■ 2: Polynomial Theory

In this section we start developping the theory of polynomials. By this, we can examine where and how we want the *Theorema* system to support such a process. First we present our chosen definition of polynomials. We describe how the list of goal theorems is derived and how the proofs can be grouped. After a short description of the style used to show the steps and formulae of the proofs we go through the three proof groups. The proof discriptions are done incrementally, which means that we only discuss the steps that have not appeared in the proofs before.

■ 2.1: Definition of Polynomials

■ 2.1.1: Approaches in the Literature

While the notion of polynomials has historically emerged from polynomial functions over $\mathbb{Z}$ or $\mathbb{R}$, it has disadvantages to simply define them as such, as it is done e.g. in [Bôcher, 1907], where the author says:

"[…] By an integral rational function of *x*, or, as we shall say for brevity, a polynomial in *x*,
is meant a function of *x* determined by an expression of the form
(1) $c_1\, x^{\alpha_1} + c_2\, x^{\alpha_2} + \cdots + c_k\, x^{\alpha_k}$, […]"

Although this is an intuitive approach, problems occur when we extend this definition to other domains, such as $\mathbb{Z}_2$. There, for example, the two polynomials x and x^2 denote the same function, but we do not want to consider them the same polynomial.

Polynomials can be defined in several ways. The created domains are all isomorphic, but with different properties. One main reason for inventing new representations was that they allow more efficient computations. For example, a computer algebra system could use the distributive instead of the recursive representation of multivariate polynomials for being able to have faster access to a coefficient. Other representations have nice theoretical properties, such as the representation as equivalence classes of arithmetical terms, where associativity can be proved easily, because it is ‘inherited’ from the associativity on term level. But here computations become impossible, because we cannot work with equivalence classes of infinite size. One could introduce representatives for each equivalence class, but this would already be a different representation.

In modern algebraic textbooks often the representation as infinite sequences of coefficients with finitely many non–zero elements is used. Examples for this are [Kostrikin, 1982] or [Lüneburg, 1993]. Only as a second step they introduce the polynomials in the form they are normally written by hand. For example we can use $5\,x^3 + 2\,x^2$ by defining 2, 5, x as special coefficient sequences, such that $5\,x^3 + 2\,x^2$ is in fact a term built from these sequences with the use of the functions plus and times.

Another way to represent polynomials is to take a domain of monomials, with a monomial ordering, and to form sequences of monomials. This approach is especially useful when one aims at doing Gröbner basis computations.

Basically, the notion of polynomials is captured in a more general way: With "Polynomials" we denote the class of domains isomorphic to (one of) these representations.

■ 2.1.2: Our Definition

Each of the ways to define polynomials should be possible in the *Theorema* system. It should even be possible to use the system for proving that they are isomorphic. In this paper, we decide to use the representation as ordered tuples of monomials, because we want to use these polynomials for Gröbner basis computations in the future, and we want to be independent of the actual representation of monomials. By this, we do not have to distinguish between univariate and multivariate polynomials, since both can be represented in the same way.

The following functor is executable *Theorema* code. It takes a monomial domain M as argument and defines:
– the polynomial element predicate $\in \mathtt{P}$,
– the signature $\underset{\mathtt{P}}{\Sigma}$ of the polynomial domain,
– the constant polynomials $\underset{\mathtt{P}}{0}$ and $\underset{\mathtt{P}}{1}$, the zero and one polynomial,
– the polynomial addition, negative and multiplication: $\underset{\mathtt{P}}{+}$, $\underset{\mathtt{P}}{-}$ and $\underset{\mathtt{P}}{*}$.

The new–defined functor P takes one of the symbols ∈, Σ, 0, 1, +, −, ∗ and gives back the corresponding polynomial predicate, constant or function. Internally, this is written e.g. as P[−][…], whereas externally this appears as $\underset{\mathtt{P}}{-}$ **…** . A special case is the P[∈][…], which is not written in this 'canonical' way, as … $\underset{\mathtt{P}}{\in}$, but is written as … ∈ P for better readability. One always should be aware that this unary ∈ (epsilon) predicate from the functor is not the binary ∈ (element) from the language of set theory, because we use tuples, not sets, as underlying construct.

We will use the following functor for defining the polynomials. It is similar to the one used in [Buchberger, Kriftner, 1997], however, the main difference is that we go from monomials to polynomials in one stroke, while in the mentioned paper an intermediate functor defining ordered tuples is used. Furthermore we do not require the functor to be executable code at the present state of the system. This allows us to use labels, which are necessary to refer to the definitions during proofs, but which cannot be dealt with at the moment when using the functor to initialize a domain variable.

$$
\begin{array}{l}
\texttt{PolynomialsAsMonomialTuples[M\_] :=}\\
\texttt{Functor}\Big[\langle P\rangle,\ \langle m, n, \bar{p}, x, p, \bar{q}, r\rangle,\\
\\
(\langle\rangle \in P)\quad \langle\rangle \in P;\\
(\langle m\rangle \in P)\quad \langle m\rangle \in P :\Leftrightarrow m \in M;\\
(\in P)\quad \langle m, n, \bar{p}\rangle \in P :\Leftrightarrow m \in M \wedge \langle n, \bar{p}\rangle \in P \wedge m \underset{M}{>} n;\\
(\mathrm{not} \in P)\quad x \in P := \mathrm{False};\\
\\
\left(\underset{P}{\Sigma}\right)\quad \underset{P}{\Sigma} := \{0, 1, +, -, .\};\\
\\
\left(\underset{P}{0}\right)\quad \underset{P}{0} := \langle\rangle;\\
\left(\underset{P}{1}\right)\quad \underset{P}{1} := \left\langle \underset{M}{1}\right\rangle;\\
\\
\left(\underset{P}{0}\underset{P}{+}\right)\quad \langle\rangle \underset{P}{+} p := p;\\
\left(\underset{P}{+}\underset{P}{0}\right)\quad p \underset{P}{+} \langle\rangle := p;\\
\left(\underset{P}{+}\right)\quad \langle m, \bar{p}\rangle \underset{P}{+} \langle n, \bar{q}\rangle :=\\
\quad \Big\| m \smile \left(\langle\bar{p}\rangle \underset{P}{+} \langle n, \bar{q}\rangle\right) \Leftarrow m \underset{M}{>} n,\\
n \smile \left(\langle m, \bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle\right) \Leftarrow n \underset{M}{>} m,\\
\qquad\qquad \langle\bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle \Leftarrow m \ = \underset{M}{-} n,\\
\left(m \underset{M}{+} n\right) \smile \left(\langle\bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle\right) \Big\|;\\
\\
\left(\underset{P}{-}\underset{P}{0}\right)\quad \underset{P}{-}\langle\rangle := \langle\rangle;\\
\left(\underset{P}{-}\right)\quad \underset{P}{-}\langle m, \bar{p}\rangle := \underset{M}{-} m \smile \underset{P}{-}\langle\bar{p}\rangle;\\
\\
\left(\underset{P}{0}\underset{P}{*}\right)\quad \langle\rangle \underset{P}{*} p := \langle\rangle;\\
\left(\underset{P}{*}\underset{P}{0}\right)\quad p \underset{P}{*} \langle\rangle := \langle\rangle;\\
\left(\underset{P}{*}\right)\quad \langle m, \bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle :=\\
\Big\| r \Leftarrow \text{zero-}\underset{M}{\text{d}}\text{ivisor}[m, n],\ \left(m \underset{M}{*} n\right) \smile r \Big\|,\\
\text{where } r = \langle m\rangle \underset{P}{*} \langle\bar{q}\rangle\ \underset{P}{+}\ \langle\bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle\\
\Big]
\end{array}
$$

■ 2.1.3: Details of the Definition

Corresponding to different layers of the *Theorema* system, the definitions in this functor use different language elements:

– the language elements of our predicate logic, e.g. the logical connectives, the case construct $\|\ldots\|$, the `where` construct,

– the tuple construct $\langle\ldots\rangle$ and the $\smile$, which means prepending an element to a tuple,

– the constant $\underset{M}{1}$, the functions $\underset{M}{+}$, $\underset{M}{-}$ and $\underset{M}{*}$, the relation $\underset{M}{>}$ and the predicates $\in$ M and $\text{zero} - \underset{M}{\text{divisor}}$ from the monomial functor,

– the already defined polynomial functions, e.g. $\underset{P}{*}$ is defined using $\underset{P}{+}$.

We will now explain this functor in very detail:

The first line

```
PolynomialsAsMonomialTuples[M_] :=
```

gives the name of the functor. We did not simply choose "Polynomials", because other representations of polynomials can be also defined by functors. Later on, the functor is called by a statement like

```
UserPolynomials = PolynomialsAsMonomialTuples[UserMonomials]
```

after the user has defined a certain monomial domain `UserMonomials` . The domain variable `UserPolynomials` is assigned to the actual newly created polynomial domain.

The argument `M` is marked with an underscore "_", to let *Mathematica* know that this is a variable, not a constant. It would also be possible to customize the functor in the way that the user can provide it with the constant, function and predicate symbols, that should be used for the polynomials. In this version we simply chose one fixed set of symbols.

```
Functor[⟨P⟩, ⟨m, n, p̄, x, p, q̄, r⟩,
```

This line denotes that the output domain will locally be denoted by `P`. Then the local variables are listed, some of them sequence variables (as indicated by the overbar). When such a variable appears in a formula inside the functor, it can be interpreted as universally quantified over the formula, e.g. $\langle m \rangle \in P :\Leftrightarrow m \in M$ can be read as $\underset{m}{\forall} \; (\langle m \rangle \in P :\Leftrightarrow m \in M)$.

$$
\begin{array}{ll}
(\langle\rangle \in P) & \langle\rangle \in P; \\
(\langle m\rangle \in P) & \langle m\rangle \in P :\Leftrightarrow m \in M; \\
(\in P) & \langle m, n, \bar{p}\rangle \in P :\Leftrightarrow m \in M \wedge \langle n, \bar{p}\rangle \in P \wedge m \underset{M}{>} n; \\
(\text{not} \in P) & x \in P := \text{False};
\end{array}
$$

These four lines define the $\in$ predicate, the polynomial membership, inductively. Defining equivalences and equalities are written with a colon at the side of the definiendum (= "to be defined") . The base cases of the definition are: The empty tuple is a polynomial, and a singleton is a polynomial, if the element is a monomial. It is very important to note that we assume the monomial domain not to have a zero element, otherwise we would have to add the condition that a singleton must not be built by monomial zero. We will see later show an example of such monomials without zero, but it will also become visible that having a zero would simplify some of the monomials' properties.

The third line is the recursive part of the definition: A tuple consisting of at least two elements is a polynomial, if the first element is a monomial, the rest of the tuple is a polynomial and the first element is greater (in the monomial order) than the second one.

When the predicate is applied, it returns `True` for a tuple that falls into these cases, otherwise it returns `False` . This is accomplished by the last line, which, as executable statement, catches all the cases that do not match the above lines. The logical meaning of this last line is: Having a polynomial, we know that it has to be of one of the above forms, nothing else is a polynomial. In the polynomial prover the first three lines will be used directly, and all four of them are needed to derive the polynomial induction principle.

$$\left(\underset{P}{\Sigma}\right) \qquad \underset{P}{\Sigma} := \{0,\ 1,\ +,\ -,\ .\};$$

Here the signature of the polynomial domain is defined. This is the set of function symbols of this domain. (Constants are viewed as 0–ary functions.) Contrary to the standard *Mathematica* language, we use curly braces for sets (as in mathematical textbooks), not for lists.

$$\left(\underset{P}{0}\right) \qquad \underset{P}{0} := \langle\rangle;$$

$$\left(\underset{P}{1}\right) \qquad \underset{P}{1} := \left\langle \underset{M}{1} \right\rangle;$$

These two lines define the polynomial zero and one. Zero is defined as the empty tuple, while the one is defined by means of the monomial one, $\underset{M}{1}$, as tuple containing only this element. When the functor is executed, this $\underset{M}{1}$ is instantiated by calling the 0–ary function `M[1][]` of the supplied monomial domain. Note that so far, for efficiency reasons, we do not use these constants in this functor, but we immediately replace them by their representations. For example, in the following, we do not write $\underset{P}{0} \underset{P}{+} p := p$, but we already replace $\underset{P}{0}$ by its representation.

$$\left(\underset{P}{0}\ \underset{P}{+}\right) \qquad \langle\rangle \underset{P}{+} p := p;$$

$$\left(\underset{P}{+}\underset{P}{0}\right) \qquad p \underset{P}{+} \langle\rangle := p;$$

$$\left(\underset{P}{+}\right) \qquad \langle m,\ \bar{p}\rangle \underset{P}{+} \langle n,\ \bar{q}\rangle :=$$

$$\begin{aligned}
\Big\| m \smile \left(\langle\bar{p}\rangle \underset{P}{+} \langle n,\ \bar{q}\rangle\right) &\Longleftarrow m \underset{M}{>} n,\\
n \smile \left(\langle m,\ \bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle\right) &\Longleftarrow n \underset{M}{>} m,\\
\langle\bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle &\Longleftarrow m = \underset{M}{-}n,\\
\left(m \underset{M}{+} n\right) \smile \left(\langle\bar{p}\rangle \underset{P}{+} \langle\bar{q}\rangle\right) &\Big\|;
\end{aligned}$$

This is now the recursive definition of the polynomial plus operation. Simple cases are the ones where one of the polynomials is the empty tuple. In general, the sum of two polynomials is defined like this, using the sequence variables $\bar{p}$ and $\bar{q}$ to name the contents of the rest of the polynomal tuple: The two leading monomials `m, n` are compared regarding the monomial order $\underset{M}{>}$. Depending on this comparison, the result of the addition is defined in each case.

The double bracketing bars $\|\ldots\|$ are used to write this case distinction. Inside these bracketing bars, the different results and the corresponding conditions are listed, in the way that first a result is given, then the $\Longleftarrow$, which means "if", then the condition is given. Note that the conditions are abbreviated in the following way: Each condition denotes in fact its conjunction with the negations of all conditions that appeared before. In this way, the logical meaning matches the computational behaviour of *Mathematica*, namely that a condition is only tested when all conditions before were already found out to be `False` .

The case distinction in the definition of plus says the following: If the two leading monomials are comparable, the result is the larger one of these monomials (e.g. `m`) prepended to the polynomial sum of the rest polynomial (e.g. $\langle\bar{p}\rangle$) and the polynomial with the smaller leading monomial (e.g. $\langle n,\ \bar{q}\rangle$). Otherwise they are tested whether one is the monomial negative of the other. If yes, the result is the polynomial sum of the two rest polynomials. Otherwise, the monomial sum of the two leading monomials has to be prepend to this sum of the rests.

The following is would be an alternative way of formatting the above case distinction. It is not part of the language right now, but we want to show it because we think that mathematicians could be more familiar with this style:

$$\left(\underset{P}{+}\right) \quad \langle m, \bar{p}\rangle \underset{P}{+} \langle n, \bar{q}\rangle := \begin{cases} m \underset{M}{>} n: & m \smile \left(\langle \bar{p}\rangle \underset{P}{+} \langle n, \bar{q}\rangle\right) \\ n \underset{M}{>} m: & n \smile \left(\langle m, \bar{p}\rangle \underset{P}{+} \langle \bar{q}\rangle\right) \\ m \underset{M}{=} -n: & \langle \bar{p}\rangle \underset{P}{+} \langle \bar{q}\rangle \\ \text{else}: & \left(m \underset{M}{+} n\right) \smile \left(\langle \bar{p}\rangle \underset{P}{+} \langle \bar{q}\rangle\right) \end{cases}$$

This would also make nested case statements easier readable. For example, one could want to collect the last two cases in one case ('eqivalent monomials'), which would look like this:

$$\left(\underset{P}{+}\right) \quad \langle m, \bar{p}\rangle \underset{P}{+} \langle n, \bar{q}\rangle := \begin{cases} \vdots & \\ \text{else}: & \begin{cases} m \underset{M}{=} -n: & \langle \bar{p}\rangle \underset{P}{+} \langle \bar{q}\rangle \\ \text{else}: & \left(m \underset{M}{+} n\right) \smile \left(\langle \bar{p}\rangle \underset{P}{+} \langle \bar{q}\rangle\right) \end{cases} \end{cases}$$

Following is the recursive definition of the negative, which will later be proven to provide the inverse element for plus:

$$\left(\underset{P}{-}\underset{P}{0}\right) \quad \underset{P}{-}\langle\rangle := \langle\rangle;$$
$$\left(\underset{P}{-}\right) \quad \underset{P}{-}\langle m, \bar{p}\rangle := \underset{M}{-}m \smile \underset{P}{-}\langle \bar{p}\rangle;$$

The negative of the zero polynomial is again the zero polynomial. The negative of a polynomial represented by a tuple having the element m and the sequence $\bar{p}$ as contents is defined by prepending the monomial negative of m to the polynomial negative of the polynomial represented by the tuple with the contents $\bar{p}$.

$$\left(\underset{P}{0}\underset{P}{*}\right) \quad \langle\rangle \underset{P}{*} p := \langle\rangle;$$
$$\left(\underset{P}{*}\underset{P}{0}\right) \quad p \underset{P}{*} \langle\rangle := \langle\rangle;$$
$$\left(\underset{P}{*}\right) \quad \langle m, \bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle :=$$
$$\left\| r \Leftarrow \text{zero-}\underset{M}{\text{divisor}}[m, n], \left(m \underset{M}{*} n\right) \smile r \right\|,$$
$$\text{where } r = \langle m\rangle \underset{P}{*} \langle \bar{q}\rangle \underset{P}{+} \langle \bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle$$
$$]$$

This is the the last definition in the functor, the polynomial times. In the general case, where both arguments are unequal zero, we use another construct of the *Theorema* language, the where . It is useful if a larger subterm occurs more than once in a term. We simply use one more variable, in this case r, to replace the long subterm. After we have defined the term in a convenient way, using the r, we have to state what this variable should be replaced for. This is done after the keyword where .

An alternative definition for $\underset{P}{*}$ would be to add the monomial product instead of prepending it, i.e. to use

$$\left\| r \Leftarrow \underset{M}{\text{zero-divisor}}[m, n], \left\langle m \underset{M}{*} n \right\rangle \underset{P}{+} r \right\|$$

as case term. In our proofs using $(\underset{P}{*})$ we will see the consequences of these different versions.

During all the previous definitions we made sure that the head of the result is always different from 'monomial zero' (assumed as not available). In the definition of $\underset{P}{+}$ this was done by distinguishing between the third and fourth case of the case distinction. The negative can also never get a leading zero under the assumption that $\underset{M}{-}$ keeps a monomial distinct from zero, i.e. is closed. In the definition of $\underset{P}{*}$ we have to do case dinstinction again, using the monomial function $\underset{M}{\text{zero-divisor}}$ to test whether the head monomials are zero divisors. In this case we do not prepend this monomial product to the rest term of the definition, otherwise we do.

■ 2.1.4: Monomials

We will now give an overview on the properties of admissible monomial domains and give an example of a suitable monomial functor.

There are two kinds of requirements to the argument `M` of the functor `PolynomialsAsMonomialTuples`. When building an instance of polynomials, initially we only require the signature Σ of `M` to contain the objects `∈, >, 1, -, +, *, zero-divisor` appearing in the definitions. But of course we want these symbols to carry the expected meaning, e.g. we want $\underset{M}{>}$ to be an order relation. To determine which properties we additionally require, we simply examine which of them we need during the polynomial proofs. By this, we get the set of 'weakest preconditions', such that we do not restrict the class of domains that could fit into the notion of 'monomials' more than necessary.

During our investigations up to now we collected the following list of assumptions (in the order they appear in the proofs):

Let l, m, n range over M.

($\underset{M}{-}$ closed) $\quad \underset{m\in M}{\forall} \; \underset{M}{-}m \in M,$

($\underset{M}{-}\underset{M}{>}\underset{M}{-}$) $\quad \underset{m,n}{\forall} \; m \underset{M}{>} n \Longrightarrow \underset{M}{-}m \underset{M}{>} \underset{M}{-}n,$

($\underset{M}{+}$ closed) $\quad \underset{m,n\in M}{\forall} \; m \underset{M}{\sim} n \wedge m \neq \underset{M}{-}n \Longrightarrow m \underset{M}{+} n \in M,$

($\underset{M}{+}\underset{M}{>}$) $\quad \underset{l,m,n}{\forall} \; m \underset{M}{>} n \wedge l \underset{M}{\sim} m \wedge l \neq \underset{M}{-}m \Longrightarrow l \underset{M}{+} m \underset{M}{>} n,$

($\underset{M}{*}$ closed) $\quad \underset{m,n\in M}{\forall} \; m \underset{M}{*} n \in M,$

($\underset{M}{-}\underset{M}{-}$) $\quad \underset{m}{\forall} \; \underset{M}{-}\underset{M}{-}m = m,$

($\underset{M}{+}$ comm) $\quad \underset{m,n}{\forall} \; m \underset{M}{\sim} n \wedge m \neq \underset{M}{-}n \Longrightarrow m \underset{M}{+} n = n \underset{M}{+} m,$

($\underset{M}{>}\underset{M}{+}$) $\quad \underset{l,m,n}{\forall} \; l \underset{M}{>} m \wedge l \underset{M}{>} n \wedge m \underset{M}{\sim} n \wedge m \neq \underset{M}{-}n \Longrightarrow l \underset{M}{>} m \underset{M}{+} n,$

($\underset{M}{>}$ trans) $\quad \underset{l,m,n}{\forall} \; l \underset{M}{>} m \wedge m \underset{M}{>} n \Longrightarrow l \underset{M}{>} n,$

$$(\underset{M}{\sim}\ \underset{M}{>})\qquad \underset{l,m,n}{\forall}\ l \underset{M}{>} m \wedge l \underset{M}{\sim} n \Longrightarrow n \underset{M}{>} m,$$

$$(\underset{M}{>}\ \underset{M}{\sim})\qquad \underset{l,m,n}{\forall}\ l \underset{M}{>} n \wedge m \underset{M}{\sim} n \Longrightarrow l \underset{M}{>} m,$$

$$(\underset{M}{\sim}\ \underset{M}{+})\quad \underset{l,m,n}{\forall}\ l \underset{M}{\sim} m \wedge m \underset{M}{\sim} n \wedge m \neq -n \Longrightarrow l \underset{M}{\sim} m \underset{M}{+} n$$

$$(\underset{M}{-}\ \underset{M}{+})\qquad \underset{m,n}{\forall}\ m \underset{M}{\sim} n \wedge m \neq -n \Longrightarrow \underset{M}{-}\left(m \underset{M}{+} n\right) = \underset{M}{-}m \underset{M}{+} \underset{M}{-}n,$$

$$(\text{no } \underset{M}{+}\underset{M}{0})\qquad \underset{m,n}{\forall}\ m \underset{M}{\sim} n \wedge m \neq -n \Longrightarrow m \neq m \underset{M}{+} n,$$

$$(\underset{M}{-}m \underset{M}{+} \left(m \underset{M}{+}\right))\quad \underset{m,n}{\forall}\ m \underset{M}{\sim} n \Longrightarrow \underset{M}{-}m \underset{M}{+} \left(m \underset{M}{+} n\right) = n,$$

$$(\underset{M}{+}\ \text{assoc})\qquad \underset{l,m,n}{\forall}\ l \underset{M}{\sim} m \wedge m \underset{M}{\sim} n \wedge l \neq \underset{M}{-}m \wedge m \neq \underset{M}{-}n \Longrightarrow \left(l \underset{M}{+} m\right) \underset{M}{+} n = l \underset{M}{+} \left(m \underset{M}{+} n\right),$$

$$(\underset{M}{1}\ \text{not zerodiv})\ \underset{m}{\forall}\ \neg\, \underset{M}{\text{zero-divisor}}\left[m, \underset{M}{1}\right],$$

$$(\text{neutral } \underset{M}{1})\qquad \underset{m}{\forall}\ m \underset{M}{*} \underset{M}{1} = m,$$

$$(\underset{M}{*}\ \text{comm})\qquad \underset{m,n}{\forall}\ m \underset{M}{*} n = n \underset{M}{*} m,$$

$$(\text{zerodiv symm})\ \underset{m,n}{\forall}\ \underset{M}{\text{zero-divisor}}[m, n] \Longleftrightarrow \underset{M}{\text{zero-divisor}}[n, m].$$

We have introduced an additional object here: the monomial equivalence relation $\underset{M}{\sim}$. One could have done all the proofs without using this, replacing it by its definition

$$(\underset{M}{\sim})\qquad \underset{m,n\in M}{\forall}\ m \underset{M}{\sim} n :\Longleftrightarrow \neg\, m \underset{M}{>} n \wedge \neg\, n \underset{M}{>} m.$$

It could be defined like this inside the monomial functor or as ‘external definition’, in the same way we will also define a function `ldm`, leading monomial, later, outside the polynomial functor. It is not decided yet whether to allow such additional definitions: The advantage is, that one does not have to extend a functor definition or even create a new functor to add the definition, on the other hand the system’s structure, based on fuctors, becomes less clear, because it is not immediate where the now definition belongs to.

We do not assume the weakest version of a formula all the time. This seems to be in contradiction to what we want and stated above, but we will show the reason why to proceed like this in an example:

During one of our proofs, we would like to have

$$(\underset{M}{+}\ \underset{M}{>}.2)\ \underset{l,m,n}{\forall}\ l \underset{M}{>} n \wedge l \underset{M}{\sim} m \wedge l \neq \underset{M}{-}m \Longrightarrow l \underset{M}{+} m \underset{M}{>} n,$$

which would be the weakest precondition for the prove. However, during the following proofs, it becomes necessary to state $(\underset{M}{-}\underset{M}{-})$. Since we need $(\underset{M}{-}\underset{M}{-})$ anyways, we can use it already in the initial proof to derive $(\underset{M}{+}\ \underset{M}{>}.2)$ from $(\underset{M}{+}\ \underset{M}{>})$, which we have.

All of the above formulae about monomial plus are implications of the type: If the arguments are equivalent, then some property has to hold. Later on we will see in an example proof that this equivalence condition is necessary that the property can hold in ‘standard’ monomial models, where the stronger formula without the equivalence check does not hold. However, our polynomial prover can in fact use e.g. the stronger

$(\underset{M}{+}$ comm.2) $\quad \underset{m,n}{\forall} \quad m \underset{M}{+} n = n \underset{M}{+} m\,,$

instead of the original ($\underset{M}{+}$ comm) internally for proving, such that it is not necessary to check the equivalence condition all the time commutativity is applied. We do not need to care whether $\underset{M}{+}$ is applied only for admissible arguments, because the only way it is introduced during our proofs is by using the definition of the polynomial $\underset{P}{+}$ in the case where the equivalence condition holds. Of course we want the stronger version because it is ‘simpler’, it has no condition, so it can be used directly as a rewrite rule during simplification of a term.

We will now describe the effects of not having a zero monomial. This fact itself can only be described by ‘not having a neutral element’. This even has to be used in one of our proofs, so it is stated as (no $\underset{M}{+}\underset{M}{0}$). One immediate consequence of not having ‘$\underset{M}{0}$ ’ is, that we have to demand the formula $(\underset{M}{-} m \underset{M}{+} \left(m \underset{M}{+}\right))$, which could be avoided totally if we had the unconditional associativity of $\underset{M}{+}$ available. However we do not, because otherwise a term like $\underset{M}{-}\dot{q}_0 \underset{M}{+} \dot{q}_0$ would appear in our proof, which is not a monomial, as required. (See proof description of ($\underset{P}{+}$ assoc).)

An example of a functor creating monomial domains is the following, presented similarly in [Buchberger, Kriftner, 1997]. It takes a coefficient domain (with $\underset{C}{\Sigma} = \left\{\underset{C}{0}, \underset{C}{1}, \underset{C}{+}, \underset{C}{-}, \underset{C}{*}\right\}$) and a domain of terms (with $\underset{T}{\Sigma} = \left\{\underset{T}{1}, \underset{T}{*}, \underset{T}{>}\right\}$) as input and represents monomials as cartesian product with additional operations. The cartesian product is generated by calling the functor $\times$, written in infix notation.

$\text{Monomials}[C\_, T\_] :=$
$\text{Functor}\Big[\langle M\rangle, \langle c_1, c_2, t, t_1, t_2\rangle, \langle C'\rangle,$

$M = C \times T, \text{ where } C' = C \Big/ \underset{C}{0};$

$\left(\underset{M}{\boldsymbol{\Sigma}}\right) \quad \underset{M}{\boldsymbol{\Sigma}} := \underset{M}{\boldsymbol{\Sigma}} \bigcup \{1, +, -, ., >, \text{zero-divisor}\};$

$\left(\underset{M}{1}\right) \quad \underset{M}{1} := \left\langle \underset{C}{1}, \underset{T}{1}\right\rangle;$

$\left(\underset{M}{+}\right) \quad \langle c_1, t\rangle \underset{M}{+} \langle c_2, t\rangle := \left\langle c_1 \underset{C}{+} c_2, t\right\rangle;$

$\left(\underset{M}{-}\right) \quad \underset{M}{-}\langle c, t\rangle := \left\langle \underset{C}{-} c, t\right\rangle$

$\left(\underset{M}{*}\right) \quad \langle c_1, t_1\rangle \underset{M}{*} \langle c_2, t_2\rangle := \left\langle c_1 \underset{C}{*} c_2, t_1 \underset{T}{*} t_2\right\rangle;$

$\left(\underset{M}{>}\right) \quad \langle c_1, t_1\rangle \underset{M}{>} \langle c_2, t_2\rangle :\Leftrightarrow t_1 \underset{T}{>} t_2;$

$\left(\text{zero-}\underset{M}{\text{divisor}}\right) \quad \text{zero-}\underset{M}{\text{divisor}}[\langle c_1, t_1\rangle, \langle c_2, t_2\rangle] :\Leftrightarrow c_1 \underset{C}{*} c_2 = \underset{C}{0}\Big]$

The signatures of the created monomial domains have the elements needed in the polynomial functor, but the above list of necessary properties of M translates into a list of properties of C and T , which have to be proved to be able to add the derived polynomial theorems to the knowledge base. For example the corresponding monomial prover can prove (no $\underset{M}{+}\underset{M}{0}$) like this:

Let the coefficient domain C be arbitrary but fixed.

Let the terms domain T be arbitrary but fixed.

Let M be Monomials[C, T] .

Let m, n range over M .

Let c, d range over C .

Let t range over T.

We prove

(no $\underset{M}{+}\underset{M}{0}$) $\underset{m,n}{\forall}\ m \underset{M}{\sim} n \Longrightarrow m \neq m \underset{M}{+} n$

under the provided assumptions.

We represent monomials as tuples, $m = \langle c, t\rangle$, $n = \langle d, t\rangle$ and prove

$\underset{c,d,t,u}{\forall}\ \langle c, t\rangle \neq \langle c, t\rangle \underset{M}{+} \langle d, t\rangle$.

We take $\langle c_0, t_0\rangle$, $\langle d_0, t_0\rangle \in M$ arbitrary but fixed and prove

$\langle c_0, t_0\rangle \neq \langle c_0, t_0\rangle \underset{M}{+} \langle d_0, t_0\rangle$

by contradiction.

We assume

$\langle c_0, t_0\rangle = \langle c_0, t_0\rangle \underset{M}{+} \langle d_0, t_0\rangle$.

We simplify the right hand side

to $\left\langle c_0 \underset{C}{+} d_0, t_0\right\rangle$ by $(\underset{M}{+})$.

We solve the equation and get

(assum) $d_0 = \underset{C}{0}$.

and prove a contradiction.

We prove

(contr) $\left\langle \underset{C}{0}, t_0\right\rangle \in M$.

By (assum), this follows from

$\langle d_0 , t_0 \rangle \in M$.

(contr) contradicts ($\in$ M) .

2.2: Basic Domain Properties

We will now start to go through all proofs needed for proving that our polynomials defined before are a commutative ring with identity. This is the top–down way of creating a theory: We have a proposition, a goal, and create all necessary subgoals and lemmata only at the moment when they are needed in the proof. In our special case, we also want to find out all necessary assumptions about the monomial domain. For doing this, we first assume nothing about the monomials and add the monomial assumptions, such as "monomial times is closed", only when they become necessary during a proof. The bottom–up way of creating a theory whould be to select some "easy" lemmata by hand, and to start proving them, in the hope that they will show up to be useful during later more complicated proofs. This way of theory creation in a textbook possibly leads to the situation, where the reader does not see the reason why a certain lemma is important for the following development, and why it has to look exactly as it does.

2.2.1: Algebraic Definitions and Their Application to Polynomials

We want to show that the polynomial domain $(P, \underset{P}{0}, \underset{P}{1}, \underset{P}{+}, \underset{P}{-}, \underset{P}{*})$ defined by our functor is a commutative ring with identity, independent of the actual representation of the monomials. For this, at the moment we apply the definitions of a ring, an abelian group, a monoid, … by hand. In the future these definitions will be used as input to the predicate logic prover, which then will call the corresponding special prover, in our case the polynomial prover, only after having applied these definitions.

A more general way to state the domain properties would be to say: $(P, \underset{P}{+}, \underset{P}{*})$ is a commutative ring with identity. Here one would have to prove the existence of suitable constants $\underset{P}{0}$, $\underset{P}{1}$ and $\underset{P}{-}$ first. Proving existentially quantified formulae, as for example

$$\underset{n}{\exists}\ \underset{p}{\forall}\ n \underset{P}{+} p = p ,$$

is however not something we can expect from an automatic prover, since it would have to find the right term for replacing the existentially quantified variable, which is easy here (taking $\underset{P}{0}$), but can be very difficult in general. Therefore we already provide the prover with the necessary constants and functions to prevent having existential quantifiers. However, during further development of polynomial theory, e.g. when we want to prove euclidean division ('There exist quotient and remainder …'), it will become a problem to predefine the needed functions.

The definitions of the properties printed in bold face give the universally quantified formulae we have to prove.

To prove that $(P, \underset{P}{0}, \underset{P}{1}, \underset{P}{+}, \underset{P}{-}, \underset{P}{*})$ is a commutative ring with identity, we have to prove:

- $(P, \underset{P}{0}, \underset{P}{+}, \underset{P}{-}, \underset{P}{*})$ is a ring.
- $\underset{P}{*}$ **is commutative**.

– $\underset{P}{1}$ **is an element of** P.
– $\underset{P}{1}$ **is a neutral element of** $\underset{P}{*}$.

To prove that $(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-},\ \underset{P}{*})$ is a ring, we have to prove:
– $(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-})$ is an abelian group.
– $(P,\ \underset{P}{*})$ is a semigroup.
– $\underset{P}{*}$ **is left and right distributive over** $\underset{P}{+}$.

To prove that $(P,\ \underset{P}{*})$ is a semigroup, we have to prove:
– $\underset{P}{*}$ **is closed in** P.
– $\underset{P}{*}$ **is associative**.

To prove that $(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-})$ is an abelian group, we have to prove:
– $(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-})$ is a group.
– $\underset{P}{+}$ **is commutative**.

To prove that $(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-})$ is a group, we have to prove:
– $(P,\ \underset{P}{0},\ \underset{P}{+})$ is a monoid.
– $\underset{P}{-}$ **is closed in** P.
– $\underset{P}{-}$ **is an inverse of** $\underset{P}{+}$ **.**

To prove that $(P,\ \underset{P}{0},\ \underset{P}{+})$ is a monoid, we have to prove:
– $(P,\ \underset{P}{+})$ is a semigroup.
– $\underset{P}{0}$ **is an element of** P.
– $\underset{P}{0}$ **is a neutral element of** $\underset{P}{+}$.

To prove that $(P,\ \underset{P}{+})$ is a semigroup, we have to prove:
– $\underset{P}{+}$ **is closed in** P.
– $\underset{P}{+}$ **is associative**.

■ 2.3: The Order of Proving

■ 2.3.1: Closure

The proofs of the formulae are ordered according to the following principle: Before using a function, the corresponding closure property is proved. The reason for this is, that, when we know that the function is closed, we can apply it (by using its definition) in the following proofs without having to worry about whether we stay in the domain. This simulates exactly the way mathematicians do typechecking: During textbook proofs, e.g. when an equality is used as rewrite rule, the reader never sees (and never wants to see) the typechecking reasoning. When the author generated the proof, he did this checking only in his head, without writing it down explicitely. Our prover now will stick to this method. That means, the subproof which infers that we are allowed to use a certain formula in a proof step is not printed, but it nevertheless has to be generated in the prover, since the prover must check internally whether the variables in the formula it wants to use have the correct type.

The prover not only checks for correctness of the types at each proof step. Knowing about the types of the occuring variables in a proof, it can immediately reduce conditional to unconditional equalities, when the condition only declares the needed type of the variables. The main advantage of this is, that our general *Theorema* simplifier can be called. This simplifier can only be used with unconditional equalities. We give an example: Let the formula

$$\underset{m\in M,\, f\in P}{\forall}\ \mathtt{ldm}[m \smile f] = m$$

be one of the assumptions in a proof. This formula is equivalent to

$$\underset{m,f}{\forall}\ m\in M \wedge f\in P \Longrightarrow \mathtt{ldm}[m \smile f] = m\ ,$$

therefore it is a conditional assumption, where the conditions are type constraints on the variables `m` and `f`. We would like to use this formula in the following, unconditional form:

$$\underset{m,f}{\forall}\ \mathtt{ldm}[m \smile f] = m.$$

This is only possible when we can be sure that the conditions hold anyways, whenever the equation is used in a simplification. We have to know that, whenever a term of the pattern

$\mathtt{ldm}[\ldots \smile \ldots]$

appears, the two arguments are of the types `M` and `P`. This can be derived when we know the types of the appearing variables and the fact that the occuring functions are closed.

■ 2.3.2: Number of nested Inductions

Consider the proofs concerning polynomial plus. There we have associativity, the neutral element, the inverse and commutativity. This order is ‘natural’ in the sense that with each additional property proven, we get a higher algebraic structure: first a semigroup, then a monoid, group and abelian group. This will also be the order used by the final prover, since the predicate logic prover will decompose the goal "$(P,\ \underset{P}{0},\ \underset{P}{+},\ \underset{P}{-})$ is abelian group" in this order. However, this order is only important if already proven theorems become useful as lemmata for further proofs. For the polynomial plus proofs this is only the case once, as we will see: The proof that the defined negative is the inverse to plus uses associativity of plus. Therefore we chose another order for presentation: The proofs are ordered by the number of polynomial variables, and therefore by the number of nested inductions. This has the advantage that we start with short proofs, where we can explain the details without losing the overview. We can then proceed to the larger proofs being able to abbreviate the descriptions.

The basic method of proving ’bottom−up’, in the reverse order the propositions were generated (’top−down’), is again reflected in the fact that we first prove all the theorems about $\underset{P}{+}$ before we start with theorems about $\underset{P}{*}$. Here the proofs are dependent, the $\underset{P}{+}$−proofs are needed for the $\underset{P}{*}$−proofs. This comes from the fact that $\underset{P}{*}$ is defined using $\underset{P}{+}$. The proofs about polynomial times again are ordered starting with the ones with fewest polynomial variables.

■ 2.3.3: Lemmata

Only some of the proofs work only by using the definitions of the appearing constants and functions. The basic tool of a human proof author is to react when a proof does not work. For example he looks at the final terms of an equality whose sides do not become equal during simplification and sees that only two subterms are different. He can try to find a lemma that would help the proof at this stage, and try to prove it. Good lemmata are as general as possible, such that they can be reused during further proofs. In our presentation we will always present the proofs of the (not yet available) lemmata just after the proofs where they were needed.

This main loop of stating a conjecture, trying a proof, examining the failed proof and finding a helping lemma will be supported by the final polynomial prover in the way that even in the case of a failing proof the created notebook will be easily readable and it will be easy to see where and why exactly a proof failed. One could even think of a future ‘analyser’ program that takes a failed proof and finds a suitable lemma.

■ 2.4: Remarks to the Proof Descriptions

During the following proof descriptions we use the following method to indicate the origin of proof parts or formulae:

– Grey background is used when we cite parts of the proofs as they are given in the appendix.

– Alternative versions of subproofs are printed on white background and marked by a cell frame. The same style is also used for formulae (assumptions, goals) not appearing in our version of the proof.

Due to limitations of the *Mathematica* environment we cannot display the cell group brackets in the proof parts we show, but we display them in the appendix. So the subgroup (and subproof) structure can only be seen there in detail – however usually one presented proof part corresponds to one subgroup. We use " ⋮ " if parts of a cell group is left out. The reader is encouraged to look at the proof in the appendix when reading the corresponding description. By this the overall proof structure can be seen much easier.

■ 2.5: Closure Proofs

According to our strategy concerning the order of proving, these are the first proofs we try. Since our prototype prover cannot deal with the $\in$ predicate at all, it is especially necessary to find out the wanted output of the future polynomial prover by hand typed models. At each proof we describe the additionally appearing proof structures and the necessary formatting features.

■ 2.5.1: Zero is a Polynomial

Our goal is: $0_P \in P$.

Since this is the first proof we examine, we will go to the proof notebook in every detail. For this, we start with all cell groups closed, and open them one by one.

This is how the notebook looks like, when all cell groups are closed:

> Let `P` be arbitrary but fixed, generated by `PolynomialsBySequenceVariables` .
>
> We prove
>
> $(0_P \in P)$ $0_P \in P$

As one can see, these are whole english phrases (except for missing punctuation, which cannot be avoided in the current design). It is important to try not to show truncated phrases at any level of the proof. As the reader can freely decide which cells are interesting for him and which ones not, we have to make sure that the proof stays readable whenever a cell is closed or opened.

The first phrase introduces the symbol `P` as being an arbitrary polynomial domain constant. It further provides information about the used knowledge base – it is extracted from the definition of the named functor.

The second phrase consists of three parts: The words "We prove" mark the following formula to be a goal. They are written in an extra line such that the label of the formula, $(0_P \in P)$, stays on the left side of the page. By this, that label can be found easily when one looks for the formula. The labels, especially the ones generated by the prover itself, should always give a hint on the semantics of a formula. Labels are useful to be able to refer to a certain formula at some location later in the proof. In *Mathematica* it is even possible to create a hyperlink to a formula. By this, one can easily take a look at the actual formula, if the label itself does not give enough information.

The last line, giving the goal, is the head of the closed proof cell group. Opening this group, the proof is displayed like this:

> Let `P` be arbitrary but fixed, generated by `PolynomialsBySequenceVariables` .
>
> We prove
>
> $(0_P \in P)$ $0_P \in P$
>
> under the provided assumptions

by simplification.

Again a whole natural language phrase is shown. This layer provides the reader with information about the main proof method. For some readers, this is already enough information, others can now accept the offered possibility to look at the assumptions needed or to look at the proof in more detail, by opening the corresponding cell group. The head of the assumptions group is the cell "under the provided assumptions", this is the opened group:

under the provided assumptions

$(\langle\rangle \in \mathrm{P})\quad \langle\rangle \in \mathrm{P}$,

$(\underset{\mathrm{P}}{0})\quad \underset{\mathrm{P}}{0} := \langle\rangle$

(In the following, we will not always redisplay the group headers, when the closed group has already been shown before.)

As expected, the proof that $\underset{\mathrm{P}}{0}$ is a polynomial needs only the definition of $\underset{\mathrm{P}}{0}$ and part of the definition of the element predicate $\in \mathrm{P}$. The prover's knowledge base contained the whole definition of $\in \mathrm{P}$, but it only lists the assumptions that are really needed during the proof.

The first layer of the actual proof becomes visible if we open the cell group with the head "by simplification". The two subgroups are closed here:

We simplify

and prove the equivalent

$\langle\rangle \in \mathrm{P}$.

The first subgroup presents a call to the term simplification procedure. The inference rule of simplification of a formula transforms it into an equivalent one, where a subterm is replaced by a 'simpler' one (corresponding to a built in order) during such a term simplification. Opening the group gives the information about which subterm of the formula was simplified, the labels of the applied equalities and the resulting terms after each application step:

We simplify

$\underset{\mathrm{P}}{0}$

to $\langle\rangle$ by $(\underset{\mathrm{P}}{0})$

At this stage there are no more closed subgroups. At each step we give the label of the used assumption, which is a hyperlink to the corresponding formula. This means the formula labeled $(\underset{\mathrm{P}}{0})$ can be displayed by clicking on the label, then the user can go back to the simplification by selecting 'Go Back' in the *Mathematica* menu.

For proving the remaining subgoal, $\langle\rangle \in \mathrm{P}$, we do not state the applied 'conclusion is assumption' inference rule, we only give the label of the corresponding assumption:

This follows from ($\langle\,\rangle \in \mathtt{P}$).

This ends the proof, since no more subgoals are left.

■ 2.5.2: One is a Polynomial

Our goal is: $\underset{\mathtt{P}}{1} \in \mathtt{P}$.

The proof works analogously to the last one. It uses the assumptions $\left\langle \underset{\mathtt{M}}{1} \right\rangle \in \mathtt{P}$ and $\underset{\mathtt{P}}{1} := \left\langle \underset{\mathtt{M}}{1} \right\rangle$, both extracted from the polynomial functor. The only difference is the appearence of the monomial domain constant M, and its connection to P. This information is given in the different proof header lines:

Let the monomial domain M be arbitrary but fixed .

Let P be `PolynomialsBySequenceVariables[M]` .

We prove
$\vdots$

■ 2.5.3: Negative is Closed

Our goal is: $\underset{\mathtt{p} \in \mathtt{P}}{\forall}\ \underset{\mathtt{P}}{-}\mathtt{p} \in \mathtt{P}$.

Initially we again declare M to be an arbitrary but fixed monomial domain and P to be the corresponding polynomials, but we additionaly make use of the possibility to omit domain underscripts in labels and formulae, by the line:

In the following, we omit the underscripts of $\underset{\mathtt{P}}{0}$, $\underset{\mathtt{P}}{-}$, $\underset{\mathtt{M}}{>}$.

Of couse it is a matter of taste to leave them out, one could also decide to keep some or all of them, because writing $-$ for $\underset{\mathtt{P}}{-}$ could cause confusion, when there is also an $\underset{\mathtt{M}}{-}$ appearing. It is good style to abbreviate the $\underset{\mathtt{M}}{>}$ symbol to $>$ anyways, since this stays unambiguous.

Omitting the underscripts could also be seen not as an abbreviation on the metalevel, but as defining local constants, with lines like

$0 := \underset{\mathtt{P}}{0}$.
$\vdots$

We prefer the first version, because it reflects the reason why we do the abbreviation: to improve readability. By this we can prevent having terms with underscripts below most symbols.

In order not to state the type of symbols every time they appear in formulae under a quantifier, we use type declarations. Their scope is again the proof notebook.

Let $\mathtt{p},\ \langle \bar{\mathtt{p}} \rangle$ range over $\mathtt{P}$.

Let $\mathtt{m},\ \mathtt{n}$ range over $\mathtt{M}$.

This also declares that the sequence variable $\bar{\mathtt{p}}$ ranges over 'the contents of a polynomial in sequence representation'. From now on, we can leave out the domain information under a quantifier, but we can also keep it, as our proposition shows:

We prove

(– closed) $\underset{\mathtt{p} \in \mathtt{P}}{\forall}\ \ -\mathtt{p} \in \mathtt{P}$

In such closure formulae it supports readability to always write all membership information, not to write $\underset{\mathtt{p}}{\forall}\ \ -\mathtt{p} \in \mathtt{P}$.

The assumptions needed for this proof are presented in three groups:

under the provided assumptions
$\vdots$
definitions
$\vdots$
and lemmata
$\vdots$

– First the assumptions about the lower level domains are listed, in our case the properties of the monomial functions. For this proof we need closure of monomial negative ($\bar{\mathtt{M}}$ closed) and preservation of the monomial order by negative ($\bar{\mathtt{M}} > \bar{\mathtt{M}}$).

– The used definitions, here ($\smile$), the definition of the global 'prepend' and the definitions of the used polynomial functions and predicates, are listed next. We need the three cases of the monomial element definition ($\langle\ \rangle \in$ P), ($\langle$m$\rangle \in$ P) and ($\in$ P), and the definition of polynomial negative, consisting of the base case ($-\mathtt{0}$) and the recursive general case ($-$).

– Finally, we list all lemmata that are already part of the current functor's theory or still have to be proved. This proof uses two of them, ($\smile \in \mathtt{P}$) speaks about the conditions for a term constructed by prepend to be a polynomial, and ($\mathtt{ldm}[\smile]$) generalizes the notion of a 'leading monomial' to polynomials constructed by prepend.

At the highest opened level of the proof, it states the main proof method, which is using an induction principle:

We prove
$\vdots$
by induction.

We open the next layer and see an preparatory step:

We represent polynomials in sequence notation and prove

$$\underset{\bar{p}}{\forall} \; -\langle \bar{p} \rangle \in P$$

by polynomial induction.

This is due to our general principle to make use of representation information as soon as possible. The original formula follows from this new one by substituting $p = \langle \bar{p} \rangle$.

We will now describe the polynomial induction cell group, opened by double−clicking on the cell bracket of "by polynomial induction". It looks like this:

Induction base, p zero polynomial: We prove

$$-\langle\,\rangle \in P\,.$$

Induction base, p single monomial: We take $\dot{p}_0 \in M$ arbitrary but fixed and prove

$$-\langle \dot{p}_0 \rangle \in P\,.$$

Induction step for p:
We take $\left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle \in P$ arbitrary but fixed, assume the induction hypothesis

(− closed.V.IH) $\;-\left\langle \ddot{p}_0,\ \overline{p_0} \right\rangle \in P$

and prove the induction conclusion

$$-\left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle \in P\,.$$

At this level only the formulae generated by application of the induction principle are presented. Each of the subgoals generated in this way is the head of a cellgroup containing its proof. The two base cases are not simply numbered, but named according to the special meaning of the case: We do not simply speak of ‘induction base one’, but we say that here we consider the zero polynomial. We always give the name of the induction variable, which helps navigation when we have nested induction, but for readability we use the former p instead of the actual $\langle \bar{p} \rangle$.

As can be seen here, only the generated assumption, the induction hypothesis, is labeled, since it will be used and therefore cited during the proof of the conclusion. The label was created by adding appendices to the label of the initial goal: The ".V" is appended because the sequence variables have been introduced, and the ".IH" denotes the induction hypothesis. By this, the label is not only an abbreviation for the formula, but also gives information about the location of the formula within the proof. Furthermore this ensures that labels are never created twice in a proof. During the following proofs we will present different versions of handling labelling, but surely the option of having labels attached to all generated formulae should always be available for the user, since only then he can cite each particular formula without having to describe its location.

Polynomial induction for polynomials represented as ordered lists of monomials generates three subgoals and is applied like this:

– The formula for the zero polynomial base case is generated by replacing the induction variable with the representation of the zero polynomial and omitting the quantifier.

– The formula for the single monomial base case is generated by replacing the induction variable by a tuple containing a new arbitrary but fixed monomial constant $\dot{p}_0$ and again omitting the quantifier.

– The induction step formula, which is a universally quantified implication, is not printed in the proof, but immediately decomposed in the following way: We use the proof rule of taking arbitrary but fixed constants $\dot{p}_0$, $\ddot{p}_0$, $\overline{p_0}$ and the modus ponens rule in one step, because we do not want to show the intermediate formula, and we do not want to introduce another level of cell nesting. We assume the left side of the implication, the induction hypothesis. This formula is generated by replacing the induction variable by the sequence $\ddot{p}_0, \overline{p_0}$. Similarly, the right side of the induction step implication is built replacing the induction variable by the sequence $\dot{p}_0, \ddot{p}_0, \overline{p_0}$. This formula, the induction conclusion, is now the goal and during its proof we can use the induction hypothesis.

Note that the described procedure is *not* an immediate application of the inference rule for polynomial induction,

$$\frac{A_{\bar{p}\leftarrow},\ \underset{m\in M}{\forall} A_{\bar{p}\leftarrow m},\ \underset{m\in M,\langle n,\bar{p}\rangle\in P}{\forall} A_{\bar{p}\leftarrow n,\bar{p}} \wedge m \underset{M}{>} n \Longrightarrow A_{\bar{p}\leftarrow m,n,\bar{p}}}{\underset{\bar{p}}{\forall} A}.$$

If we only used this rule, the second base case and the induction step would be presented like this:

⋮

Induction base, p single monomial: We prove

$$\underset{m}{\forall}\ -\langle m\rangle \in P.$$

Induction step for p: We prove

$$\underset{m,n,\bar{p}}{\forall}\ -\langle n, \bar{p}\rangle \in P \wedge m > n \Longrightarrow -\langle m, n, \bar{p}\rangle \in P.$$

Here these subproofs would need another layer to introduce the arbitrary but fixed constants, so the induction step subgroup would be

We take $\dot{p}_0 \in M$ and $\langle \ddot{p}_0, \overline{p_0}\rangle \in P$ arbitrary but fixed and prove

$$-\langle \ddot{p}_0, \overline{p_0}\rangle \in P \wedge \dot{p}_0 > \ddot{p}_0 \Longrightarrow -\langle \dot{p}_0, \ddot{p}_0, \overline{p_0}\rangle \in P.$$

with another subgroup, decomposing the implication:

We assume the induction hypothesis

(– closed.V.IH) $-\langle \ddot{p}_0, \overline{p_0}\rangle \in P \wedge \dot{p}_0 > \ddot{p}_0$

and prove the induction conclusion

$- \langle \dot{p}_0, \ddot{p}_0, \overline{p_0} \rangle \in P$.

The used version combines these steps of applying induction, introducing constants and assuming the hypothesis. Additionally it makes use of the fact that $\dot{p}_0 > \ddot{p}_0 \wedge \langle \ddot{p}_0, \overline{p_0} \rangle \in P$ by ($\in$ P) implies $\langle \dot{p}_0, \ddot{p}_0, \overline{p_0} \rangle \in P$. By this, the induction hypothesis can be reduced to the interesting part $- \langle \ddot{p}_0, \overline{p_0} \rangle \in P$.

The proof of the zero polynomial induction base case works by simplification as shown in the proof 'zero is a polynomial', using (-0) to get a formula equal to $(\langle\ \rangle \in P)$.

Next we examine the proof of the 'single monomial' base case, shown here with closed simplification subgroup:

Induction base, p single monomial: We take $\dot{p}_0 \in M$ arbitrary but fixed and prove

$- \langle \dot{p}_0 \rangle \in P$.

We simplify

and prove the equivalent

$\langle \underset{M}{-} \dot{p}_0 \rangle \in P$.

By $(\langle m \rangle \underset{P}{\in})$, this follows from

$\underset{M}{-} \dot{p}_0 \in M$.

This follows from ($\underset{M}{-}$ closed).

As always, simplification is done first. Here it applies the definition of the negative, using $(-)$, (-0) and $(\smile)$. The subproof is presented in a linear way. This is possible because the applied inference rules all generate only one new goal. We do not introduce a new proof layer for this single goal, but simply append its proof to the current cell group. Otherwise we would get a deeply nested proof structure, which is difficult to read. Presenting the proof in this way would have the only advantage that one could show only some steps at the beginning of the chain of subgoals and hide the rest. However, users normally want to see such a chain as a whole without having to open all the steps one by one, so this is how we format the output. This linearization of subproofs is a task for the postprocessor who creates the cell structure, the initially produced proof object should still contain a proof subobject for each subproof.

The two rules giving only one subgoal are simplification and the 'modus ponens' rule, that appears here for the first time in our proof descriptions. It is one of the inference rules implemented in the predicate logic prover and makes use of an assumption which is an implication. In our case, we use the implication $(\langle m \rangle \underset{P}{\in})$ to replace the goal $\langle \underset{M}{-} \dot{p}_0 \rangle \in P$ by $\underset{M}{-} \dot{p}_0 \in M$. This backwards reasoning is expressed by the phrase "By $(\langle m \rangle \underset{P}{\in})$, this follows from …", which also states the applied implication.

In fact we do not use $(\langle m \rangle \underset{P}{\in})$ directly, but a ground instance of it. This is a hidden application of the inference rule of 'specialization', by which we can add instances of a universally quantified assumption to the knowledge base. We could also show this specialization in detail, by listing the used variable substitutions, in this case $m \leftarrow \underset{M}{-} \dot{p}_0$. This is only useful when the substitution cannot be seen immediately by the reader, so we omit this information in the proofs presented in this paper, as it is done normally in mathematical textbooks. The same holds for the last step in this subproof, where an instance of the universally quantified ($\underset{M}{-}$ closed) is used with the ‘conclusion is assumption’ rule.

These hidden applications of the ‘specialization’ rule internally make use of type checking, since the variables under the quantifiers have range information attached. However it is task of the prover to internally do these checks (and task of the programmer designing the prover to know whether the checking could even be omitted), so they do not appear in the output proof.

Proving the induction step again starts with a simplification, followed by modus ponens:

We simplify

and prove the equivalent

$\underset{M}{-}\dot{p}_0 \smile \underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle \in P$.

By ($\smile \in$ P), this follows from

$\underset{M}{-}\dot{p}_0 > \mathtt{ldm}[\underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle] \vee \underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle = \langle\,\rangle$.

Simplification uses ($-$), leaving us with a goal that can be proved by using the lemma ($\smile \in$ P). This lemma has to be introduced here, because otherwise the proof would fail: There is no more applicable inference rule. Of course the final goal depends on the term order used by the simplification: Using another order it could have given $\underset{M}{-}\dot{p}_0 \smile - \left\langle \ddot{p}_0, \overline{p_0} \right\rangle \in P$ or even the initial $- \left\langle \dot{p}_0, \ddot{p}_0, \overline{p_0} \right\rangle \in P$ as goals. But none of these goals can be decomposed further.

Since inventing ‘good’ lemmata is a task left to the proof author, we created ($\smile \in$ P) and added it to the initial assumptions. It introduces the externally defined function `ldm`, whose definition does not come from a functor, but only appears in the proof of the lemma. The name is an abbreviation for ‘leading monomial’. During our actual proof we only need the property (`ldm[ ⌣ ]`) of `ldm`. This lemma has to be proved afterwards.

Since the application of ($\smile \in$ P) generated a conjunction, the further proof uses the (predicate logic prover) rule of assuming the negation of one part to prove the other. Another way of doing it would be to use a lemma ($\smile \neq \langle\,\rangle$), defined as

$$(\smile \neq \langle\,\rangle) \quad \underset{e,\bar{s}}{\forall} \; e \smile \bar{s} \neq \langle\,\rangle$$

to be able to continue with

$\vdots$

$\underset{M}{-}\dot{p}_0 > \mathtt{ldm}[\underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle] \vee \underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle = \langle\,\rangle$.

By ($\smile \neq \langle\,\rangle$), this follows from

$\underset{M}{-}\dot{p}_0 > \mathtt{ldm}[\underset{M}{-}\ddot{p}_0 \smile - \langle\overline{p_0}\rangle]$.

We reach the same goal without a lemma:

We assume

$${}_{\bar{\mathrm{M}}}\ddot{\mathrm{p}}_0 \smile - \langle \overline{\mathrm{p}_0} \rangle \neq \langle\ \rangle$$

and prove

$${}_{\bar{\mathrm{M}}}\dot{\mathrm{p}}_0 > \mathrm{ldm}[{}_{\bar{\mathrm{M}}}\ddot{\mathrm{p}}_0 \smile - \langle \overline{\mathrm{p}_0} \rangle] .$$

The last subproof again starts with simplification. Here the property ($\mathrm{ldm}[\smile]$) is used to be able to apply ldm to a term constructed by prepend, like ${}_{\bar{\mathrm{M}}}\ddot{\mathrm{p}}_0 \smile - \langle \overline{\mathrm{p}_0} \rangle$. This gives the last goal,

$${}_{\bar{\mathrm{M}}}\dot{\mathrm{p}}_0 > {}_{\bar{\mathrm{M}}}\ddot{\mathrm{p}}_0 .$$

The final modus ponens step again is an abbreviation. It simply says

By (${}_{\bar{\mathrm{M}}} > {}_{\bar{\mathrm{M}}}$), this follows from the definition of $\dot{\mathrm{p}}_0$ and $\ddot{\mathrm{p}}_0$.

This refers to the line at the beginning of the induction step, saying

We take $\left\langle \dot{\mathrm{p}}_0 , \ddot{\mathrm{p}}_0 , \overline{\mathrm{p}_0} \right\rangle \in \mathrm{P}$ arbitrary but fixed, …

In our version this hides the part $\dot{\mathrm{p}}_0 > \ddot{\mathrm{p}}_0$ of the assumption hypothesis.

The expanded form of the final step would have to use the expanded version of the induction hypothesis,

(– closed.V.IH) $- \left\langle \ddot{\mathrm{p}}_0 , \overline{\mathrm{p}_0} \right\rangle \in \mathrm{P} \wedge \dot{\mathrm{p}}_0 > \ddot{\mathrm{p}}_0$,

as it was presented as alternative. Then the subproof could be written as:

By (${}_{\bar{\mathrm{M}}} > {}_{\bar{\mathrm{M}}}$), this follows from

$$\dot{\mathrm{p}}_0 > \ddot{\mathrm{p}}_0 .$$

This follows from (– closed.V.IH).

■ 2.5.4: Lemma: Leading Monomial of a Polynomial constructed by Prepending

Our goal is: $\underset{\mathrm{m} \in \mathrm{M},\, \mathrm{p} \in \mathrm{P}}{\forall} \mathrm{ldm}[\mathrm{m} \smile \mathrm{p}] = \mathrm{m}$.

We present two proof versions that use the different available induction principles for our polynomial functor: polynomial and sequence induction. Both of them again use type declarations for the variables $\mathtt{p}$, $\bar{\mathtt{p}}$ and $\mathtt{m}$, and they assume only the definitions of prepending '$\smile$' and the leading monomial $\mathtt{ldm}$, which could in fact be generalized to 'first element of tuple' by leaving out the type constraints on the parameter. The assumptions are

$$(\smile)\quad \underset{e,s}{\forall}\ e \smile \langle \bar{s} \rangle := \langle e,\ \bar{s} \rangle$$

$$(\mathtt{ldm})\quad \underset{m,\bar{p}}{\forall}\ \mathtt{ldm}[\langle m,\ \bar{p} \rangle] := m$$

The preparatory steps are also the same in both versions. Since the goal formula is universally quantified over a monomial variable, we introduce an arbitrary but fixed constant $\mathtt{m}_0$ in addition to shifting to sequence variables: (Here the order of the two steps is not important.)

We take $\mathtt{m}_0 \in \mathtt{M}$ arbitrary but fixed and prove

$$\underset{p}{\forall}\ \mathtt{ldm}[m_0 \smile p] = m_0 .$$

We represent polynomials in sequence notation and prove

$$\underset{\bar{p}}{\forall}\ \mathtt{ldm}[m_0 \smile \langle \bar{p} \rangle] = m_0$$

First we present the proof by polynomial induction. Its inference rule is derived from the polynomial element predicate $\in \mathtt{P}$, but we do not list the formulae defining $\in \mathtt{P}$ in our assumptions cell group, because they are not explicitely used during the proof. This works because they are never cited by their label, which would of course make it necessary to enable the link from label to formula.

The induction subgoals are

$$\mathtt{ldm}[m_0 \smile \langle\,\rangle] = m_0$$
$$\vdots$$
$$\mathtt{ldm}\left[m_0 \smile \left\langle \dot{p}_0 \right\rangle\right] = m_0$$
$$\vdots$$
$$\mathtt{ldm}\left[m_0 \smile \left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle\right] = m_0$$

All of the subproofs work in the same way, by a 'proof by simplification'. By opening one additional layer of the proof, only this information is displayed, not the actual subproof. We show the induction conclusion proof as example:

$$\vdots$$

and prove the induction conclusion

$$\mathtt{ldm}\left[m_0 \smile \left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle\right] = m_0$$

by simplification.

Here opening the subgroup added just the last line. A proof by simplification works by calling the term simplifier for both sides of a ground equation and getting two syntactically equal simplified terms. It is displayed by two term simplification subgroups for the left and right hand side of the equality. In our three cases (the induction subgoals) the right hand side is always m_0, which cannot be simplified any more, so this simplification subgroup is omitted. We show the open term simplification again only for the induction conclusion:

We simplify the left hand side

to $\mathtt{ldm}\left[\left\langle m_0, \dot{p}_0, \ddot{p}_0, \overline{p_0}\right\rangle\right]$ by ($\smile$),

to m_0 by ($\mathtt{ldm}$)

The two base cases use the same sequence of applied equations: first ($\smile$), then ($\mathtt{ldm}$).

A different natural possibility would be to present simplification proofs like this:

We simplify the left hand side

and the right hand side

and get equal sides.

However it is not necessary to mention the equal sides explicitely here, since the fact that the simplification proof succeeds already contains the information that the two sides become equal.

This polynomial induction proof now never made use of the fact that the appearing sequences are ordered, which is the main property of polynomial tuples. So we know that the lemma holds for any tuple p, not just for $p \in P$. This is the reason why we do not have to use polynomial induction, but we can also succeed by simply taking the general sequence variable induction inference rule

$$\frac{A_{\bar{x}\leftarrow}, \ \forall_{x,\bar{x}} A \Longrightarrow A_{\bar{x}\leftarrow x,\bar{x}}}{\forall_{\bar{x}} A}.$$

The used method, induction over finite sequences, is again indicated in the head of the subproof:

We represent polynomials in sequence notation and prove

$\forall_{\bar{p}} \ \mathtt{ldm}[m_0 \smile \langle \bar{p} \rangle] = m_0$

by sequence induction.

As in the polynomial induction, we immediately introduce the arbitrary but fixed constants and break the induction step implication into local assumption (induction hypothesis) and goal (induction conclusion):

Induction base, $\bar{p}$ empty sequence: We prove

$\mathtt{ldm}[m_0 \smile \langle \rangle] = m_0$

Induction step for $\bar{p}$:

We take $\langle \overset{\bullet}{p}_0, \overline{p_0} \rangle \in P$ arbitrary but fixed, assume the induction hypothesis

$\mathrm{ldm}[m_0 \smile \langle \overline{p_0} \rangle] = m_0$

and prove the induction conclusion

$\mathrm{ldm}[m_0 \smile \langle \overset{\bullet}{p}_0, \overline{p_0} \rangle] = m_0$

Note that in both version of the proof the induction hypothesis is not used. This can be immediately made visible by not attaching a label to it, since it is never refered to. Inductive proofs that do not use the hypothesis could be seen as an application of an inference rule of ‘structural case distinction’, because the cases only differ in the structure that is assumed for the induction variable. In our case the first proof version distinguished between the tuple p having zero, one or more than one element, whereas the second version collects the last two cases into one case, p having more than zero elements. However it does not make sense to build ‘structural case distinction’ as extra inference rule into the prover, since it is just a special case of induction, and it is of course better to immediately add the induction hypothesis to the knowledge base instead of first trying the proof without it.

Again the proof works by simplification for all subgoals. The first subgoal of sequence induction is always equal to the first subgoal of polynomial induction, so their proofs are equal.

We will see during further proofs that is it often difficult to see whether the weaker sequence induction is already sufficient for proving. One could always try polynomial induction first, and then see whether the tuple order was used, as we did here. This lowers the risk of getting failing proofs, but brings two important disadvantages in the cases where sequence induction works. These problems are neglectable comparing the two versions here, but they will become far more noticeable in larger proofs:

– The simple fact that in polynomial induction the induction step formulae are more complicated can lead to difficult proofs for the induction step.

– Even worse is the danger of having to do similar work twice. In our case we only have to do an additional simplification subproof in the polynomial induction, but in general this can nearly double the overall length of the proof, because the proof of the ‘one monomial’ base case normally is nearly as complex as the proof of the induction step. An example is the proof of commutativity of polynomial plus, which we present in the polynomial induction version as output generated by a prototype prover and in the sequence induction version as hand typed proof notebook. The hand typed version has to go through the case distinction only once, while the polynomial induction proof contains four similar case distinctions, because of two levels of nested polynomial induction.

Therefore sequence induction should be used like simplification: Simplification is tried before going into any induction. If it works we are done and have a simple proof, if not we use the result, the simplified formula, for induction. Similarly, sequence induction should be tried before polynomial induction. If it works the proofs become short and elegant, if not we can at least use the base case subproof as zero polynomial base case in the following polynomial induction.

■ 2.5.5: Lemma: Conditions for Prepending to give a Polynomial

Our goal is: $\underset{m\in M,\, p\in P}{\forall} m > \mathrm{ldm}[p] \vee p = \langle\rangle \Longrightarrow m \smile p \in P.$

Sequence induction works again for this proof. In the proofs we showed so far, the initial simplification, tried before the induction, never had an effect on the subgoal that enters induction. In this proof it does, and this step is displayed as

$$\vdots$$

$$\underset{\bar{p}}{\forall}\ m_0 > \mathtt{ldm}[\langle \bar{p} \rangle] \vee \langle \bar{p} \rangle = \langle\rangle \Longrightarrow m_0 \smile \langle \bar{p} \rangle \in P\,.$$

We simplify

and prove the equivalent

$$\underset{\bar{p}}{\forall}\ m_0 > \mathtt{ldm}[\langle \bar{p} \rangle] \vee \langle \bar{p} \rangle = \langle\rangle \Longrightarrow \langle m_0, \bar{p} \rangle \in P\,.$$

Here the simplification subgroup is closed. Since simplification only works on ground formulae, the alternative explicit version would be to change to the constant $\overline{\bar{p}_0}$ before doing simplification, and changing back to the quantified version with variable $\bar{p}$ afterwards. This inference rule of proving a formula with a constant by proving the (usually) stronger universally quantified version is called 'generalization'. This is the explicite version:

$$\vdots$$

$$\underset{\bar{p}}{\forall}\ m_0 > \mathtt{ldm}[\langle \bar{p} \rangle] \vee \langle \bar{p} \rangle = \langle\rangle \Longrightarrow m_0 \smile \langle \bar{p} \rangle \in P\,.$$

We take $\langle \overline{\bar{p}_0} \rangle \in P$ arbitrary but fixed and prove

$$m_0 > \mathtt{ldm}[\langle \overline{\bar{p}_0} \rangle] \vee \langle \overline{\bar{p}_0} \rangle = \langle\rangle \Longrightarrow m_0 \smile \langle \overline{\bar{p}_0} \rangle \in P\,.$$

We simplify

and prove the equivalent

$$m_0 > \mathtt{ldm}[\langle \overline{\bar{p}_0} \rangle] \vee \langle \overline{\bar{p}_0} \rangle = \langle\rangle \Longrightarrow \langle m_0, \overline{\bar{p}_0} \rangle \in P\,.$$

by generalization.

We prove

$$\underset{\bar{p}}{\forall}\ m_0 > \mathtt{ldm}[\langle \bar{p} \rangle] \vee \langle \bar{p} \rangle = \langle\rangle \Longrightarrow \langle m_0, \bar{p} \rangle \in P\,.$$

It has the simplification subgroup

We simplify

$$m_0 \smile \langle \overline{\bar{p}_0} \rangle$$

to $\langle m_0, \overline{\bar{p}_0} \rangle$ by $(\smile)$

In order to hide this shifting from variables to constants and back we proceed like this: The generated proof object still contains the constant $\overline{\bar{p}_0}$ and the steps to create it and remove it. During the creation of the natural language proof notebook by the postprocessor the constant is replaced by the original variable $\bar{p}$ in the simplification terms. Therefore the actual simplification in our proof looks like this:

We simplify

$m_0 \smile \langle \bar{p} \rangle$

to $\langle m_0, \bar{p} \rangle$ by ($\smile$)

The two subgoals of the sequence induction are both ground implications, so they are both proved by assuming the left hand side and proving the right hand side. However, both the base case and the induction step show interesting details:

The base case formula is

$$m_0 > \mathrm{ldm}[\langle\rangle] \vee \langle\rangle = \langle\rangle \Longrightarrow \langle m_0 \rangle \in P.$$

In order to use this formula, we have to define the leading monomial of the zero polynomial $\mathrm{ldm}[\langle\rangle]$, or we allow undefined terms to appear. Since the formula is proved by proving the right hand side by ($\langle m \rangle \in P$) without using the left hand side, having an undefined term in the unused formula causes no problems.

The induction conclusion contains no undefined terms, but an equality that can be immediately evaluated: $\langle m_1, \overline{p_0} \rangle = \langle\rangle$. But as first step we simplify the term $\mathrm{ldm}\left[\left\langle \dot{p}_0, \overline{p_0} \right\rangle\right]$ by (ldm):

We simplify

and prove the equivalent

$m_0 > \dot{p}_0 \vee \langle m_1, \overline{p_0} \rangle = \langle\rangle \Longrightarrow \left\langle m_0, \dot{p}_0, \overline{p_0} \right\rangle \in P.$

Because of the built in properties of equality, the subformula $\langle m_1, \overline{p_0} \rangle = \langle\rangle$ is false. Therefore predicate logic allows to change the formula:

This is equivalent to

$m_0 > \dot{p}_0 \Longrightarrow \left\langle m_0, \dot{p}_0, \overline{p_0} \right\rangle \in P.$

Only here we assume $m_0 > \dot{p}_0$ and prove $\left\langle m_0, \dot{p}_0, \overline{p_0} \right\rangle \in P$ as last subproof. There the assumption is used, therefore we display the label (assum). The long exact label ($\smile \in$ P .F.V.S.IC.S.=.assum) would show the proof steps applied so far: "F" for taking a fixed constant, "V" for using sequence variables, "S" for the simplification step, "IC" for being in the proof of the induction conclusion, another simplification, "=" for evaluating a ground equality and finally the "assum", indicating that we broke an implication into parts. Our short version (assum) shows another way of labelling: We omit as much of the long label as possible and indicate only the last proof step used to generate the formula. This improves readability in short proofs where navigation is not a problem (so the exact label is not needed) and certain types of formulae occur only once with a label (like in our case the local assumption in the proof of an implication).

We show the last subproof (with goal $\langle m_0 , \dot{p}_0 , \overline{p_0} \rangle \in P$) first in an expanded alternative version:

By ($\in$ P), this follows from

$\langle \dot{p}_0 , \overline{p_0} \rangle \in P \wedge m_0 > \dot{p}_0$.

We prove the individual conjunctive parts.

The first of the two subproofs would refer to "We take $\langle \dot{p}_0 , \overline{p_0} \rangle \in P$ arbitrary but fixed, …", saying

We prove

$\langle \dot{p}_0 , \overline{p_0} \rangle \in P$.

This follows from the definition of $\dot{p}_0$ and $\overline{p_0}$.

The second one would simply state

We prove

$m_0 > \dot{p}_0$.

This follows from (assum).

Similarly to the proof of closure of polynomial negative, we abbreviate this last subproof in one line, but here we have to give the two justifications for the conjunctive parts. We combine them like this:

⋮

and prove

$\langle m_0 , \dot{p}_0 , \overline{p_0} \rangle \in P$.

By ($\in$ P), this follows from (assum) and the definition of $\dot{p}_0$ and $\overline{p_0}$.

■ 2.5.6: Plus is Closed

Our goal is: $\underset{p,q \in P}{\forall} \; p \underset{P}{+} q \in P$.

One of the introduced abbreviations is to use + for $\underset{P}{+}$.

This is now the first proof using case distinction, because we have to use the definition of polynomial plus, which is done by cases in the general definition (+):

(+0) $\underset{p}{\forall}\ p + \langle\rangle := \langle\rangle,$

(0+) $\underset{p}{\forall}\ \langle\rangle + p := \langle\rangle,$

(+) $$\underset{m,n,\bar{p},\bar{q}}{\forall}\ \langle m, \bar{p}\rangle + \langle n, \bar{q}\rangle := \left\| \begin{array}{ll} m \smile (\langle\bar{p}\rangle + \langle n, \bar{q}\rangle) & \Leftarrow m > n, \\ n \smile (\langle m, \bar{p}\rangle + \langle\bar{q}\rangle) & \Leftarrow n > m, \\ \langle\bar{p}\rangle + \langle\bar{q}\rangle & \Leftarrow m = -n, \\ \left(m \underset{M}{+} n\right) \smile (\langle\bar{p}\rangle + \langle\bar{q}\rangle) & \end{array} \right\|$$

Furthermore we need the monomial properties ($\underset{M}{+}$ closed), ($\underset{M}{+}$ >), (−−) and ($\underset{M}{+}$comm), the usual definitions of prepending ($\smile$) and `ldm` and the lemmata ($\smile \epsilon$ P) and

(> ldm[+]) $\underset{m,p,q}{\forall}\ m > \mathrm{ldm}[p] \wedge m > \mathrm{ldm}[q] \wedge p + q \neq \langle\rangle \Longrightarrow m > \mathrm{ldm}[p + q]$.

While ($\smile \epsilon$ P) has been proved before, we will leave out the proof of (> ldm[+]) in this paper.

The proof works by polynomial induction on `p`, and polynomial induction on `q` in the monomial base case and the induction step of `p`. This generates seven subgoals. Three of them deal with addition of zero and can be proved by simplification, the other, nontrivial ones are:

$\vdots$

(+ closed.V.IB⟨m⟩.IB⟨m⟩) $\left\langle \dot{p}_0 \right\rangle + \left\langle \dot{q}_0 \right\rangle \in P$

$\vdots$

(+ closed.V.IB⟨m⟩.IC) $\left\langle \dot{p}_0 \right\rangle + \left\langle \dot{q}_0,\ \ddot{q}_0,\ \overline{q_0} \right\rangle \in P$

$\vdots$

(+ closed.V.IC.IB⟨m⟩) $\left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle + \left\langle \dot{q}_0 \right\rangle \in P$

$\vdots$

(+ closed.V.IC.IC) $\left\langle \dot{p}_0,\ \ddot{p}_0,\ \overline{p_0} \right\rangle + \left\langle \dot{q}_0,\ \ddot{q}_0,\ \overline{q_0} \right\rangle \in P$

$\vdots$

These are the locations in the proof where case distinction is triggered. The goals contain a nontrivial (= no zero argument) ground polynomial sum. Case distinction in the proof of (+ closed.V.IB⟨m⟩.IB⟨m⟩) looks like this:

Case $\dot{p}_0 > \dot{q}_0$:

Case $\dot{q}_0 > \dot{p}_0$:

Case $\dot{p}_0 = -\dot{q}_0$:

Case otherwise:

The case assumptions are generated according to the cases in the definition of polynomial plus. The subgoals of the different cases are not yet presented at this level, because they have to be computed first, as the initial step of a case subproof, using simplification. In the case assuming $\dot{p}_0 > \dot{q}_0$ the initial simplification looks like this:

Case $\dot{p}_0 > \dot{q}_0$:

We simplify

$\langle \dot{p}_0 \rangle + \langle \dot{q}_0 \rangle$

to $\dot{p}_0 \smile (\langle\rangle + \langle \dot{q}_0 \rangle)$ by (+),

to $\dot{p}_0 \smile \langle \dot{q}_0 \rangle$ by (0+),

to $\langle \dot{p}_0 , \dot{q}_0 \rangle$ by ($\smile$)

and prove the equivalent

(+ closed.V.IB⟨m⟩.IB⟨m⟩.C1) $\langle \dot{p}_0 , \dot{q}_0 \rangle \in P$.

The simplified formula is now the case subgoal, therefore the label gets the appendix "C1", where 1 is the number of the case. Normally, the definition (+) can only be used as conditional rewrite rule, here the case assumption is designed such that there is a subterm that can be simplified without justification.

The further proofs of the case subgoals use the definitions of the polynomial element predicate, together with the respective case assumption. This case assumption is used implicitely, as can be seen in the following line that completes the proof of the first case shown above:

This follows from (ϵ P) and (⟨m⟩ ϵ P).

Since the last case of the definition of plus uses monomial plus, we need closure of the monomial plus for the proof of the last case subgoal.

During the case distinctions subproofs of the other three nontrivial induction subgoals we need the listed lemmata ($\smile$ ϵ P) and ($>$ ldm[+]), for example in the case $\dot{q}_0 > \dot{p}_0$ of the proof of (+ closed.V.IB⟨m⟩.IC). We show this subproof in detail in the appendix, here we only want to remark that there it is necessary to refer to the case assumption explicitely. One could use labels again, but we simply state

This follows from the case assumption.

Until now we could always abbreviate subproofs using ($\in$ P). During some of the case subproofs we get into proof situations where this abbreviation would hide important proof steps that explain why we need a monomial assumption like ($\underset{M}{+}$ $>$). So there we use ($\in$ P) explicitely:

(+ closed.V.IB⟨m⟩.IC.C4) $\langle \dot{p}_0 \underset{M}{+} \dot{q}_0 , \ddot{q}_0 , \overline{q_0} \rangle \in P$.

By (ϵ P), this follows from

$\dot{p}_0 \underset{M}{+} \dot{q}_0 \in M \wedge \dot{p}_0 \underset{M}{+} \dot{q}_0 > \ddot{q}_0 \wedge \langle \ddot{q}_0 , \overline{q_0} \rangle \in P$.

The proof of the second conjunctive part needs $(\underset{M}{+}>)$:

We prove

$$\dot{p}_0 \underset{M}{+} \dot{q}_0 > \ddot{q}_0$$

By $(\underset{M}{+}>)$, this follows from the case assumptions and

$$\dot{q}_0 > \ddot{q}_0 .$$
$\vdots$

In situations where the proof fails, one can try to find a useful lemma, add it to the assumptions and use it for proving. This creates the necessity to prove the lemma afterwards. The following example appears in the 'otherwise' case of proving (+ closed.V.IC.IB⟨m⟩), so the prover implicitly knows $\dot{p}_0 \sim \dot{q}_0$ and $\dot{p}_0 \neq -\dot{q}_0$. We have to prove

$$\dot{p}_0 \underset{M}{+} \dot{q}_0 > \ddot{p}_0 .$$

The lemma that fits exactly to the given assumptions and the goal is

$$(\underset{M}{+}>.2) \quad \forall_{l,m,n}\ l > n \wedge l \sim m \wedge l \neq -m \Longrightarrow l \underset{M}{+} m > n .$$

By using this lemma, one could immediately create the subproof:

By $(\underset{M}{+}>.2)$, this follows from the case assumptions and

$$\dot{p}_0 > \ddot{p}_0 .$$

This follows from the definition of $\dot{p}_0$ and $\ddot{p}_0$.

However, we already have a similar assumption available:

$$(\underset{M}{+}>) \quad \forall_{l,m,n}\ m > n \wedge l \sim m \wedge l \neq -m \Longrightarrow l \underset{M}{+} m > n .$$

The new lemma $(\underset{M}{+}>.2)$ follows from $(\underset{M}{+}>)$ when we assume $(--)$ and $(\underset{M}{+}\text{comm})$. Since we have to demand these properties anyways for later proofs (e.g. the proof of commutativity of polynomial plus), we add them to the current assumptions and avoid $(\underset{M}{+}>.2)$ by using $(\underset{M}{+}>)$ instead:

We prove

$$\dot{p}_0 \underset{M}{+} \dot{q}_0 > \ddot{p}_0$$

We simplify

$\dot{p}_0 \underset{M}{+} \dot{q}_0$

to $\dot{q}_0 \underset{M}{+} \dot{p}_0$ by ($\underset{M}{+}$ comm)

and prove the equivalent

$\dot{q}_0 \underset{M}{+} \dot{p}_0 > \ddot{p}_0$

By ($\underset{M}{+}$ >), this follows from the case assumptions and

$\dot{p}_0 > \ddot{p}_0$.

$\vdots$

(We use (−−) implicitely to prove $\dot{q}_0 \neq -\dot{p}_0$ from the case assumption $\dot{p}_0 \neq -\dot{q}_0$. This is hidden in "... this follows from the case assumptions ...".) By doing the subproof like this, we do not have to introduce the new lemma. However, programming the prover to proceed like this will cause problems, because we need simplification to change $\dot{p}_0 \underset{M}{+} \dot{q}_0$ to $\dot{q}_0 \underset{M}{+} \dot{p}_0$ by ($\underset{M}{+}$ comm), while during other simplifications of this proof we have to keep $\dot{p}_0 \underset{M}{+} \dot{q}_0$ unchanged.

The last case of the final induction step proof shows again how reasoning about matching types is hidden from the proof:

$\vdots$

(+ closed.V.IC.IC.C4) $\left\langle \dot{p}_0 \underset{M}{+} \dot{q}_0 \right\rangle \smile \left(\left\langle \ddot{p}_0 , \overline{p_0} \right\rangle + \left\langle \ddot{q}_0 , \overline{q_0} \right\rangle \right) \in \mathtt{P}$.

By ($\smile \in$ P), this follows from

$\dot{p}_0 \underset{M}{+} \dot{q}_0 > \mathrm{ldm}\left[\left\langle \ddot{p}_0 , \overline{p_0} \right\rangle + \left\langle \ddot{q}_0 , \overline{q_0} \right\rangle \right] \vee \left\langle \ddot{p}_0 , \overline{p_0} \right\rangle + \left\langle \ddot{q}_0 , \overline{q_0} \right\rangle = \langle \rangle$.

Here the lemma ($\smile \in$ P) is applied with its polynomial variable p substituted by the term $\left\langle \ddot{p}_0 , \overline{p_0} \right\rangle + \left\langle \ddot{q}_0 , \overline{q_0} \right\rangle$, which has to be internally checked to be a polynomial. It would also make sense to display this typechecking as subproof, to make visible that the induction hypothesis

(+ closed.V.IH) $\underset{\overline{q}}{\forall} \left\langle \ddot{p}_0 , \overline{p_0} \right\rangle + \langle \overline{q} \rangle \in \mathtt{P}$

is used:

$\vdots$

By ($\smile \in$ P),

which we can use because

$\left\langle \ddot{p}_0, \overline{p_0} \right\rangle + \left\langle \ddot{q}_0, \overline{q_0} \right\rangle \in P$,

this follows from

Opening the next level of the subproof (with the goal $\left\langle \ddot{p}_0, \overline{p_0} \right\rangle + \left\langle \ddot{q}_0, \overline{q_0} \right\rangle \in P$ as head cell), we then get

$\vdots$

$\left\langle \ddot{p}_0, \overline{p_0} \right\rangle + \left\langle \ddot{q}_0, \overline{q_0} \right\rangle \in P$

This follows from (+ closed.V.IH) .

■ 2.5.7: Times is Closed

Our goal is: $\underset{p,q\in P}{\forall} p \underset{P}{*} q \in P$.

The usual abbreviations are given by:

In the following, we omit the underscripts of $\underset{P}{0}$, $\underset{P}{*}$, $\underset{P}{+}$.

We assume the monomial times to be closed:

($\underset{M}{*}$ closed) $\underset{m,n\in M}{\forall} m \underset{M}{*} n \in M$.

The definition of polynomial times used for this proof is

$(*0)\quad \underset{p}{\forall}\ p * \langle\rangle := \langle\rangle$,

$(0*)\quad \underset{p}{\forall}\ \langle\rangle * p := \langle\rangle$,

$(*)\quad \underset{m,n,\bar{p},\bar{q}}{\forall}\ \langle m, \bar{p}\rangle * \langle n, \bar{q}\rangle := \left\langle m \underset{M}{*} n \right\rangle + (\langle m\rangle * \langle \bar{q}\rangle + \langle \bar{p}\rangle * \langle n, \bar{q}\rangle)$.

This is *not* the same definition as given in the functor `PolynomialsBySequenceVariables`, saying

$$\left(\underset{P}{*}\right) \quad \langle m, \bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle :=$$
$$\left\| r \Leftarrow \underset{M}{\text{zero-divisor}}[m, n],\ \left(m \underset{M}{*} n\right) \smile r \right\|,$$
$$\text{where } r = \langle m\rangle \underset{P}{*} \langle \bar{q}\rangle \underset{P}{+} \langle \bar{p}\rangle \underset{P}{*} \langle n, \bar{q}\rangle$$

in the general case. It differs in two aspects:

– We leave out the first case of the case statement, since we assume the monomial times $\underset{M}{*}$ to be closed. Therefore the `zero-divisor` predicate never evaluates to true, since we have no zero monomial. On the other hand demanding ($\underset{M}{*}$ closed) is very strong, it excludes interesting monomial domains, for example monomials with coefficients from a cyclic group. Therefore we will use the `zero-divisor` case distinction in the other proofs.

– The term $\mathtt{m} \underset{M}{*} \mathtt{n}$ is combined with the rest of the defining term not by prepending but by polynomial addition. This turns out to make the closure proof much easier, since the closure of polynomial addition can be applied here, which is already proved. In order to use this proof with the definition of the original functor we would have to prove that the two versions are equivalent.

The proof we present makes use of a particular trick that a human proof author can easily do, but a proving program cannot be expected to perform by itself: The proof works by sequence induction first on q, the second universally bound variable, then on p . The polynomial prover instead always takes the leftmost variable as the induction variable. This enables the user to give the order of inductions by ordering the variables under the quantifier correspondingly. Therefore in our case the proof works if we give

($*$ closed) $\underset{q,p \in P}{\forall}\ p * q \in P$

as goal.

Finding out that this trick could work was possible by looking at the failed sequence induction proof for the original formula. So we have to give the user the possibility to decide whether he wants the prover to proceed with polynomial induction, if sequence induction fails, or to stop and display the failed sequence induction proof for examination. In our case, the failing proof version showed the induction hypotheses

(* closed.V.IH) $\underset{\bar{q}}{\forall}\ \langle \overline{p_0} \rangle * \langle \bar{q} \rangle \in P$

$\vdots$

(* closed.V.IC.IH) $\left\langle \dot{p}_0,\ \overline{p_0} \right\rangle * \langle \overline{q_0} \rangle \in P$

and the goals

$\left\langle \dot{p}_0 \right\rangle * \langle \bar{q}_0 \rangle \in P$

$\vdots$

$\langle \overline{p_0} \rangle * \left\langle \dot{q}_0,\ \overline{q_0} \right\rangle \in P$

while the version with swapped induction variables shows the hypotheses

(* closed.V.IH) $\underset{\bar{p}}{\forall}\ \langle \bar{p} \rangle * \langle \overline{q_0} \rangle \in P$

$\vdots$

(* closed.V.IC.IH) $\langle \overline{p_0} \rangle * \left\langle \dot{q}_0,\ \overline{q_0} \right\rangle \in P$

and the same goals. Now the goals can be proved directly.

■ 2.6: Proofs using Addition

■ 2.6.1: Zero is Neutral Element

Our goal is: $\underset{p\in P}{\forall}\ p \underset{P}{+} \underset{P}{0} = p.$

This proof works immediately by simplification, using the definition of $\underset{P}{0}$ and the formula $(+0)$ from the definition of $\underset{P}{+}$.

■ 2.6.2: Negative is an Inverse of Plus

Our goal is: $\underset{p\in P}{\forall}\ p \underset{P}{+} \underset{P}{-}p = \underset{P}{0}$

This is another proof working with sequence induction. Besides the monomial assumption $(\underset{M}{-}\ \underset{M}{-})$ and the definitions of polynomial zero, plus and inverse, we need the already proved ($-$ closed), the lemma

$$(\smile\ =\ +)\quad \underset{m,p}{\forall}\ m \smile p \in P \Longrightarrow m \smile p = \langle m\rangle + p$$

and associativity of polynomial plus.

The simplification proof of the induction conclusion shows a new structure that has not been needed so far: The use of conditional rewrite rules during simplification makes it necessary to create subproofs that verify the conditions. Simplification with closed subproofs looks like this:

We simplify the left hand side

to $\left\langle \dot{p}_0,\ \overline{p_0}\right\rangle + \underset{M}{-}\dot{p}_0 \smile - \langle\overline{p_0}\rangle$ by ($-$),

to $\left\langle \dot{p}_0,\ \overline{p_0}\right\rangle + \left(\left\langle \underset{M}{-}\dot{p}_0\right\rangle + -\langle\overline{p_0}\rangle\right)$ by ($\smile\ =\ +$),

to $\left(\left\langle \dot{p}_0,\ \overline{p_0}\right\rangle + \left\langle \underset{M}{-}\dot{p}_0\right\rangle\right) + -\langle\overline{p_0}\rangle$ by (+ assoc),

to $(\langle\overline{p_0}\rangle + \langle\,\rangle) + -\langle\overline{p_0}\rangle$ by (+),

to $\langle\overline{p_0}\rangle + -\langle\overline{p_0}\rangle$ by (+0),

to $\langle\,\rangle$ by (+−.V.IH)

Here the conditional rewrite rule ($\smile\ =\ +$) is applied. Furthermore we use (+) as conditional rewrite rule, as if we had

$$\underset{m,n,\bar{p},\bar{q}}{\forall}\ m = \underset{M}{-}n \Longrightarrow \langle m,\ \bar{p}\rangle + \langle n,\ \bar{q}\rangle = \langle\bar{p}\rangle + \langle\bar{q}\rangle.$$

The proof line showing the application of a conditional equation is the head of the subproof of the corresponding condition. So the above simplification contains two closed subgroups. Of course opening them makes the simplification unreadable – see the following proof part, where the last line continues the first line phrase. We have not found a good way to keep readability in all situations yet.

The following subproof of the condition for applying ($\smile\ =\ +$) shows once again the problem that equations have to be used in different directions during simplification. Here the equation ($-$) has to be used from right to left, while in the above simplification it was used from left to right on the same ground terms.

$\vdots$

to $\left\langle \dot{\mathtt{p}}_0\,,\ \overline{\mathtt{p}_0} \right\rangle + \left(\left\langle \underset{M}{-}\dot{\mathtt{p}}_0 \right\rangle + -\langle \overline{\mathtt{p}_0} \rangle \right)$ by ($\smile\ =\ +$),

which we can use because

$\underset{M}{-}\dot{\mathtt{p}}_0 \smile -\langle \overline{\mathtt{p}_0} \rangle \in \mathtt{P}.$

We simplify

$\underset{M}{-}\dot{\mathtt{p}}_0 \smile -\langle \overline{\mathtt{p}_0} \rangle$

to $-\left\langle \dot{\mathtt{p}}_0\,,\ \overline{\mathtt{p}_0} \right\rangle$ by ($-$)

and prove the equivalent

$-\left\langle \dot{\mathtt{p}}_0\,,\ \overline{\mathtt{p}_0} \right\rangle \in \mathtt{P}.$

This follows from ($-$ closed).

to $\left(\left\langle \dot{\mathtt{p}}_0\,,\ \overline{\mathtt{p}_0} \right\rangle + \left\langle \underset{M}{-}\dot{\mathtt{p}}_0 \right\rangle \right) + -\langle \overline{\mathtt{p}_0} \rangle$ by (+ assoc),

$\vdots$

The problem of formatting a simplification occurs even in the case of unconditional rewrite rules, as soon as we want to add information about the applied variable substitution for each line, possibly in a subgroup again. In our proofs we do not give this information, because the reader can figure out immediately which ground instanciation of the rule was applied.

■ 2.6.3: Lemma: Prepending is Summation

Our goal is: $\underset{\mathtt{m}\in\mathtt{M},\,\mathtt{p}\in\mathtt{P}}{\forall}\ \mathtt{m} \smile \mathtt{p} \in \mathtt{P} \Longrightarrow \mathtt{m} \smile \mathtt{p} = \langle \mathtt{m} \rangle \underset{\mathtt{P}}{+} \mathtt{p}.$

This is another example of a proof by sequence induction, and of using the definition of $\underset{\mathtt{P}}{+}$ as conditional rewrite rule, here using the case $\mathtt{m} \underset{\mathtt{M}}{>} \mathtt{n}$ of the case statement in ($\underset{\mathtt{P}}{+}$).

■ 2.6.4: Plus is Commutative

Our goal is: $\underset{\mathtt{p},\mathtt{q}\in\mathtt{P}}{\forall}\ \mathtt{p} \underset{\mathtt{P}}{+} \mathtt{q} = \mathtt{q} \underset{\mathtt{P}}{+} \mathtt{p}.$

The proof works by nested sequence induction and case distinction in the proof of the induction step. The proofs of the cases work by simplification. There the definition (+) is used as conditional rewrite rule several times, often without necessity to prove the condition, when it is already given as case assumption. An example is the case $\dot{p}_0 > \dot{q}_0$, where the first two cases of the definition (+) are used, both needing the condition $\dot{p}_0 > \dot{q}_0$.

Simplification in the proof of the case $\dot{p}_0 = -\dot{q}_0$ needs an explicit subproof of the condition $\dot{q}_0 = -\dot{p}_0$, which is done by simplification again. For this we use the case assumption as rewrite rule, such that we can apply (−−) afterwards. This rewriting of $\dot{p}_0$ to $-\dot{q}_0$ however is not really a 'simplification', since in most usual term orders the second term will not be smaller than the first one, so rewriting would not take place. The prover can avoid this problem by using a lemma for the subproof instead:

$$(=\ -)\quad \underset{m,n\in M}{\forall}\ m = -n \Longleftrightarrow n = -m\,.$$

This follows from (−−). Using it, the proof of the condition would simply be

$\vdots$

$\dot{q}_0 = -\dot{p}_0$

By (= −), this follows from the case assumption.

However introducing the lemma (= −) only shifts the simplification problem into the proof of the lemma.

■ 2.6.5: Plus is Associative

Our goal is: $\underset{p,q,r\in P}{\forall}\ \left(p \underset{P}{+} q\right) \underset{P}{+} r = p \underset{P}{+} \left(q \underset{P}{+} r\right).$

The proof consists of polynomial induction over all three variables. At each level the zero polynomial base case works by simplification. The remaining nontrivial subgoals are:

(+ assoc.V.IB⟨m⟩.IB⟨m⟩.IB⟨m⟩) $\left(\langle\dot{p}_0\rangle + \langle\dot{q}_0\rangle\right) + \langle\dot{r}_0\rangle = \langle\dot{p}_0\rangle + \left(\langle\dot{q}_0\rangle + \langle\dot{r}_0\rangle\right)$

$\vdots$

(+ assoc.V.IB⟨m⟩.IB⟨m⟩.IC) $\left(\langle\dot{p}_0\rangle + \langle\dot{q}_0\rangle\right) + \langle\dot{r}_0, \ddot{r}_0, \overline{r_0}\rangle = \langle\dot{p}_0\rangle + \left(\langle\dot{q}_0\rangle + \langle\dot{r}_0, \ddot{r}_0, \overline{r_0}\rangle\right)$

$\vdots$

(+ assoc.V.IB⟨m⟩.IC) $\underset{\bar{r}}{\forall}\ \left(\langle\dot{p}_0\rangle + \langle\dot{q}_0, \ddot{q}_0, \overline{q_0}\rangle\right) + \langle\bar{r}\rangle = \langle\dot{p}_0\rangle + \left(\langle\dot{q}_0, \ddot{q}_0, \overline{q_0}\rangle + \langle\bar{r}\rangle\right)$

$\vdots$

(+ assoc.V.IC) $\underset{\bar{q},\bar{r}}{\forall}\ \left(\langle\dot{p}_0, \ddot{p}_0, \overline{p_0}\rangle + \langle\bar{q}\rangle\right) + \langle\bar{r}\rangle = \langle\dot{p}_0, \ddot{p}_0, \overline{p_0}\rangle + \left(\langle\bar{q}\rangle + \langle\bar{r}\rangle\right)$

The last two of them are not ground and have to be decomposed with induction again. The proofs of the first ones and of the further generated ground subgoals all work by a two (sometimes three) level case distinction on all possible order configurations of the constants $\dot{p}_0$, $\dot{q}_0$ and $\dot{r}_0$. We only examine the shortest of the subproofs, the proof of (+ assoc.V.IB⟨m⟩.IB⟨m⟩.IB⟨m⟩), as we only want to show how nested case distinction works. The other subproofs will be more difficult, since there one has to use corresponding induction hypotheses, but even in the part we present the deepest occuring cell group nesting is already 26. Therefore it would already be good to present the subproofs in several hyperlinked notebooks.

The outermost case distinction is done on the order of $\dot{p}_0$ and $\dot{q}_0$. This is decided by taking the first polynomial sum with arguments matching the $\langle m,\ \bar{p}\rangle + \langle n,\ \bar{q}\rangle$ from the definition (+), which is $\langle \dot{p}_0 \rangle + \langle \dot{q}_0 \rangle$, and doing case distinction on the m and n .

The first case $\dot{p}_0 > \dot{q}_0$ has the goal

$$\langle \dot{p}_0,\ \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle = \langle \dot{p}_0 \rangle + (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle)\ .$$

It again triggers a proof by case distinction. Now the left hand side term of the equality matches the $\langle m,\ \bar{p}\rangle + \langle n,\ \bar{q}\rangle$, so we consider the order of $\dot{p}_0$ and $\dot{r}_0$. Here the first case has the assumption $\dot{p}_0 > \dot{r}_0$, and after simplification we get the goal

$$\dot{p}_0 \sim (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle) = \langle \dot{p}_0 \rangle + (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle)$$

A third level of case distinction is now done to decompose the term $\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle$. Now all four cases can be proved by simplification. The assumptions $\dot{p}_0 > \dot{q}_0$, $\dot{p}_0 > \dot{r}_0$ and the case assumption about the order of $\dot{q}_0$ and $\dot{r}_0$ are allowing to use (+) as rewrite rule by fulfilling the necessary conditions.

It is important to note that the polynomial prover must not only be able to prove conditions of useful equations, it also has to be able to use the given assumptions to disprove conditions of non−applicable equations. An example is the simplification of the term $\langle \dot{p}_0 \rangle + \langle \dot{q}_0 \rangle$ under the assumption $\dot{q}_0 > \dot{p}_0$. The ‘conditional simplifier’ has to find out which of the conditional rewrite rules in (+) can be used. For this the conditions are tested in the order they are given in (+). So the first step is to test whether the rule

$$(+.1)\qquad \underset{m,n,\bar{p},\bar{q}}{\forall}\quad m > n \implies \langle m,\ \bar{p}\rangle + \langle n,\ \bar{q}\rangle = m \sim (\langle \bar{p}\rangle + \langle n,\ \bar{q}\rangle)$$

can be applied. The ground instance of the condition is $\dot{p}_0 > \dot{q}_0$, the prover now must be able to apply antisymmetry of > to disprove this, such that it knows that this rule cannot be used.

Another strategy of using conditional equalities as rewrite rules would be to use only unconditional rewrite rules during simplification and to generate the necessary rules in advance. This would allow us to keep the simplifier small and efficient. In the example, the assumption $\dot{q}_0 > \dot{p}_0$ allows us to create the following ones from (+):

$$\underset{\bar{p},\bar{q}}{\forall}\quad \langle \dot{q}_0,\ \bar{p}\rangle + \langle \dot{p}_0,\ \bar{q}\rangle = \dot{q}_0 \sim (\langle \bar{p}\rangle + \langle \dot{p}_0,\ \bar{q}\rangle) \quad \text{and}$$

$$\underset{\bar{p},\bar{q}}{\forall}\quad \langle \dot{p}_0,\ \bar{p}\rangle + \langle \dot{q}_0,\ \bar{q}\rangle = \dot{q}_0 \sim (\langle \dot{p}_0,\ \bar{p}\rangle + \langle \bar{q}\rangle)\ .$$

Now the second formula matches $\langle \dot{p}_0 \rangle + \langle \dot{q}_0 \rangle$ and is used to rewrite it.

Using this approach we get the problem to ensure that we provide everything that could be needed by the following simplification. An example for this is the proof of the fourth case in the third level case distinction. Local assumptions are $\dot{p}_0 > \dot{q}_0$, $\dot{p}_0 > \dot{r}_0$ and (implicitely, since we are in the ‘otherwise’ case) $\dot{q}_0 \sim \dot{r}_0$ and $\dot{q}_0 \neq -\dot{r}_0$.

During simplification we need the condition $\dot{p}_0 > \dot{q}_0 \underset{M}{+} \dot{r}_0$ to hold. It follows from the assumptions by $(> \underset{M}{+})$. So we have to apply $(> \underset{M}{+})$ before simplification. Similarly there are lots of other monomial implications that have to be used to expand the knowledge base in two steps immediately after a new case assumption has been added. Some of them, like (> trans), $(\sim\ >)$, $(>\ \sim)$, $(+\ >)$, are used in the same way as $(> \underset{M}{+})$ in the first step to generate formulae that match necessary conditions. In the second step others like $(\underset{M}{+}$ comm), $(\underset{M}{+}$ assoc) or $(+)$ are used to generate unconditional rewrite rules from the case assumptions and the formulae found in the first step.

During the further proof we get into the situation where we would like to use lemma $(\underset{M}{+}\ >.2)$, as shown in the description of the proof of ‘plus is closed’. A human prover could use backwards reasoning to derive that he has to apply $(\underset{M}{+}$ comm) first for being able to use $(\underset{M}{+}\ >)$ instead:

$\vdots$

to $\left\langle \dot{q}_0 \underset{M}{+} \dot{r}_0 \right\rangle \smile \left(\left\langle \dot{p}_0 \right\rangle + \langle\,\rangle \right)$ by (+),

which we can use because

$\dot{q}_0 \underset{M}{+} \dot{r}_0 > \dot{p}_0$.

We simplify

$\dot{q}_0 \underset{M}{+} \dot{r}_0$

to $\dot{r}_0 \underset{M}{+} \dot{q}_0$ by $(\underset{M}{+}$ comm)

and prove the equivalent

$\dot{r}_0 \underset{M}{+} \dot{q}_0 > \dot{p}_0$.

By $(\underset{M}{+}\ >)$, this follows from the case assumptions.

In the above ‘knowledge expanding’ approach this could be simulated by using $(\underset{M}{+}$ comm) not only during the second step, but also during the first one. This means in our situation to add $\dot{q}_0 \underset{M}{+} \dot{r}_0 > \dot{p}_0$ to the knowledge base when it already contains $\dot{r}_0 \underset{M}{+} \dot{q}_0 > \dot{p}_0$.

Proceeding like this has the disadvantage that we use forward reasoning to derive everything deriveable, which could be very much, while goal–driven backwards reasoning would be much more efficient by only generating what is really used.

This third level of case distinction, introduced in the case $\dot{p}_0 > \dot{r}_0$ of proving

$$\left\langle \dot{p}_0 , \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle = \left\langle \dot{p}_0 \right\rangle + \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle \right),$$

could be replaced by using the lemmata $(\smile\ =\ +)$, $(\smile \in \mathrm{P})$ and (>ldm[+]). We show part of the corresponding proof:

Case $\dot{p}_0 > \dot{r}_0$:

We prove by simplification.

We simplify the left hand side

to $\dot{p}_0 - \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle\right)$ by (+)

and the right hand side

to $\dot{p}_0 - \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle\right)$ by ($-$ = +),

which we can use because

$\dot{p}_0 - \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle\right) \in \mathtt{P}$.

By ($- \in \mathtt{P}$), this follows from

$\dot{p}_0 > \mathtt{ldm}\left[\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle\right] \lor \left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle = \langle\rangle$.

We assume

$\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle \neq \langle\rangle$

and prove

$\dot{p}_0 > \mathtt{ldm}\left[\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle\right]$.

By (>ldm[+]), this follows from

$\dot{p}_0 > \mathtt{ldm}\left[\left\langle \dot{q}_0 \right\rangle\right] \land \dot{p}_0 > \mathtt{ldm}\left[\left\langle \dot{r}_0 \right\rangle\right] \land \left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle \neq \langle\rangle$.

$\vdots$

At this point it is clear that the proof works, by using the available case assumptions $\dot{p}_0 > \dot{q}_0$ and $\dot{p}_0 > \dot{r}_0$. This alternative way of proving has the advantage that it is more general, e.g. it works in the same way later in the proof where we have the same case assumptions but the modified goal

$$\left\langle \dot{p}_0, \dot{q}_0 \right\rangle + \left\langle \dot{r}_0, \ddot{r}_0, \overline{r_0} \right\rangle = \left\langle \dot{p}_0 \right\rangle + \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \dot{r}_0, \ddot{r}_0, \overline{r_0} \right\rangle\right)$$

Here a third level case distinction on the order of $\dot{r}_0$ and $\dot{q}_0$ does not avoid using the lemmata any more: In the case $\dot{r}_0 > \dot{q}_0$ the goal simplifies to

$$\dot{p}_0 - \left(\dot{r}_0 - \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \ddot{r}_0, \overline{r_0} \right\rangle\right)\right) = \left\langle \dot{p}_0 \right\rangle + \left(\dot{r}_0 - \left(\left\langle \dot{q}_0 \right\rangle + \left\langle \ddot{r}_0, \overline{r_0} \right\rangle\right)\right),$$

which again has to be proved by the three lemmata. So it is better to immediately use the lemmata and to avoid the third case distinction at all. However it cannot be known in advance whether applying lemmata creates a shorter proof than using case distinction, so the prover should always go into case distinction to be on the safe side.

The proofs of the other three cases of the outermost case distinction on the order of $\dot{p}_0$ and $\dot{q}_0$ again work by case distinction. The corresponding simplified goals are

Case $\dot{q}_0 > \dot{p}_0$:

$\vdots$

$$\langle \dot{q}_0, \dot{p}_0 \rangle + \langle \dot{r}_0 \rangle = \langle \dot{p}_0 \rangle + (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle)$$

Case $\dot{p}_0 = -\dot{q}_0$:

$\vdots$

$$\langle \dot{r}_0 \rangle = \langle \dot{p}_0 \rangle + (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle)$$

Case otherwise:

$\vdots$

$$\langle \dot{p}_0 \underset{M}{+} \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle = \langle \dot{p}_0 \rangle + (\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle)$$

The corresponding case distinction subproofs work by comparing $\dot{q}_0$ and $\dot{r}_0$. In the case $\dot{q}_0 > \dot{p}_0$ this is a case distinction on the value of $\langle \dot{q}_0, \dot{p}_0 \rangle + \langle \dot{r}_0 \rangle$ and turns out to set the value of $\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle$ as well. In the case $\dot{p}_0 = -\dot{q}_0$ the second argument of the outermost plus does not match the definition needing tuples as arguments, so case distinction is done on the value of the inner sum $\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle$. The third case shows two possible case distinctions: either on the value of the left hand side sum or on $\langle \dot{q}_0 \rangle + \langle \dot{r}_0 \rangle$ again. Here we choose the second one, since the involved arguments are simpler. This avoids having to add more monomial assumptions, such as

$$\underset{l,m,n}{\forall} \; l \sim m \wedge l \neq -m \wedge l \underset{M}{+} m > n \Longrightarrow m > n .$$

In the proof of case $\dot{p}_0 > \dot{q}_0$, the equational assumption of the subcase $\dot{q}_0 = -\dot{r}_0$ was only used to provide the condition for the rewrite rule corresponding to the case $m = -n$ of (+). In the subcase $\dot{q}_0 = -\dot{r}_0$ of the third case, $\dot{p}_0 = -\dot{q}_0$, the equational case assumptions are now used as rewrite rules, similarly to how it was done in the proof of commutativity of plus. Again rewriting of $\dot{p}_0$ to the larger $-\dot{q}_0$ and $\dot{q}_0$ to $-\dot{r}_0$ cannot be avoided.

The 'otherwise' cases of the subproofs are the places where the last case of (+) has to be used as rewrite rule. There the condition is an inequality. An example for this is the last subcase $\dot{q}_0 \neq -\dot{r}_0$ in the proof of the case $\dot{p}_0 = -\dot{q}_0$. There simplification contains the following application of (+):

$\vdots$

to $\langle \dot{p}_0 \rangle + \langle \dot{q}_0 \underset{M}{+} \dot{r}_0 \rangle$ by ($\smile$),

to $(\dot{p}_0 \underset{M}{+} (\dot{q}_0 \underset{M}{+} \dot{r}_0)) \smile (\langle\rangle + \langle\rangle)$ by (+),

which we can use because

$\dot{\mathtt{p}}_0 \neq -\left(\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} \dot{\mathtt{r}}_0\right)$.

$\vdots$

We prove an inequality by contradiction:

We assume

$\dot{\mathtt{p}}_0 = -\left(\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} \dot{\mathtt{r}}_0\right)$

and prove a contradiction.

We simplify

and assume the equivalent

$-\dot{\mathtt{q}}_0 = -\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} -\dot{\mathtt{r}}_0$.

This contradicts (no $\underset{\mathtt{M}}{+0}$) .

The simplification step shown here is in fact composed of simplifying two different subterms. We collect these two term simplifications into one by saying:

We simplify

$\dot{\mathtt{p}}_0$

to $-\dot{\mathtt{q}}_0$ by the case assumption, and

$-\left(\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} \dot{\mathtt{r}}_0\right)$

to $-\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} -\dot{\mathtt{r}}_0$ by ($-\underset{\mathtt{M}}{+}$)

Again, the rewriting performed here does not really make the term ‘simpler’ but aims at reaching a formula that matches the negation of (no $\underset{\mathtt{M}}{+0}$). Guessing that the formula (no $\underset{\mathtt{M}}{+0}$) will provide the necessary contradiction is nontrivial, but without this guess we can only use forward reasoning for this proof: We have to start from the constradiction assumption and then search the space of deriveable formulae until we are lucky to find a formula that matches the negation of another formula in the knowledge base.

Several times we also want to use the lemma ($-\mathtt{m} \underset{\mathtt{M}}{+} \left(\mathtt{m} \underset{\mathtt{M}}{+}\right)$) for simplifying. Often it is again necessary to to backward reasoning to bring the formula in a form where this lemma can be applied, as for example in the following simplification steps, with case assumption $\dot{\mathtt{p}}_0 = -\dot{\mathtt{q}}_0$:

$\vdots$

to $\left\langle \dot{\mathtt{p}}_0 \underset{\mathtt{M}}{+} \left(\dot{\mathtt{q}}_0 \underset{\mathtt{M}}{+} \dot{\mathtt{r}}_0\right) \right\rangle$ by ($\smile$), to $\left\langle -\dot{q}_0 \underset{M}{+} \left(\dot{q}_0 \underset{M}{+} \dot{r}_0\right)\right\rangle$ by the case assumption,

to $\left\langle \dot{r}_0 \right\rangle$ by $(-m \underset{M}{+} (m \underset{M}{+}))$

This simplification also shows that $(-m \underset{M}{+} (m \underset{M}{+}))$ is needed because we do not have a zero monomial $\underset{M}{0}$. Otherwise we could use a more natural

$\vdots$

to $\left\langle -\dot{q}_0 \underset{M}{+} \left(\dot{q}_0 \underset{M}{+} \dot{r}_0\right)\right\rangle$ by the case assumption,

to $\left\langle \left(-\dot{q}_0 \underset{M}{+} \dot{q}_0\right) \underset{M}{+} \dot{r}_0\right\rangle$ by $(\underset{M}{+}$ assoc),

to $\left\langle \underset{M}{0} \underset{M}{+} \dot{r}_0\right\rangle$ by $(-\underset{M}{+})$,

to $\left\langle \dot{r}_0 \right\rangle$ by $(\underset{M}{0} \underset{M}{+})$

In our situation we cannot apply $(\underset{M}{+}$ assoc), because its conditions ensure that we do not get zero subterms. Furthermore we do not have the assumptions $(-\underset{M}{+})$ and $(\underset{M}{0} \underset{M}{+})$ speaking about $\underset{M}{0}$.

In the last cases of the proof we know that all the three monomial constants are equivalent with respect to the order. In the special case of monomials represented as tuples of coefficient and power product term this means that they have only different coefficients, but the same power product. A special proof situation appears when $\dot{p}_0 \neq -\dot{q}_0$ and $\dot{q}_0 \neq -\dot{r}_0$. There the simplified subgoal is

$$\left\langle \dot{p}_0 \underset{M}{+} \dot{q}_0 \right\rangle + \left\langle \dot{r}_0 \right\rangle = \left\langle \dot{p}_0 \right\rangle + \left\langle \dot{q}_0 \underset{M}{+} \dot{r}_0 \right\rangle .$$

Case distinction on the order of $\dot{p}_0 \underset{M}{+} \dot{q}_0$ and $\dot{r}_0$ now only has to examine the case $\dot{p}_0 \underset{M}{+} \dot{q}_0 = -\dot{r}_0$ or its negation. The other two cases are not shown. This is a way of expressing that these hidden cases introduce contradictions, since by $(\sim \underset{M}{+})$ we know that $\dot{p}_0 \underset{M}{+} \dot{q}_0 \sim \dot{r}_0$. It would also be possible to present all the cases and to show the proofs of the contradictions explicitly for the first two of them. The case $\dot{p}_0 \underset{M}{+} \dot{q}_0 = -\dot{r}_0$ once again needs $(-m \underset{M}{+} (m \underset{M}{+}))$, because we have to avoid monomial zero, while finally in the case $\dot{p}_0 \underset{M}{+} \dot{q}_0 = -\dot{r}_0$ the conditions for monomial associativity are fulfilled and $(\underset{M}{+}$ assoc) is used for simplification.

■ 2.7: Proofs using Multiplication

■ 2.7.1: One is Neutral Element

Our goal is: $\underset{p \in P}{\forall} \; p \underset{P}{*} \underset{P}{1} = p.$

To prove this, we need the monomial one to be the neutral element of monomial times (neutral $\underset{M}{1}$). We also need the fact that monomial one can be multiplied with all monomials ($\underset{M}{1}$ not zerodiv) for using ($\underset{P}{*}$) as conditional rewrite rule.

In contrast to the proof of closure of $\underset{P}{*}$ we now take the definition of times exactly as it is given in the functor. However, this proof would also work with the alternative definition using $\underset{P}{+}$ instead of prepending to add the head element of the sum, but then the proof would need polynomial induction instead of sequence induction: During the simplification proof of the induction conclusion, prepending works immediately to create the polynomial tuple $\left\langle \dot{p}_0 \underset{M}{*} \underset{M}{1}, \overline{p_0} \right\rangle$ from the term $\left(\dot{p}_0 \underset{M}{*} \underset{M}{1} \right) \smile \langle \overline{p_0} \rangle$. Using the definition with $\underset{P}{+}$ instead, we could not further simplify the term $\left(\dot{p}_0 \underset{M}{*} \underset{M}{1} \right) \underset{P}{+} \langle \overline{p_0} \rangle$ here. The corresponding term in polynomial induction would be $\left(\dot{p}_0 \underset{M}{*} \underset{M}{1} \right) \underset{P}{+} \left\langle \ddot{p}_0, \overline{p_0} \right\rangle$, where the definition of $\underset{P}{+}$ could be applied, because the term matches the left hand side of ($\underset{P}{+}$).

■ 2.7.2: Times is Commutative

Our goal is: $\underset{p, q \in P}{\forall} \; p \underset{P}{*} q = q \underset{P}{*} p .$

We need the monomial properties of commutativity of monomial times ($\underset{M}{*}$ comm) and symmetry of the `zero-devisor` predicate. We use the definition of polynomial times from the functor, but the proof would also work with the definition using $\underset{P}{+}$ instead of prepending, because the definitions of these functions are never applied. Additionally we need the lemmata

$$(* \langle\rangle \text{ distrib}) \quad \underset{m, p, \bar{q}}{\forall} \; \langle m, \bar{q} \rangle \in P \Longrightarrow p * \langle m, \bar{q} \rangle = p * \langle m \rangle + p * \langle \bar{q} \rangle ,$$

$$(\langle\rangle * \text{ distrib}) \quad \underset{m, p, \bar{q}}{\forall} \; \langle m, \bar{q} \rangle \in P \Longrightarrow \langle m, \bar{q} \rangle * p = \langle m \rangle * p + \langle \bar{q} \rangle * p .$$

We use sequence induction on both variables. The goal ($*$ comm.V.IC.IC) is then proved by case distinction according to the cases in the definition of times. Both cases work by simplification. It is very important to apply the induction hypotheses ($*$ comm.V.IH) and ($*$ comm.V.IC.IH) as soon as possible, because otherwise they could not be used any more, which could cause the proof to fail. In our case the first step of the last simplification gives

$$\text{to} \quad \left\langle \dot{q}_0 \underset{M}{*} \dot{p}_0 \right\rangle \smile \left(\left\langle \dot{q}_0 \right\rangle * \langle \bar{p}_0 \rangle + \langle \overline{q_0} \rangle * \left\langle \dot{p}_0, \overline{p_0} \right\rangle \right) \quad \text{by } (*) .$$

If we decomposed the subterm $\left\langle \dot{p}_0, \overline{p_0} \right\rangle$ immediately by ($* \langle\rangle$ distrib) we would not be able to apply

$(\star\ \text{comm.V.IC.IH})\quad \left\langle \overset{\bullet}{p}_0\,,\ \overline{p_0}\right\rangle \star \langle \overline{q_0} \rangle = \langle \overline{q_0} \rangle \star \left\langle \overset{\bullet}{p}_0\,,\ \overline{p_0}\right\rangle$

any more and the proof would fail.

■ 2.7.3: Proofs not examined yet

The following theorems and lemmata still have to be proved for $(\mathtt{P},\ \underset{P}{0},\ \underset{P}{1},\ \underset{P}{+},\ \underset{P}{-},\ \underset{P}{\star})$ to be a commutative ring with identity:

– the lemmata ($\star \langle\rangle$ distrib) and ($\langle\rangle \star$ distrib) used in ‘times is commutative’

– associativity of polynomial times: $\underset{p,q,r \in P}{\forall} \left(p \underset{P}{\star} q\right) \underset{P}{\star} r = p \underset{P}{\star} \left(q \underset{P}{\star} r\right)$

– left distributivity of times over plus: $\underset{p,q,r \in P}{\forall} p \underset{P}{\star} \left(q \underset{P}{+} r\right) = p \underset{P}{\star} q \underset{P}{+} p \underset{P}{\star} r$

– right distributivity of times over plus: $\underset{p,q,r \in P}{\forall} \left(p \underset{P}{+} q\right) \underset{P}{\star} r = p \underset{P}{\star} r \underset{P}{+} q \underset{P}{\star} r$