logic.

Superposition is highly successful; the first-order division of the CADE ATP Systems Competition (CASC) [24] has been dominated by a superposition prover since 2002. The calculus has been extended to higher-order logic, as combinatory superposition [4] and λ-superposition [25]. 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 $\bot$.

The calculus 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:

$$\frac{C_1 \;\; \cdots \;\; C_l \quad C' \vee t[\vec{s}\,] \neq 0}{D \vee C' \vee \bigvee_{\vec{b} \in B} t[\vec{b}\,] \neq 0}\;\textsc{Summation}$$

(By commutativity of $\vee$ and $=$, the rule also applies to a rightmost premise of the form $t'[\vec{s}\,] \neq t[\vec{s}\,] \vee C'$, for example.) The following side conditions apply:

- $t[\vec{s}\,]$ is an expression that can be brought into the general form $k\sum_{i=m}^{n} t' + t''$ using the group properties (e.g., $t[\vec{s}\,]$ could be $3 - \sum_{i=0}^{n} n^i$), or using the ring properties if $t[\vec{s}\,]$ is known to be in a ring;
- the procedure selects, generalizes and performs an induction on the subterms $\vec{s}$ of $t[\vec{s}\,]$ (Sect. 3.2);
- the procedure succeeds at proving the induction step based on initial recurrences derived from the premises $C_1, \ldots, C_l$ (Sect. 3.4) 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 (not necessarily numerals) of the same length as $\vec{s}$ (Sect. 5); and
- the subclause D captures potential conditions determined by the procedure.

It can also be useful to make the rule less prolific by requiring that $t[\vec{s}\,] \neq 0$ is eligible for superposition inferences, a notion that is defined in the standard superposition calculus. In the rule's conclusion, the notation $t[\vec{b}\,]$ stands for the term $t[\vec{s}]$ in which the distinguished occurrence of each subterm s_i, where s_i is among the components of $\vec{s}$, is replaced by the corresponding term b_i from $\vec{b}$. For example, if $t[u_1, u_2] = \mathsf{g}\ u_1\ u_2$, $\vec{s} = (\mathsf{f\ a}, \mathsf{a})$, and $\vec{b} = (0, 1)$, then generalization would transform $t[\mathsf{f\ a}, \mathsf{a}] = \mathsf{g}\ (\mathsf{f\ a})\ \mathsf{a}$ into $t[0, 1] = \mathsf{g}\ 0\ 1$.

The intuition behind the rule is that the conclusion, which is only about 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, meaning that 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 that expand finite sums. Built-in arithmetic procedures, as found for example in Vampire [13] and Zipperposition [12], can easily reason about the arithmetic expressions. At the end, the prover derives the empty clause $\bot$.

2.2 Other Proof Calculi

Besides superposition, our procedure can be integrated into other proof calculi. We briefly sketch integrations into higher-order versions of SMT and tableaux.

SMT solvers [2] combine a SAT solver, which decides propositional logic, and theory solvers. One theory solver is the congruence closure, which decides the theory of equality over uninterpreted functions; for example, from $\mathsf{a} = \mathsf{b}$, it infers that $\mathsf{f\ a} = \mathsf{f\ b}$. The congruence closure maintains a partition of terms in equivalence classes. In addition, it keeps track of disequations.

When the congruence closure becomes aware of a disequation $t[\vec{s}] \neq 0$ where $t[\vec{s}]$ can be brought into the general form $k \sum_{i=m}^{n} t' + t''$, the envisioned integration would invoke our procedure and, on success, add the implication

$$t[\vec{s}] \neq 0 \longrightarrow D \vee \bigvee_{\vec{b} \in B} t[\vec{b}] \neq 0$$

to the set of clauses to refute, where D, B, and $\vec{s}$ are as for superposition. This implication can safely be added regardless of the current SAT model, because it is a tautology in the theory of summations. Even though it is not a decision procedure, in this respect the integration would behave like a theory solver. The procedure could use any initial recurrences derived from clauses $C_1, \ldots, C_l$ found in the current set of clauses.

As for tableaux [7], an integration would work as follows. Whenever a clause $C' \lor t[\vec{s}] \neq 0$ emerges in a leaf node of the proof tree, where $t[\vec{s}]$ can be brought into the form $k \sum_{i=m}^{n} t' + t''$, the integration would invoke our procedure and, on success, add the clause $D \lor C' \lor \bigvee_{\vec{b} \in B} t[\vec{b}] \neq 0$ next to $C' \lor t[\vec{s}] \neq 0$ in the current leaf, where D, B, and $\vec{s}$ are as above. The procedure could use any initial recurrences derived from clauses $C_1, \ldots, C_l$ found in the branch containing the current leaf.

3 Initialization

The first step of our procedure is to recognize the 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.

3.1 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 refers to 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 finite intervals than $[0, m]$ are expressible as differences: $\sum_{n=k}^{m} f_n = \sum_{n=0}^{m} f_n - \sum_{n=0}^{k-1} f_n$.

Ring multiplication may be absent, so we do not take it as predefined. Instead, we search candidate binary operators from the negated goal. Often multiplication 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. Distributivity is the only necessary property to apply the procedure, but associativity, commutativity, and the unit element can also be used in simplifications.

3.2 Term Generalization

Term generalization transforms Skolem constants or complex terms into variables and then performs an induction on the variables. We propose a straightforward heuristic: For each nonnumeral subterm s of type $\mathbb{Z}$ occurring in the negated goal, generalize s if s stays variable-free even after recursively applying this heuristic on the proper subterms of s itself. For example, in the following variable-free integer terms, the underlined subterms would be generalized:

$$\underline{\mathsf{a}} \qquad 123 \qquad \underline{\mathsf{f}\ 0\ 2} \qquad 2\mathsf{f}\ (\underline{\mathsf{g}\ (-1)})\ (-3\underline{\mathsf{a}}) \qquad \mathsf{f}\ 1\ (\mathsf{g}\ (\underline{\mathsf{a}} + 1)) \qquad \mathsf{f}\ (\mathsf{g}\ \underline{\mathsf{a}})\ 7\underline{\mathsf{a}}$$

Let $\vec{s} = (s_1, \ldots, s_d)$ be the subterms chosen for generalization. Then, based on the negated goal $C' \vee t[\vec{s}] \neq 0$ (as in the SUMMATION rule), generalization sets up the goal $\forall \vec{n} \in N.\ t[\vec{n}] = 0$ where $N \subseteq \mathbb{Z}^d$ collects the bounds of $\vec{s}$ (often $N = \mathbb{N}^d$). We try to prove this goal up to base cases and other mild conditions.

The generalization makes it possible to use induction to prove that the goal sequence term $t[\vec{n}]$—a function of $\vec{n}$—equals zero on N. We try to prove the generalized goal assuming $\neg C'$ and some extra conditions E such as the base cases of the induction. Then, instantiating $\vec{n} := \vec{s}$, we conclude $C' \vee \neg E \vee t[\vec{s}] = 0$. This, together with the negated goal $C' \vee t[\vec{s}] \neq 0$, 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 $\vec{s} = \mathsf{f}\ s'$ and $\vec{n} = n$ are singletons and that the generalized goal contains the factorial $n!$. Its recurrence must be in a conditional clause—e.g., $(m+1)! = (m+1)m! \vee 0 \nleq m$. To use this recurrence 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 $\vec{s}$.

3.3 Form of sequence terms

Sequence terms are terms of the underlying higher-order logic that our procedure can work with. From their structure, we distinguish (pointwise) addition and multiplication, summation, and affine substitution. This gives a first-order grammar to express the sequence terms.

Definition 2. *Sequence terms on a ring A are inductively defined as follows. The logic's terms of type A with distinguished integer variables $\vec{n}$ are sequence terms. If $f_{\vec{n}}$ and $g_{\vec{n}}$ are sequence terms with d variables $\vec{n}$, then so are $f_{\vec{n}} + g_{\vec{n}}$, $f_{\vec{n}} \cdot g_{\vec{n}}$, $\sum_{i=0}^{\vec{c} \bullet \vec{n} + a} f_{\{n_j \mapsto i\}\vec{n}}$, and $\sigma f_{\vec{n}} = f_{\sigma \vec{n}}$ where $\vec{c}$ is a vector, a is an integer, $\vec{c} \bullet \vec{n} = c_1 n_1 + \cdots + c_d n_d$, and σ is an affine substitution (meaning $\sigma \vec{m} = q\vec{m} + \vec{b}$ for a matrix q and a vector $\vec{b}$); a, the entries of $\vec{c}$, and the entries of σ (meaning the entries of q and $\vec{b}$) must be numerals.*

Remark 3. In Definition 2 and in the sequel, a commutative group can be used instead of a ring if ring multiplication is absent. In this case, all formulas involving ring multiplication (e.g., $f_{\vec{n}} \cdot g_{\vec{n}}$) should be ignored.

We view sequence terms as functions $\mathbb{Z}^d \to A$, so in particular natural number indices must be extended. For example, the factorial $n!$ for $n \in \mathbb{N}$ can be extended as 0 onto the negative integers by replacing its recurrence $n! - n(n-1)! = 0$ with $n \cdot n! - n^2(n-1)! = 0$ and adding the initial value $(-1)! = 0$. This multiplication by n is necessary to make a cut at 0 and avoid the contradiction $1 = 0! = 0 \cdot (-1)! = 0$. We then write the sequence terms from the definition compactly as $f+g$, $f \cdot g$, $\sum_j^a f$, and σf, and call $a = \vec{c} \bullet \vec{n} + d$ an affine variable sum. Moreover, since $\cdot$, $\sum_j^a$, and σ all distribute over $+$, we can write any sequence term as $c_1 f^1 + \cdots + c_k f^k$ where the coefficients c_j are numerals and the sequence terms f^j are distinct and do not contain $+$. Finally, we forbid variable shadowing: $\sum_{n_j=0}^a$ binds n_j, and while $\sum_j^a \sum_j^b g$ and $\sum_j^{n_j} g$ and other references to n_j outside $\sum_j^a$ are syntactically valid, we avoid such forms by renaming them during encoding and never reintroducing them.

3.4 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

$$(n^2+1)f_{n+2,m+1} + m f_{n+1,m} = nm f_{n,m+1} - 2h_{n,m} + (m-n)h_{n,m+1} + 1 \quad (1)$$

Here, the sequences $f, h, 1$ 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.

The general form is $0 = P_1 g^1 + \cdots + P_k g^k = \vec{P} \bullet \vec{g}$ where $\vec{g} = (g^1, ..., g^k)$ is a tuple of sequence terms and $\vec{P}$ is a tuple of operator polynomials as defined below. If $k = 1$, we have a homogeneous recurrence of g^1; otherwise, it is heterogeneous.

Definition 4. *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 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: $(S_j f)_{\vec{n}} = f_{\{n_j \mapsto n_j+1\}\vec{n}}$.

With d index variables, the operator polynomials look like ordinary polynomials $\mathbb{Z}[M_1, ..., M_d, S_1, ..., S_d]$, but the composition product is noncommutative since $S_i M_i = M_i S_i + S_i$ for all $i = 1, \ldots, d$ (a derivation of which is given in the next section directly above equation (2)). As an example of expressing recurrences in terms of operator polynomials, consider the previous archetypical recurrence (1). Taking n as the first and m as the second variable, the recurrence reads

$$((M_1^2+1)S_1^2 S_2 + M_2 S_1 - M_1 M_2 S_2) \cdot f + (2 - (M_2 - M_1)S_2) \cdot h + (-1) \cdot 1 = 0$$

Remark 5. The expression $\vec{P} \bullet \vec{g}$ identifying a recurrence is itself a sequence term. It suffices to observe that if f is a sequence term, then so are the substitution $S_j f$ and the product $M_j f = (\vec{n} \mapsto n_j) \cdot f$ with the projection sequence term $\vec{n} \mapsto n_j$. If we are not in a ring setting, the integer-group product is still a bihomomorphism, which is all that we need in the sequel.

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 nonempty 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.

In other contexts, an analogous step is known as lemma filtering or premise selection [5, Sect. 2]. Clutter 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 clutter.

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.
- We could attempt to prove them when the initial recurrences are selected.

In our ongoing implementation, we chose the first option, but what the best option is remains an open question.

4 Propagation

Holonomic sequences can be defined by homogeneous recurrences with polynomial coefficients and finitely many base cases. They are closed under the four operations that build sequence terms $(+, \cdot, \sum_j^a, \sigma)$, which especially makes their equality decidable [28]. The closure is realized by four procedures to derive recurrences of a sequence term from the recurrences of its immediate subterms, which we call *propagation*. We can propagate independently of the base cases and hence work on nonholonomic sequence terms [10]. Although we expect the holonomic subcase to be decidable in our setting, in general decidable equality is lost. Additionally, unlike in the holonomic setting, we allow heterogeneous recurrences. We will build this into our noncommutative Gröbner basis setup that is used in the propagation procedures.

4.1 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 6. The system $\{ab^2 = a + b,\, a^2b = a + 1\}$ does not prove its consequence $a^2 = b$ by rewriting. (We can see that $a^2 = b$ is a consequence by multiplying the first equation by a and the second equation by b and then by subtracting the two equations.) 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 $a^2b \xrightarrow[a^2=b]{} b^2 \xrightarrow[b^2=a+1]{} a + 1$.

A theory of Gröbner bases exists for various polynomial algebras [16]. In our setting, 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 operator polynomials of Definition 4 fall into this category with the natural choice of taking all multiplier and shift operators as indeterminates. Indeed, for any sequence term f, we have the noncommutation relations

$$(S_j M_j f)_{\vec{n}} = (S_j(\vec{n} \mapsto n_j f_{\vec{n}}))_{\vec{n}} = (n_j + 1)(S_j f)_{\vec{n}} = ((M_j S_j + S_j)f)_{\vec{n}}$$

and all other pairs of multipliers and shifts commute exactly. That is:

$$S_i M_j = M_j S_i + \delta_{i,j} S_i \qquad S_i S_j = S_j S_i \qquad M_i M_j = M_j M_i \tag{2}$$

for all i and j, where $\delta_{i,j}$ denotes the Kronecker delta, which equals 1 if $i = j$ and 0 otherwise. When we consider a formal polynomial algebra (necessary to perform Gröbner basis computations), we will usually mean polynomials with integer coefficients and indeterminates $M_1, M_2, \ldots, S_1, S_2, \ldots$ satisfying (2). Exceptionally, when we use propagation to substitution, we will consider compositions of shifts formally as further individual indeterminates, as explained in Sect. 4.3. Apart from this exceptional setting, we fix a choice of monomials as follows.

Definition 7. In our setting, a *monomial* is a polynomial of the form $M_1^{x_1} \cdots M_d^{x_d} S_1^{y_1} \cdots S_d^{y_d}$ where the exponents $x_j, y_j \in \mathbb{N}$ are numerals.

Due to the (non)commutation relations (2), polynomials 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 product, as illustrated by $S_1 \cdot M_1 = M_1 S_1 + S_1$. This complicates the definition of monomial order below, which in turn defines how to interpret a polynomial equation as a rewrite rule.

Definition 8. A *monomial order* $\preccurlyeq$ is a well-founded total order on monomials such that for all monomials A, B, C, if $A \preccurlyeq B$, then the leading monomial of CA is $\preccurlyeq$-smaller than the leading monomial of CB; here, the *leading monomial* of a nonzero polynomial P means the $\preccurlyeq$-largest monomial occurring in P. (Compare this with the term order in superposition, which is well founded and compatible with contexts.)

Buchberger's algorithm to compute Gröbner bases (also in a noncommutative context) is similar to saturation-based theorem proving. It repeatedly derives from polynomial equations $P = 0$ and $R = 0$ new equations $AP - BR = 0$ where coefficient–monomial products A, B make the leading monomials of AP and BR cancel. It suffices to take A, B with smallest total degree and coprime coefficient. A and B play a similar role to the most general unifier in superposition. Since S_j is semantically bijective, we can and always do cancel it, replacing $S_j R = 0$ by $R = 0$. Unlike many saturation procedures, this modified completion into a Gröbner basis always terminates. The standard termination proof reduces to applying noetherianity of commutative polynomials over $\mathbb{Z}$ or Dickson's lemma [16].

The similarity with superposition eases presentation and implementation of Gröbner bases in the context of superposition provers. Hence, we will present propagations as applications of elimination by Gröbner bases. Other methods can be an order of magnitude faster on complicated propagations [18]; however, we do not know what would work best in practice on average cases in a theorem prover.

A single operator polynomial P_1 perfectly encodes a linear homogeneous recurrence $0 = P_1 g$ of a 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. We can encode this by a single operator polynomial for the duration of one Gröbner basis computation as follows. Without loss of generality, let $\vec{f}$ enumerate exactly once all the sequence terms needed to express the current recurrences with the help of operator polynomials. Let $\vec{f}$ depend on d variables. For each f^j, we consider a shift $F_j := S_{d+j}$ w.r.t. a so far unused variable. Then the operator polynomial $\vec{P} \bullet \vec{F}$ encodes $0 = \vec{P} \bullet \vec{f}$.

This encoding does not respect the semantics of operator polynomials; to recover it, we must apply the substitution $\{\vec{F} \mapsto \vec{f}\}$. However, products such as $F_1 F_2$ remain uninterpretable even with ring-valued sequences because the operator product—function composition—is different from multiplication of f^j's. Hence, we will simply discard uninterpretable polynomials after the Gröbner basis computation. Moreover, from now on, we will freely write f^j for F_j.

Definition 9. Let $X_1, \ldots, X_n$ be an enumeration of all multiplier and shift indeterminates. An $(X_1, \ldots, X_k)$-*elimination order* is a monomial order such that $X_j \succ X_{k+1}^{a_{k+1}} \cdots X_n^{a_n}$ for all indices $j \leq k$ and all exponents $a_{k+1}, \ldots, a_n \in \mathbb{N}$.

Our default choice for the elimination order is to first compare the total degree over $X_1, \ldots, X_k$, then compare the total degree over $X_{k+1}, \ldots, X_n$, and then break ties by comparing the n-tuples of exponents lexicographically, considering smaller exponents as larger [11, Chapter 2 §2].

Procedure 10. *Eliminating* indeterminates $X_1, \dots, X_k$ from a finite system of equations E means computing a Gröbner basis G of E w.r.t. an $(X_1, \dots, X_k)$-elimination order and then discarding all polynomials from G that contain any of $X_1, \dots, X_k$ or that are not linear in the indeterminates encoding sequence terms. (As mentioned above, during the Gröbner basis computation, whenever we derive a polynomial $S_j R$, we replace it by R.)

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). The nonlinear polynomials can be discarded as soon as they are derived during the Gröbner basis computation instead of only at the end.

Recurrence equations produced by elimination are logical consequences of the input equations. To see this, observe that $0 = \vec{P} \bullet \vec{f}$ implies $0 = A\vec{P} \bullet \vec{f}$ for every operator monomial A before the encoding. Hence, multiplication by monomials that do not contain F_j's has the correct semantics. The monomials of any encoded polynomial $\vec{P} \bullet \vec{F}$ have the same total degree in the F_j's, i.e., $\vec{P} \bullet \vec{F}$ is homogeneous in the F_j's. Now homogeneity in the F_j's is preserved by monomial multiplication and monomial canceling addition. Moreover, these operations never lower the degree of homogeneity (i.e., the total degree in the F_j's). Consequently, Buchberger's algorithm derives only polynomials homogeneous in the F_j's, and linear interpretable ones are never derived from the nonlinear uninterpretable ones. This proves that a Gröbner basis G computed by Buchberger's algorithm has the desired property that polynomials linear in the F_j's, when interpreted as equations, are logical consequences of the input equations. Then every other Gröbner basis has this desired property as well because G rewrites their elements to 0 and rewriting preserves homogeneity in the F_j's.

Despite the formally equivalent roles of all sequence terms f^i in the recurrence $0 = \vec{P} \bullet \vec{f}$, we associate with every recurrence a sequence term f^j. It is often convenient to write such a recurrence of f^j as $P_j f^j + e = 0$ where the *excess terms* $e = \vec{P} \bullet \vec{f} - P_j f^j$ contain all sequence terms f^i except f^j. The choice of f^j among $\vec{f}$ will be determined by the definition of excess terms (Definition 20). However, this choice remains irrelevant for the individual propagation steps, described below. We adapt these steps from the four closure properties of holonomic sequences by carrying excess terms along.

4.2 Propagation to Addition

Procedure 11. 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 10, this involves computing a Gröbner basis for these equations and then discarding the equations containing f or g as well as the corresponding nonlinear terms.)

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. Such propagations could also be done by iterating more primitive propagations.

Example 12. 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 $S_1^2 g = g + S_1 a + S_1^2 a$. The defining recurrence of the sum $f_n := \sum_{j=0}^{n} a_j$ is $S_1 f = f + S_1 a$. 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 10) and the total-degree-based (f, g)-elimination order with $f \prec g$. Leading monomials are shown in bold:

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

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 a proof 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 (f and h) as $\sum_{j=0}^{0} a_j \neq g_0 + a_0 \vee \sum_{j=0}^{1} a_j \neq g_1 + a_0$.

4.3 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 remain 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. This amounts to pushing all indeterminates X leftwards. The relevant formula is

$$P = \sum_{k=1}^{T} c_k \left(\prod_{j=1}^{t_k} X_{kj} \right) f_k \implies P' = \sum_{k=1}^{T} c_k \left(\prod_{j=1}^{t_k} P'_{X_{kj}} \right) f_k$$

For multipliers, we have $\sigma(M_1,\dots,M_d) = (a(M_1,\dots,M_D)+\vec{b})\sigma$. In contrast, shifts are easily pushed only rightwards—namely, $S_j\sigma = \sigma S_1^{a_{1j}}\cdots S_d^{a_{dj}}$. This holds because, using the notation $g|\vec{m} = g_{\vec{m}}$, we have

$$\begin{aligned}S_j\sigma f|\vec{n} &= \sigma f|\{n_j \mapsto n_j+1\}\vec{n} = f|(a\{n_j\mapsto n_j+1\}\vec{n}+\vec{b}) = \\ &= f|((a\vec{n})_k + a_{kj} + \vec{b}_k)_{k=1}^d = f|((a\vec{n}+\vec{b})_k + a_{kj})_{k=1}^d = \\ &= S_1^{a_{1j}}\cdots S_d^{a_{dj}} f|(a\vec{n}+\vec{b}) = \sigma S_1^{a_{1j}}\cdots S_d^{a_{dj}} f|\vec{n}\end{aligned}$$

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

$$\mathbb{S}_j M_k = (M_k + a_{kj})\mathbb{S}_j \qquad \mathbb{S}_i\mathbb{S}_j = \mathbb{S}_j\mathbb{S}_i \qquad \mathbb{S}_i S_j = S_j\mathbb{S}_i \tag{3}$$

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 $\mathbb{S}_1,\mathbb{S}_2,\dots$ satisfying the relations (3), while also keeping (2). We note that, as operators, the indeterminates further satisfy (essentially by definition) the relations

$$\mathbb{S}_j \prod_{k\colon a_{kj}<0} S_k^{|a_{kj}|} = \prod_{k\colon a_{kj}>0} S_k^{a_{kj}} \quad \text{for } j\in\{1,\dots,D\} \tag{4}$$

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 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 7 to mean any polynomial of the form $M_1^{x_1}\cdots M_d^{x_d}S_1^{y_1}\cdots S_d^{y_d}\mathbb{S}_1^{z_1}\cdots\mathbb{S}_D^{z_D}$ where the exponents $x_j, y_j, z_j \in \mathbb{N}$ are numerals.

Procedure 13. 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 (4). 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.

All shifts in σe, including composite ones, are interpreted as substitutions.

Example 14. 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 $\binom{\cdot}{\cdot}_{n_1,n_2} = \binom{n_1}{n_2} = \frac{n_1!}{n_2!(n_1-n_2)!}$, 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 $((M_2+1)S_1S_2 - M_1 - 1)\binom{\cdot}{\cdot} = 0$. (These recurrences could be propagated from the recurrence of the factorial, except that our procedure cannot handle the domain extension.) We want to propagate these recurrences 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 (4) read

$$(\mathbb{S}_1 S_2 - S_1)\binom{\cdot}{\cdot} = 0 \quad \text{(i)} \qquad\qquad (\mathbb{S}_2 - S_2)\binom{\cdot}{\cdot} = 0 \quad \text{(ii)}$$

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:

$$(-S_1 + \mathbb{S}_1\mathbb{S}_2)\binom{\cdot}{\cdot} = 0 \qquad\qquad (-S_2 + \mathbb{S}_2)\binom{\cdot}{\cdot} = 0$$
$$(\mathbb{S}_1\mathbb{S}_2^2 - \mathbb{S}_2 - 1)\binom{\cdot}{\cdot} = 0 \qquad\qquad ((M_2+1)\mathbb{S}_2 - M_1 + M_2)\binom{\cdot}{\cdot} = 0$$
$$((M_1+1)\mathbb{S}_1\mathbb{S}_2 - (M_1 - M_2 + 2)\mathbb{S}_1 - M_1 - 1)\binom{\cdot}{\cdot} = 0$$
$$((M_1 - M_2 + 1)(M_1 - M_2 + 2)\mathbb{S}_1 - M_1M_2 - M_2)\binom{\cdot}{\cdot} = 0$$

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

$$(S_1S_2^2 - S_2 - 1)f = 0 \qquad\qquad ((M_2 - M_1 + 1)S_2 - 2M_1 + M_2)f = 0$$
$$((M_1+1)S_1S_2 - (2M_1 - M_2 + 2)S_1 - M_1 - 1)f = 0$$
$$((2M_1 - M_2 + 1)(2M_1 - M_2 + 2)S_1 - (M_1+1)(M_2 - M_1))f = 0$$

The next step is to propagate to the summation. We postpone it to Example 18, after we have presented the required theory.

4.4 Propagation to Product

Let $\cdot$ be ring multiplication or more generally a group bihomomorphism. If the sequence terms f and g depend on disjoint sets of variables, recurrences of $fg = f \cdot g$ 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 15. Let f and g be sequence terms parameterized by the variables $\vec{n} = (n_j)_{j=1}^d$. Let $\vec{m} = (n_{j+d})_{j=1}^d$ 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 the union of the recurrences of $f_{\vec{n}}$ multiplied on the right by $g_{\vec{m}}$ and of those of $g_{\vec{m}}$ multiplied on the left by $f_{\vec{n}}$. 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 using Procedure 13.

4.5 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 $\sum_{n_1=0}^{n_2} Pf_{\vec{n}} + \sum_{n_1=0}^{n_2} e_{\vec{n}} = 0$ 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 16. 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 $\sum_{n_1=0}^{n_2} Pf_{\vec{n}} + \sum_{n_1=0}^{n_2} e_{\vec{n}} = 0$. 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 some 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

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

Remark 17. The summation $\sum_{n_1=0}^{n_2} f_{\vec{n}}$ is defined if $f_{\vec{n}}$ is defined when $0 \leq n_1 \leq n_2$. This does not imply that $\sum_{n_1=0}^{n_2} Pf_{\vec{n}}$ is defined because P could shift n_1 more than n_2. Consequently, one might have to consider for example $\sum_{n_1=0}^{n_2-2} Pf_{\vec{n}}$ instead and add $f(n_2, n_2 - 1, \ldots) + f(n_2, n_2, \ldots)$ retroactively. This is awkward; we assume instead that sequence terms are globally defined. A similar issue affects propagation to substitution.

Example 18. 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 14. 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:

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

Then we push the remaining shifts S_2 leftwards:

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

Hence, in total we have

$$\begin{array}{lllll}\multicolumn{5}{l}{\displaystyle\sum_{n_1=0}^{n_2}(S_1S_2^2-S_2-1)f-(S_2^2-S_2-1)\sum_{n_1=0}^{n_2}f}\\ =\{n_1\mapsto n_2\}S_1S_2^2f & -\{n_1\mapsto 0\}S_2^2f & -S_2^2\{n_1\mapsto n_2\}f & -S_2\{n_1\mapsto n_2\}(S_2-1)f & \\ =\binom{n_2+1}{1} & -\binom{0}{n_2+2} & -\binom{n_2+2}{0} & -\binom{n_2+1}{1}+\binom{n_2+1}{0} & \\ =\cancel{\binom{n_2+1}{1}} & -0 & -1 & -\cancel{\binom{n_2+1}{1}}+1 & =0\end{array}$$

Since $(S_1S_2^2-S_2-1)f=0$, 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.

We can derive closed formulas for the summation propagation result after elimination of multiplication M_1 by the sum's index variable. This can help understand the form of the result while the recursive formulas of Procedure 16 remain a viable way to compute it. The telescoping formula amounts to Euclidean division of polynomials by S_1-1 (where S_1 commutes with remaining indeterminates because M_1 is eliminated). Namely, if $P(S_1)=(S_1-1)Q(S_1)+P(1)$, then $P(S_1)f_{\vec{n}}+e_{\vec{n}}=0$ implies

$$\begin{aligned}-\sum_{n_1=0}^{n_2}e_{\vec{n}}=\sum_{n_1=0}^{n_2}P(S_1)f_{\vec{n}}&=\sum_{n_1=0}^{n_2}((S_1-1)Q(S_1)f_{\vec{n}}+P(1)f_{\vec{n}})\\ &=\{n_1\mapsto n_2+1\}Q(S_1)f_{\vec{n}}-\{n_1\mapsto 0\}Q(S_1)f_{\vec{n}}+\sum_{n_1=0}^{n_2}P(1)f_{\vec{n}}.\end{aligned}$$

Here, all terms except the last summation can be considered as excess to be eliminated later. The operator polynomial $P(1)$ now contains neither M_1 nor S_1.

Concerning the second identity, let $\sigma=\{n_1\mapsto n_2\}$. Then, for example

$$\begin{aligned}\sum_{n_1=0}^{n_2}S_2^3f_{\vec{n}}&=S_2\sum_{n_1=0}^{n_2}S_2^2f_{\vec{n}}-S_2\sigma S_2^2f_{\vec{n}}\\ &=S_2^2\sum_{n_1=0}^{n_2}S_2f_{\vec{n}}-S_2^2\sigma S_2f_{\vec{n}}-S_2\sigma S_2^2f_{\vec{n}}\\ &=S_2^3\sum_{n_1=0}^{n_2}f_{\vec{n}}-S_2(S_2^2\sigma+S_2\sigma S_2+\sigma S_2^2)f_{\vec{n}}\end{aligned}$$

and in general $\sum_{n_1=0}^{n_2} S_2^a f_{\vec{n}} - S_2^a \sum_{n_1=0}^{n_2} f_{\vec{n}} = -S_2 \sum_{j=0}^{a-1} S_2^j \sigma S_2^{a-1-j} f_{\vec{n}}$. Since $S_2\sigma = \sigma S_1 S_2$, we can rewrite this as

$$S_2 \sum_{j=0}^{a-1} S_2^j \sigma S_2^{a-1-j} = \sum_{j=0}^{a-1} \sigma S_1^{j+1} S_2^a = \sigma \frac{S_1^a - 1}{S_1 - 1} S_1 S_2^a = \sigma \frac{(S_1 S_2)^a - S_2^a}{S_1 - 1} S_1.$$

So by linearity, for $P(S_2)$ containing neither M_1 nor S_1 we have

$$\sum_{n_1=0}^{n_2} P(S_2) f_{\vec{n}} - P(S_2) \sum_{n_1=0}^{n_2} f_{\vec{n}} = -\sigma \frac{P(S_1 S_2) - P(S_2)}{S_1 - 1} S_1 f_{\vec{n}}.$$

Combining the previous formulas, we can write propagation of a recurrence $0 = P(S_1, S_2) f_{\vec{n}} + e_{\vec{n}}$ in the form

$$\begin{aligned}
0 = P(1, S_2) \sum_{n_1=0}^{n_2} f_{\vec{n}} &+ \{n_1 \mapsto n_2\} \frac{P(S_1, S_2) - P(1, S_2)}{S_1 - 1} S_1 f_{\vec{n}} \\
&- \{n_1 \mapsto 0\ \} \frac{P(S_1, S_2) - P(1, S_2)}{S_1 - 1} f_{\vec{n}} \\
&- \{n_1 \mapsto n_2\} \frac{P(1, S_1 S_2) - P(1, S_2)}{S_1 - 1} S_1 f_{\vec{n}} \\
&+ \sum_{n_1=0}^{n_2} e_{\vec{n}} \\
= P(1, S_2) \sum_{n_1=0}^{n_2} f_{\vec{n}} &+ \{n_1 \mapsto n_2\} \frac{P(S_1, S_2) - P(1, S_1 S_2)}{S_1 - 1} S_1 f_{\vec{n}} \\
&- \{n_1 \mapsto 0\ \} \frac{P(S_1, S_2) - P(1, S_2)}{S_1 - 1} f_{\vec{n}} + \sum_{n_1=0}^{n_2} e_{\vec{n}}
\end{aligned}$$

4.6 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 holonomic case, e will be syntactically 0. We have also observed that e is often 0 in the nonholonomic case as well. But if 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. This process will always terminate, although it might fail to find recurrences. 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 proper 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 $\{n_1 \mapsto 2n_1\}M_1\{n_1 \mapsto 3n_1{+}1\}f \rightarrow 2M_1\{n_1 \mapsto 6n_1{+}2\}f$.

Definition 19. The *spine* of a sequence term f without addition, denoted by spine f, is the sequence term obtained intuitively by erasing operator polynomials from f. Precisely, this means fully reducing f by the rewrite rules

$$at \rightarrow t \qquad M_j t \rightarrow t \qquad \{\vec{n} \mapsto b\vec{n} + \vec{c}\}t \rightarrow \{\vec{n} \mapsto b\vec{n}\}t \qquad \textstyle\sum_j^{\vec{c}\bullet\vec{n}+a} t \rightarrow \sum_j^{\vec{c}\bullet\vec{n}} t$$

where a, $\vec{c}$, and the matrix b are all numeric.

As substitutions, shift indeterminates mix with other substitutions, which explains the last two rules. For example, spine $\{n_1 \mapsto 2n_1\}M_1\{n_1 \mapsto 3n_1 + 1\}f = \{n_1 \mapsto 2n_1\}\{n_1 \mapsto 3n_1\}f$. If we have a sequence term $c_1 g^1 + \cdots + c_k g^k$ with addition, it contains multiple spines, one for each g^j. The significance of spines is that when we derive a more complex consequence from a recurrence (during elimination by applying an operator polynomial to it), its spines do not become more complex.

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 (spine f)e_g and e_f(spine g), and propagation to the summation $\sum_{n_1=0}^{n_2} f$ produces $\{n_1 \mapsto 0\}$ (spine f), $\{n_1 \mapsto n_2\}$(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. This is an invariant we want to impose. Given how spines change under propagation, it suffices to additionally require that the term order satisfies $\sum_j^a t > \sigma t$ 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 20. Fix a Knuth–Bendix order with argument coefficients [19] with exactly three weights $W\sum_{n=0}^{a} > W(\cdot) > 3W\sigma > 0$ and all argument coefficients set to 2. Moreover, projection sequence terms corresponding to M_j's (Remark 5) must have equal weights, and substitutions with fewer bindings must have lower precedence than those with more bindings. The *excess* (*partial*) *order* on addition-free 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 the leaf nodes of the term tree and pulls summations towards the root. The resulting normal form is simply the typical way of writing terms without explicit substitutions. 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:

$$\begin{array}{rlrl}
s\cdot\sum_j^a t &\to \sum_j^a st & 1t, t1, \{\}t &\to t\\
(\sum_j^a s)\cdot t &\to \sum_j^a st & (\sigma\cup\{n_j\mapsto a\})u &\to \sigma u\\
\sigma\sum_j^a t &\to \sum_j^{\sigma a}\sigma t & (\sigma\cup\{n_j\mapsto a\})M_j &\to a\\
\sum_j^c t &\to \{n_j\mapsto 0\}t+\dots+\{n_j\mapsto c\}t & \sigma(ts) &\to \sigma t\cdot\sigma s\\
\sum_j^{-c} t &\to -\{n_j\mapsto -1\}t-\dots-\{n_j\mapsto 1-c\}t & \sigma\sigma' t &\to (\sigma\circ\sigma')t
\end{array}$$

where s, t, u are sequence terms, u does not contain the variable n_j, a is an affine variable sum, $M_j = \vec{n}\mapsto n_j$ is a projection sequence term, σ, σ' are affine substitutions, and the numeral c is nonnegative.

These rules produce additions, which must be interpreted as follows. For any rule above of the general form $t_0 \to c_1t_1+\dots+c_kt_k$, the actual rewrite on the level of entire recurrences is $f[t_0]+R = 0 \ \to\ c_1f[t_1]+\dots+c_kf[t_k]+R = 0$ where c_j are numerals, the sequence terms $f[t_j]$ are equal except for the 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 $t_1, \dots,$ t_k individually. Let us proceed down the right column. The first rule triple ($1t, t1,$ $\{\}t \to t$) is trivial. In the second rule, by assumption removing a binding from a substitution $\sigma\cup\{n_j\mapsto a\}$ lowers its precedence. The third rule needs only $(\sigma\cup\{n_j\mapsto a\})M_j > M_k$ for all j, k, which holds because M_j and M_k have the same weight. For the fourth rule, the only duplicating rule $\sigma(ts)\to\sigma t\cdot\sigma s$ requires $W\sigma+2(W(\cdot)+2(Wt+Ws)) > W(\cdot)+4W\sigma+4Wt+4Ws$, which is equivalent to the assumption $W(\cdot) > 3W\sigma$. For the last rule, combining substitutions σ and σ' into one $\sigma\circ\sigma'$ decreases the weight. The rules on the left column are similarly oriented by the assumed weight order. While comparing spines first, these orientations remain, although they might be nonstrict.

4.7 The Question of Completeness in the Holonomic Case

We do not know if our procedure is complete in the holonomic case — that is, if it propagates enough recurrences to reduce deciding equality to a finite number of base cases for any holonomic atomic sequence terms. Since our method is closely related to the holonomic decision procedure, one could have expected that proving completeness in our setting would be easy to achieve. However, there are several issues that prevented us from proving completeness.

The main issue arises in connection with propagation to substitution. It is not related to the unusual algebra of the compound shifts, but rather to the change—especially the reduction—of the number of variables. Indeed, we do not even know if propagation to $\{n_1\mapsto 0\}$ is complete. For $\{n_1\mapsto 0\}$, the compound shifts are simply all S_j for $j\geq 2$ and the propagation procedure can be reformulated as follows: Eliminate S_1 and replace M_1 by 0 in the remaining recurrences.

Example 21. Consider the operator polynomials $\{M_1(S_1-1),\ M_1(S_2-1),\ S_1(S_2-1)\}$, which form a Gröbner basis (w.r.t. any monomial order). If a sequence f satisfies the corresponding recurrences, f is holonomic. Namely, using first $M_1(S_1-1)$ and then $S_1(S_2-1)$, we get $f_{1,0} = f_{1+n_1,0} = f_{1+n_1,n_2}$ and similarly $f_{0,0} = f_{-n_1,0} = f_{-n_1,n_2}$ for $n_1 \in \mathbb{N}$ and $n_2 \in \mathbb{Z}$. Hence f is determined by only two base cases (namely, $f_{1,0}$ and $f_{0,0}$) and is therefore holonomic.

Now consider propagation to $\{n_1 \mapsto 0\}f_{\vec{n}}$. If we discard recurrences containing S_1, we are left with $M_1(S_2-1)f = 0$. But replacing M_1 by 0 here gives the triviality $0 = 0$, leading to no recurrences for $\{n_1 \mapsto 0\}f_{\vec{n}}$. In general, the replacement is the issue that this entire subsection is about. In this example, we recover thanks to the requirement that shifts are canceled during elimination, because then $S_1(S_2-1)$ yields S_2-1, and this operator polynomial gives us a useful recurrence for $\{n_1 \mapsto 0\}f_{\vec{n}}$.

In the same way that propagation to $\{n_1 \mapsto 0\}$ eliminates S_1 and replaces M_1 by 0, propagation to $\sum_{n_1}$ eliminates M_1 and replaces S_1 by 1 (and adds excess terms). However, with $\sum_{n_1}$, if replacement would map a recurrence to 0, we can avoid the problem thanks to the identity $M_1(S_1-1)-(S_1-1)(M_1-1) = -1$. Using this, we see that an ill-behaved recurrence $(S_1-1)R = 0$ implies

$$0 = M_1(S_1-1)R = (S_1-1)(M_1-1)R - R$$

where $(S_1-1)(M_1-1)R$ will map to 0, leaving $R = 0$. Higher powers of S_1-1 are canceled by iterating this. For $\{n_1 \mapsto 0\}$, there is no similar identity $PM_1 - M_1P' = -1$; this can be seen by applying it to $\delta_{n_1,0}$ and evaluating at $n_1 = 0$.

The above standard argument works even better in the differential setting [10,18,28]. There differentiation ∂_j w.r.t. variable n_j of index j takes the place of the shift S_j. Propagation to integration $\int dn_1$ is secured by $M_1\partial_1 - \partial_1 M_1 = -1$, and to substitution $\{n_1 \mapsto 0\}$ by $-\partial_1 M_1 - M_1(-\partial_1) = -1$. In fact, the proper theory of holonomicity exists in the differential setting and merely translates to the discrete in the coefficients of the power series of functions. Then the discrete substitution $\{n_1 \mapsto -1\}$ corresponds to contour integration around 0. However, only ∂_1 cancels when integrating, whereas the discrete M_1+1 corresponds to the differential product $\partial_1 M_1$ (because $\frac{\partial}{\partial z}z(f_k z^k) = (k+1)(f_k z^k)$). In other words, a Gröbner basis of our shifty operator polynomials contains different, possibly worse, information than a Gröbner basis of differential polynomials that actually define the holonomicity.

We still expect that our procedure is complete. To support this, fix a transcendental constant q and consider the $\mathbb{Z}$-algebra spanned by shifts and exponential multipliers q^{n_j} (instead of n_j, also known as M_j). For the substitution $\{n_1 \mapsto 0\}$, we observe the same issue that $q^{n_1}-1$ is replaced by 0 and uncancellable. However, using homological algebra, Sabbah proved that substitution—with a compound shift indeterminate based method—preserves holonomicity in this algebra [21]. In the analytic limit $q \to 1$, we also have $(q^{n_j}-1)/(q-1) \to n_j$, and we obtain our operator polynomial algebra. Unfortunately, this observation tells us nothing about the computational behavior of operator polynomials and their Gröbner bases.

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, for any group, the multiplication-by-± 1 map is invertible. This 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 the leading monomials' coefficients.

A common case is that variables range over natural numbers and we have a final recurrence with leading shift $S_1^{b_1} \cdots S_d^{b_d}$ w.r.t. any monomial order. Then the values $\bigcup_{j=1}^{d} \{\vec{n} \in \mathbb{N}^d \mid n_j < b_j\}$ suffice for the base cases, as a union of stacked hyperplanes that is infinite unless $d \leq 1$, but it corresponds to only $\sum_{j=1}^{d} b_j$ one-variable substitutions $\{n_j \mapsto a\}$ for $1 \leq j \leq d$ 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. If recurrences have different side conditions, then there might not be an optimal combination of base cases and conditions but rather many incomparable options to choose from.

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 $N(J, \vec{b}) := \{\vec{n} \in \mathbb{N}^d \mid n_j < b_j \text{ for all } j \in J\}$ where $J \subseteq \{1, \dots, d\}$ and $\vec{b}$ vary. One such stack is represented by $\prod_{j \in J} b_j$ substitutions $\{n_j \mapsto a_j \mid j \in J\}$ where the a_j's are chosen arbitrarily such that $0 \leq a_j < b_j$. 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 a forward induction on natural numbers starting from 0 by the change of variables $n \mapsto b - 1 - n$.

6 Examples

Our procedure can prove the induction step of holonomic sequence formulas such as Example 14, the binomial formula:

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

Heterogeneous recurrences, which are beyond the holonomic fragment, enable proving elementary general sequence formulas such as Example 12 and the following:

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

If we ignore the holonomic base case requirements, 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. They count the number of partitions of a k-element set into n subsets.

Example 22. As further demonstration, we apply our procedure to the last equation. 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 first n denotes the multiplier for the variable n, and the second 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:

$$\begin{aligned} K_1N_1-(n_1+1)N_1-1 \ \text{ and } \ H_1-1 \quad &\text{from } \left\{{k_1 \atop n_1}\right\} \\ (h_2-n_2+1)H_2-1,\ N_2-h_2+n_2, \ \text{ and } \ K_2-1 \quad &\text{from } (h_2-n_2)!^{-1} \end{aligned}$$

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 (4) 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

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

To sum over n, we first eliminate n from the previous two recurrences and conclude $(H(K-h) + (N-1)(KH - (h+1)H + 1))(\left\{{k \atop n}\right\}/(h-n)!) = 0$. 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:

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

Here, by the recurrence of the inverse of the factorial, we get $(-1)!^{-1} = 0$. So we obtain a recurrence $H(K-h)\sum_{n=0}^{h}\left\{{k \atop n}\right\}/(h-n)! = 0$ 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 $h^k/h! - \sum_{n=0}^{h}\left\{{k \atop n}\right\}/(h-n)!$.

The largest shift HK of the operator $H(K-h)$ determines that the two sets of base cases $h = 0$ and $k = 0$ are 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 $\left\{{k \atop n}\right\}$ simplifies to its initial value $\left\{{0 \atop n}\right\}$. With these initial values, only the term with $n = 0$ remains, while generally we might have to repeat Summation.

Example 23. Let $a, b \in \mathbb{R}$ be reals and $h \in \mathbb{N}$ be a natural number. We prove

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

For the sake of conciseness, let us use only the recurrence of $\binom{h}{n}$ that already lacks multiplication by n—namely, $(HN - N - 1)\binom{h}{n} = 0$ from Pascal's triangle. To propagate to the product $\binom{h}{n}a^n$ we rename variables n apart as n_1 and n_2, and proceed to eliminate their shifts N_1 and N_2 from

$$HN_1 - N_1 - 1 \qquad N_2 - a \qquad N_1N_2 - N$$

Here $N_1N_2 - N$ is the defining relation (4) for the compound shift N. Immediately, $N_2 - a$ rewrites $N_1N_2 - N$ to $aN_1 - N$. A superposition $a(HN_1 - N_1 - 1) - H(aN_1 - N) = HN - aN_1 - a$ leaves a conclusion that is rewritten to $HN - N - a$ by $aN_1 - N$, completing the propagation.

We find the obvious recurrences $(bN-1)b^{h-n}=0$ and $(H-b)b^{h-n}=0$ by propagation to substitution. To propagate to the final product $\binom{h}{n}a^n\cdot b^{h-n}$ we rename like above and eliminate N_1,N_2,H_1,H_2 from the operator polynomials

$$H_1N_1-N_1-a \qquad \begin{matrix} bN_2-1 \\ H_2-b \end{matrix} \qquad \begin{matrix} N_1N_2-N \\ H_1H_2-H \end{matrix}$$

The top-row polynomials give N_1-bN, which rewrites $H_1N_1-N_1-a$ to $bH_1N-bN-a$. The bottom-row polynomials give bH_1-H, and one rewrite later we conclude $(HN-bN-a)(\binom{h}{n}a^nb^{h-n})=0$.

The propagation to summation $\sum_{n=0}^{h}$ will map $N\mapsto 1$ in the recurrence of the main term, giving $H-b-a$, which also annihilates the left-hand side $(a+b)^h$. Finally, we check that the excess terms become 0:

$$\begin{aligned}0=\sum_{n=0}^{h}(HN-bN)\Big(\binom{h}{n}a^nb^{h-n}\Big)&=\sum_{n=0}^{h}(H-b)\Big(\binom{h}{n}a^nb^{h-n}\Big)+\\&+\{n\mapsto h+1\}(H-b)\Big(\binom{h}{n}a^nb^{h-n}\Big)-\{n\mapsto 0\}(H-b)\Big(\binom{h}{n}a^nb^{h-n}\Big)=\\&=\sum_{n=0}^{h}(H-b)\Big(\binom{h}{n}a^nb^{h-n}\Big)+\binom{h+1}{h+1}a^{h+1}-0-\binom{h+1}{0}b^{h+1}+b\binom{h}{0}b^h=\\&=(H-b)\sum_{n=0}^{h}\binom{h}{n}a^nb^{h-n}-\binom{h+1}{h+1}a^{h+1}+a^{h+1}=(H-b)\sum_{n=0}^{h}\binom{h}{n}a^nb^{h-n}\end{aligned}$$

Hence the operator polynomial $H-b-a$ annihilates (meaning maps to 0) the entire difference $(a+b)^h-\sum_{n=0}^{h}\binom{h}{n}a^nb^{h-n}$. For induction, the leading shift monomial of $H-b-a$ is H. Hence an induction step of size 1 w.r.t. h is guaranteed to succeed. The base cases are obtained by substituting $\{h\mapsto 0\}$, and a theorem prover can check them by direct simplification.

Example 24. We prove Vandermonde's convolution formula

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

The textbook proof is to expand the three binomial powers in $(x+1)^{a+b}=(x+1)^a(x+1)^b$ and then match the like powers of x. Since both sides of the summation formula are polynomials in a and b, it must hold for a and b taken from any commutative ring. Our recurrence propagation procedure can directly work in this full generality by only considering recurrences in n and h.

We start from $((n+1)N+n-a)\binom{a}{n}=0$. This propagates to $\binom{b}{h-n}=\{a\mapsto b, n\mapsto h-n\}\binom{a}{n}$ as $((h-n+1)H+h-n-b)\binom{b}{h-n}=0$ and $(HN-1)\binom{b}{h-n}=0$

with monomial order such that $H < N$. Next, to propagate to product we rename n as n_1 and n_2, and proceed to eliminate N_1 and N_2 from

$$(n_1+1)N_1 + n_1 - a \qquad \begin{matrix}(h-n_2+1)H + h - n_2 - b \\ HN_2 - 1\end{matrix} \qquad N_1N_2 - N$$

From $HN_2 - 1$ and $N_1N_2 - N$ we derive $N_1 - HN$ that rewrites the leftmost operator polynomial leading to the result

$$(n_1+1)HN + n_1 - a \qquad (h-n_2+1)H + h - n_2 - b$$

where we then identify n_1 and n_2. After this identification, if nH is the leading monomial of the right polynomial, it could rewrite the leading monomial nHN of the left polynomial. This shows that the final result of propagation steps need not be a Gröbner basis, which is unproblematic. Instead, we start with that rewrite when propagating to summation $\sum_{n=0}^{h}$ next. The result of the rewrite is $n - a + (h+1)H + N(h-n-b)$. Its leading monomial nN superposes with nH of $(h-n+1)H + h - n - b$ because we must eliminate n. After simplifying the result, we get

$$(H+1)((h+1)H + h - a - b) - bH(N-1)$$

which does not contain n. In the first term, $(h+1)H + h - a - b$ annihilates the left-hand side $\binom{a+b}{h}$ of the goal equation, while the second term $-bH(N-1)$ maps to 0 when N maps to 1. This implies that the final operator polynomial we get is $(H+1)((h+1)H + h - a - b)$, but we must also check that the excess terms become 0.

The excess term computation is routine but lengthy. We omit it because we know that it succeeds by the following argument. The sum $\sum_{n=0}^{h}\binom{a}{n}\binom{b}{h-n}$ has so-called natural boundaries, meaning that the summand is 0 outside the interval $[0,h]$. Hence the summation range $[0,h]$ can be extended without changing the value of the sum or the value of the excess terms that depends on the sum. However, excess terms are found by evaluation at the end points of the summation range. When the range is extended enough (by about the degree of the operator polynomial), it becomes obvious that the excess terms equal 0.

Example 25. Let f be group-valued and h be a natural number. We prove

$$\mathcal{L}_h := \sum_{n=0}^{h} f_{h-n} = \sum_{n=0}^{h} f_n =: \mathcal{R}_h$$

Let us start with the substitution on the left-hand side. If $h - n = k$, then $(h+1) - n = k + 1$ and $h - (n+1) = k - 1$, so $H = K$ and $N = K^{-1}$ define the relevant compound shifts. The latter equation, $N = K^{-1}$, translates to the polynomial $NK - 1$, which simplifies to $NH - 1$ when K is eliminated by the equation $H = K$. So we conclude $(NH - 1)f_{h-n} = 0$.

Next, we push the indeterminates leftwards to propagate to the summation $\mathcal{L}_h = \sum_{n=0}^{h} f_{h-n}$ and simplify the excess terms as we proceed:

$$0 = \sum_{n=0}^{h}(NHf_{h-n} - f_{h-n}) = \{n{\mapsto}h{+}1\}Hf_{h-n} - Hf_{h-0} + \sum_{n=0}^{h}(Hf_{h-n} - f_{h-n})$$
$$= f_0 - Hf_h - Hf_{h-h} + (H-1)\sum_{n=0}^{h} f_{h-n} = -Hf_h + (H-1)\mathcal{L}_h$$

Hence $(H-1)\mathcal{L}_h - Hf_h = 0$. Since we also have $(H-1)\mathcal{R}_h = Hf_h$, the final propagation to difference gives $(H-1)(\mathcal{R}_h - \mathcal{L}_h) = 0$. The rest of the proof is routine.

Example 26. Let f be group-valued and k be a natural number. We prove

$$\mathcal{L}_k := \sum_{h=0}^{k} \overbrace{\sum_{n=0}^{h} f_{h,n}}^{l_h} = \sum_{n=0}^{k} \overbrace{\sum_{h=n}^{k} f_{h,n}}^{r_{k,n}} =: \mathcal{R}_k$$

First consider the left-hand side $\mathcal{L}_k$. Trivially, $(K-1)l_h = 0$ with K denoting a shift w.r.t. k. This recurrence has no multiplication by h, so we directly proceed to pull it over $\sum_{h=0}^{k}$ to get a recurrence for $\mathcal{L}_k$:

$$0 = \sum_{h=0}^{k}(Kl_h - l_h) = -K\{h \mapsto k\}l_h + (K-1)\sum_{h=0}^{k} l_h = -Kl_k + (K-1)\mathcal{L}_k$$

Next, consider the right-hand side $\mathcal{R}_k$. Recall that $r_{k,n}$ can be expressed in terms of summations from 0 as $\sum_{h=0}^{k} f_{h,n} - \sum_{h=0}^{n-1} f_{h,n}$. Nevertheless, the constancy recurrence $(K-1)f_{h,n} = 0$ propagates as above, yielding $(K-1)r_{k,n} - Kf_{k,n} = 0$. To propagate from $r_{k,n}$ to $\mathcal{R}_k$, we pull the shift K over $\sum_{n=0}^{k}$ as follows:

$$\begin{aligned} 0 &= \sum_{n=0}^{k}(Kr_{k,n} - r_{k,n} - Kf_{k,n}) \\ &= -Kr_{k,k} + (K-1)\sum_{n=0}^{k} r_{k,n} + Kf_{k,k} - K\sum_{n=0}^{k} f_{k,n} \\ &= -Kr_{k,k} + (K-1)\mathcal{R}_k + Kf_{k,k} - Kl_k \end{aligned}$$

Here the excess term $-Kr_{k,k}$ simplifies and cancels $Kf_{k,k}$:

$$-Kr_{k,k} = -\sum_{h=0}^{k+1} f_{h,k+1} + \sum_{h=0}^{k+1-1} f_{h,k+1} =$$
$$= -f_{k+1,k+1} - \sum_{h=0}^{k} f_{h,k+1} + \sum_{h=0}^{k} f_{h,k+1} = -Kf_{k,k}$$

Hence, we conclude $(K-1)\mathcal{R}_k - Kl_k = 0$. Now the final propagation to difference gives $(K-1)(\mathcal{R}_k - \mathcal{L}_k) = 0$. The proof finishes routinely.

Example 27. Abel generalized the binomial formula on the limit $\frac{a}{h}, \frac{b}{h} \to \infty$ as

$$(a+b)^h = \sum_{n=0}^{h} \binom{h}{n} a(a-n)^{n-1}(b+n)^{h-n}$$

We must propagate the recurrence $(K-c)c^k = 0$ of the powers c^k to their three substitution instances $(a+b)^h$, $(a-n)^{n-1}$, and $(b+n)^{h-n}$. We will go through the last instance in detail and just state the results of the first two easier ones:

$$\left.\begin{matrix} H-a-b \\ A-B \end{matrix}\right\}(a+b)^h = 0 \qquad (AN-a+n)(a-n)^{n-1} = 0$$

For the substitution $\{c \mapsto b+n, k \mapsto h-n\}$, we put together the defining relations (4) of the composite shifts B, N, H and the initial recurrence $(K-c)c^k = 0$:

$$B-C \qquad NK-C \qquad H-K \qquad K-c$$

We must eliminate C and K, which is readily achieved by rewriting by $B-C$ and $K-c$. Since N is $K^{-1}C$, we have $Nc = (c+1)N$. Hence we derive from $(c+1)N-B$ and $H-c$ that the substitution transforms to

$$\left.\begin{matrix} (b+n+1)N-B \\ H-b-n \end{matrix}\right\}(b+n)^{h-n} = 0$$

Now consider $a(a-n)^{n-1}$. Without the factor a, the interpretation of $(a-n)^{n-1}$ for $a = n = 0$ is undefined, but with it we can define $0(0-0)^{0-1} = 0^0 = 1$ so that it is compatible with the recurrence below. Let $f_{a,n} = (a-n)^{n-1}$ and $g_{a,n} = a(a-n)^{n-1}$. Eliminating $f_{a,n}$ from $\{(AN-a+n)f_{a,n} = 0, af_{a,n} = g_{a,n}\}$ gives

$$(aAN - (a+1)(a-n))g_{a,n} = 0$$

Next, let us consider the product $a(a-n_1)^{n_1-1} \cdot (b+n_2)^{h-n_2}$ where we have renamed the variables n apart. After introducing the compound shift $N = N_1N_2$, we proceed to eliminate N_1 and N_2 from

$$aAN_1 - (a+1)(a-n_1) \qquad \begin{matrix} (b+n_2+1)N_2 - B \\ H-b-n_2 \end{matrix} \qquad N_1N_2 - N$$

The operator $H-b-n_2$ is good as is and does not interfere with the rest of the computation if $H > b, n_2$ w.r.t. the monomial order. From $(b+n_2+1)N_2 - B$ and $N_1N_2 - N$ we derive $BN_1 - (b+n_2+1)N$, whose superposition with $aAN_1 - (a+1)(a-n)$ eliminates N_1. We conclude

$$\left.\begin{matrix} H-b-n \\ a(b+n+1)AN-(a+1)(a-n)B \end{matrix}\right\}a(a-n)^{n-1}(b+n)^{h-n} = 0$$

Unlike before, in this example we do need to use the plurality of the recurrences of the binomial coefficient $\binom{h_1}{n_1}$. A Gröbner basis of its annihilators is given by

$$(h_1+1-n_1)H_1-h_1-1 \qquad (n_1+1)N_1-h_1+n_1 \quad N_1H_1-N_1-1$$

To these we add recurrences of $a(a-n_2)^{n_2-1}(b+n_2)^{h_2-n_2}$ and composite shifts:

$$H_2-b-n_2 \qquad a(b+n_2+1)AN_2-(a+1)(a-n_2)B \qquad N_1N_2-N \qquad H_1H_2-H$$

We must eliminate N_1, N_2, H_1, H_2. From the leftmost "H polynomials" and H_1H_2-H we get $(h_1+1-n_1)H-(h_1+1)(b+n_2)$. Similarly, from the middle "N polynomials", we get $a(b{+}n_2{+}1)(n_1{+}1)AN-(h_1{-}n_1)(a{+}1)(a{-}n_2)B$. Finally, from $N_1H_1-N_1-1$ and all the four polynomials on the second line, we get $a(H-b-n_2-1)AN-(a+1)(a-n_2)B$. So

$$\left.\begin{array}{c}(h+1-n)H-(h+1)(b+n)\\ a(b+n+1)(n+1)AN-(h-n)(a+1)(a-n)B\\ a(H-b-n-1)AN-(a+1)(a-n)B\end{array}\right\}\binom{h}{n}a(a{-}n)^{n-1}(b{+}n)^{h-n}=0$$

Next, to propagate to summation, we must eliminate n. The first operator can be written $-(H+h+1)\boldsymbol{n}+(h+1)(H-b)$. The last operator can be written $((a+1)B-aAN)\boldsymbol{n}+aAN(H-b)-a(a+1)B$. Since $H+h+1$ and $(a+1)B-aAN$ commute, eliminating n yields at least

$$\begin{aligned}((a+1)B-aAN)(h+1)(H-b)+(H+h+1)(aAN(H-b)-a(a+1)B)=\\ =aANH(H-b)-a(a+1)BH+(a+1)B(h+1)(H-a-b)=\\ =(N-1)aAH(H{-}b)+a(a{+}1)H(A-B)+(aAH{+}(a{+}1)B(h{+}1))(H-a-b)\end{aligned}$$

Here $N-1$ vanishes in propagation, and $A-B$ and $H-a-b$ are annihilators of $(a+b)^h$, while irrelevant factors are shown in gray. We omit the excess term computation because bounds are natural as for Vandermonde's convolution formula (Example 24).

So the leading monomial from one recurrence of the entire difference can be chosen to be aAH^2. This leaves as base cases the three specializations $a=0$, $h=0$, and $h=1$. The first case also takes care of the invertibility of a in aAH^2. The cases $h=0$ and $h=1$ can be easily symbolically evaluated. When $a=0$, we must detect that $a(a-n)^{n-1}=\delta_{0,n}$ (maybe by its recurrence), which again makes the sum finite. Alternatively, we could treat a as a rational expression indeterminate and hence make aA unconditionally invertible. Both of these tricks are beyond the scope of our procedure.

7 Related Work

Holonomic sequences [28] 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. Relaxing this limitation yields approximately the homogeneous version of our propagation procedure (i.e., without excess terms), whose theory Chyzak, Kauers, and Salvy laid out [10]. Heterogeneity amounts to module Gröbner bases [8, 14, 20]. Its integration into propagations makes elementary identities about general sequences automatically provable, which may be of interest for general-purpose theorem provers.

In practice, hypergeometric sums are common holonomic sequences that have much faster algorithms available. Gosper's indefinite summation [15] can be applied to compute Wilf–Zeilberger pairs [27], which offer compact proof certificates for definite sum identities. These fast methods admit generalizations to the full holonomic setting. See Koutschan's thesis [18] for an overview.

Holonomic functions have a nearly dual theory where differential equations with polynomial coefficients play the role of recurrences. The above references present the discrete and continuous holonomic theories together, sometimes unified and generalized by the language of Ore polynomials [6, 9]. We presented only the discrete case because current higher-order provers lack the theoretical groundwork needed to bootstrap usable differential algebra.

Finding a closed form instead of only checking it for a summation is a different but related task. A common approach is to perform a recurrence solving phase after recurrence computation, as in the Mathematica package Sigma [1, 22].

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 shows the induction step of a proof by induction, leaving the base cases to the theorem prover.

As future work, we want to continue implementing the procedure in Zipperposition [25]. 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 also thank Anne Baanen, Pascal Fontaine, Mark Summerfield, and the anonymous reviewers for suggesting 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 NWO under the Vidi program (project No. 639.032.613, New Diophantine Directions).

References

1. Abramov, S.A., Bronstein, M., Petkovsek, M., Schneider, C.: On rational and hypergeometric solutions of linear ordinary difference equations in $\Pi\Sigma^*$-field extensions. J. Symb. Comput. **107**, 23–66 (2021)

2. 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)
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. Bronstein, M., sek, M.P.: On ore rings, linear operators and factorisation. ETH, Eidgenössische Technische Hochschule Zürich, Departement Informatik, Institut für Wissenschaftliches Rechnen **200**(19) (1993)
7. Brown, C.E., Smolka, G.: Analytic tableaux for simple type theory and its first-order fragment. Log. Methods Comput. Sci. **6**(2) (2010)
8. Bueso, J., Gómez-Torrecillas, J., Verschoren, A.: Gröbner bases for modules. In: Algorithmic Methods in Non-Commutative Algebra: Applications to Quantum Groups, pp. 169–202. Springer (2003)
9. Chyzak, F., Salvy, B.: Non-commutative elimination in ore algebras proves multivariate identities. Journal of Symbolic Computation **26**(2), 187–227 (1998). https://doi.org/https://doi.org/10.1006/jsco.1998.0207, https://www.sciencedirect.com/science/article/pii/S0747717198902073
10. 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)
11. 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, 4th edn. (2015)
12. Cruanes, S.: Extending Superposition with Integer Arithmetic, Structural Induction, and Beyond. Ph.D. thesis, École polytechnique (2015)
13. 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)
14. Fajardo, W., Gallego, C., Lezama, O., Reyes, A., Suárez, H., Venegas, H.: Gröbner bases of modules. In: Skew PBW Extensions: Ring and Module-theoretic Properties, Matrix and Gröbner Methods, and Applications, pp. 261–286. Springer (2020)
15. Gosper, R.W.: Decision procedure for indefinite hypergeometric summation. Proc. Natl. Acad. Sci. USA **75**(1), 40–42 (1978)
16. Kandri-Rody, A., Weispfenning, V.: Non-commutative Gröbner bases in algebras of solvable type. J. Symb. Comput. **9**(1), 1–26 (1990)
17. Knuth, D.E., Bendix, P.B.: Simple word problems in universal algebras. In: Leech, J. (ed.) Computational Problems in Abstract Algebra. pp. 263–297. Pergamon Press (1970)
18. Koutschan, C.: Advanced applications of the holonomic systems approach. ACM Comm. Comput. Algebra **43**(3/4), 119 (2009)
19. Ludwig, M., Waldmann, U.: An extension of the Knuth-Bendix ordering with LPO-like properties. In: Dershowitz, N., Voronkov, A. (eds.) LPAR 2007. LNCS, vol. 4790, pp. 348–362. Springer (2007)

20. Maletzky, A., Immler, F.: Gröbner bases of modules and Faugère's f_4 algorithm in Isabelle/HOL. CoRR **abs/1805.00304** (2018)
21. Sabbah, C.: Systèmes holonomes d'équations aux q-différences. In: D-modules and microlocal geometry (Lisbon, 1990), de Gruyter, Berlin. pp. 125–147 (1993)
22. Schneider, C.: Symbolic summation assists combinatorics. Sem. Lothar. Combin. **56**, 1–36 (2007)
23. Steen, A., Benzmüller, C.: Extensional higher-order paramodulation in Leo-III. J. Autom. Reason. **65**(6), 775–807 (2021)
24. Sutcliffe, G., Desharnais, M.: The CADE-28 automated theorem proving system competition—CASC-28. AI Communications **34**(4), 259–276 (2022)
25. Vukmirović, P., Bentkamp, A., Blanchette, J., Cruanes, S., Nummelin, V., Tourret, S.: Making higher-order superposition work. J. Autom. Reason. **66**(4), 541–564 (2022)
26. Vukmirović, P., Blanchette, J., Schulz, S.: Extending a high-performance prover to higher-order logic. In: Sankaranarayanan, S., Sharygina, N. (eds.) TACAS 2023. LNCS, vol. 13994, pp. 111–129. Springer (2023)
27. Wilf, H.S., Zeilberger, D.: Rational functions certify combinatorial identities. J. Am. Math. Soc. **3**(1), 147–158 (1990)
28. Zeilberger, D.: A holonomic systems approach to special functions identities. Journal of Computational and Applied Mathematics **32**(3), 321–368 (1990)