Polynomial Terms and Sparse Horner Normal Form

David M. Russinoff

July 20, 2017

This note documents an ACL2 [1] formalization of an efficient method of establishing the equality of two integer polynomials in several variables, adapted from a similar Coq implementation by Gregoire and Mahboubi [2]. The associated proof script resides in the directory `books/projects/shnf` of the ACL2 repository. The motivation for this exercise is an analysis of the group operation defined on an elliptic curve.[3] The application of this operation to two points on the curve involves compositions of rational functions of the coordinates of the two points. A proof of associativity amounts to establishing an identity involving several further compositions of these functions. In principle, this could be achieved by performing the compositions, reducing to a polynomial equation by cross-multiplication, expanding the polynomials to sums of monomials, and canceling terms. This approach, however, is grossly impractical—the number of resulting monomials have been found to exceed 10^{25}!

The solution to this problem is an efficiently computable representation of polynomials, known as *sparse Horner normal form*. Following [2], we define an evaluation function on these normal forms and prove equality between the value of a polynomial for a given set of variable assignments and that of its representation. Thus, we may establish the equivalence of two polynomials by observing that their representations coincide.

Of course, the utility of this procedure rests on the property of completeness: equivalent polynomials necessarily produce the same representation. According to the authors of the Coq proof, which does not address this property, it cannot even be stated within their formal framework. We shall present a constructive proof of this result that we have formalized in ACL2, based on a function of two polynomials that computes a list of variable assignments for which the values of the polynomials differ, whenever such a list exists.

Our implementation is based on *S-expressions*. For this purpose, an S-expression is an integer, a symbol, or an ordered list $s = (s_0\ s_1\ \ldots\ s_n)$ of S-expressions. In the last case, the s_i are the *members* of s, $head(s) = s_0$, and for $k \in \mathbb{N}$, we define $s^{(k)} = (s_k\ \ldots\ s_n)$.

First, we define an encoding of polynomial expressions in the natural way as S-expressions constructed from variable symbols, integers, and the symbols `+`, `-`, `*`, and `EXPT`, which represent the basic arithmetic operations.

Definition 1 *Let V be a list of distinct symbols. A polynomial term over V is any of the following S-expressions:*

(a) An integer;

(b) A member of V;

(c) A list `(-` x`)`, where x is a polynomial over V;

(d) A list `(`op x y`)`, where x and y are polynomials over V and $op \in \{$`+`, `-`, `*`$\}$;

(d) A list `(EXPT` x n`)`, where x is polynomial over V and $n \in \mathbb{N}$.

$\mathcal{T}(V)$ will denote the set of all polynomial terms over V.

For example, the polynomial $-z + x^3(z + x - 3y)$ is encoded as

```
(+ (- Z) (* (EXPT X 3) (- (+ Z X) (* 3 Y)))).
```

Informally, we may use a polynomial expression to refer to the S-expression by which it is encoded.

A polynomial term is evaluated in the context of an assignment of values to variables:

Definition 2 *An alist for a list of distinct symbols* $(v_0 \ldots v_{k-1})$ *is a list of the form*

$$A = ((v_0\ \ n_0) \ldots (v_{k-1}\ \ n_{k-1})),$$

where $n_i \in \mathbb{Z}$ *for* $0 \leq i < k$*. We shall say that* A *associates* v_i *with* n_i*.*

Definition 3 *Let* A *be an alist for a list of distinct symbols* V *and let* $q \in \mathcal{T}(V)$*.*

(a) If $q \in \mathbb{Z}$*, then* $evalp(q, A) = q$*;*

(b) If q *is a member of* V *and* A *associates* q *with* n*, then* $evalp(q, A) = n$*;*

(c) If $q =$ `(- r)`*, then* $evalp(q, A) = -evalp(r, A)$*;*

(d) If $q =$ `(+ r s)`*, then* $evalp(q, A) = evalp(r, A) + evalp(s, A)$*;*

(e) If $q =$ `(- r s)`*, then* $evalp(q, A) = evalp(r, A) - evalp(s, A)$*;*

(f) If $q =$ `(* r s)`*, then* $evalp(q, A) = evalp(r, A) \cdot evalp(s, A)$*;*

(g) If $q =$ `(EXPT r n)`*, then* $evalp(q, A) = evalp(r, A)^n$*.*

We shall define an encoding of polynomial terms as list structures of another sort, constructed from integers and the two symbols `POP` and `POW`:

Definition 4 *A sparse Horner form (SHF) is any of the following S-expressions:*

(a) An integer;

(b) A list `(POP i p)`*, where* $i \in \mathbb{N}$ *and* p *is a SHF;*

(c) A list `(POW i p q)`*, where* $i \in \mathbb{N}$ *and* p *and* q *are SHFs.*

Our objective is to define a function *norm* that computes a SHF encoding of a polynomial term f with respect to a variable ordering $V = (v_0 \ldots v_{k-1})$, and a function *evalh* that evaluates a SHF with respect to a corresponding list of values $N = (n_0 \ldots n_{k-1})$, such that

$$evalh(norm(f, V), N) = evalp(f, A),$$

where

$$A = ((v_0\ \ n_0) \ldots (v_{k-1}\ \ n_{k-1})).$$

One possible approach to the definition of $norm(f, V)$ is as follows:

(1) If f is an integer constant, then $norm(f, V) = f$.

(2) Suppose v_0 occurs in f. Find polynomials g and h such that $f = v_0^i \cdot g + h$, g is not divisible by v_0, and v_0 does not occur in h. Then

$$norm(f, V) = \texttt{(POW}\ i\ p\ q\texttt{)},$$

where $p = norm(g, V)$ and $q = norm(h, V^{(1)})$.

(3) Suppose v_0 does not occur in f. Let v_i be the first variable in V that does occur in f. Then

$$norm(f, V) = \texttt{(POP}\ i\ p\texttt{)},$$

where $p = norm(f, V^{(i)})$.

For example, consider the polynomial

$$4x^4y^2 + 3x^3 + 2z^4 + 5$$

with variable ordering `(x y z)`. Rewriting the polynomial as

$$x^3(4xy^2 + 3) + (2z^4 + 5),$$

we find that the normalization is

$$\texttt{(POW 3 } p\ q\texttt{)},$$

where

$$p = norm(4xy^2 + 3, \texttt{(}x\ \ y\ \ z\texttt{)})$$

and

$$q = norm(2z^4 + 5, \texttt{(}y\ \ z\texttt{)}).$$

Continuing recursively, we arrive at the final result:

```
(POW 3 (POW 1 (POP 1 (POW 2 4 0)) 3)
       (POP 1 (POW 4 2 5))).
```

The evaluation of SHFs is defined as follows:

Definition 5 *Let h be a SHF and let N be a list of integers.*

(a) If $h \in \mathbb{Z}$, then $evalh(h, N) = h$.

(b) If $h =$ `(POP i p)`*, then then $evalh(h, N) = evalh(p, N^{(i)})$.*

(c) If $h =$ `(POW i p q)` *and $head(N) = n$, then $evalh(h, N) = n^i evalh(p, N) + evalh(q, N^{(1)})$.*

(d) If $h =$ `(POW i p q)` *and $N =$* `()`*, then $evalh(h, N) = 0$.*

It may be instructive to check that the value of the SHF in the above example for the list of values (1 2 3), for example, and the value of the represented polynomial for the corresponding alist, are both 207.

It is not difficult to see that a SHF generated by this normalization procedure conforms to the following restriction:

Definition 6 *A sparse Horner normal form (SHNF) is any of the following SHFs:*

(a) An integer;

(b) `(POP i p)`*, where $i > 0$, and p is a SHNF of the form* `(POW i q r)`*;*

(c) `(POW i p q)`*, where $i > 0$, and p and q are SHNFs, and p is not of the form* `(POW j r 0)`*.*

$\mathcal{H}$ denotes the set of all SHNFs.

Unfortunately, this top-down procedure is impractical because of the general difficulty of constructing the polynomials g and h in Case (2). Our preferred definition will provide a more efficient bottom-up procedure. We begin with the functions

$$pop : \mathbb{Z} \times \mathcal{H} \to \mathcal{H}$$

and

$$pow : \mathbb{Z} \times \mathcal{H} \times \mathcal{H} \to \mathcal{H},$$

which normalize the SHFs `(POP i p)` and `(POW i p q)`, respectively.

Definition 7 *Let $i \in \mathbb{N}$ and $p \in \mathcal{H}$.*

(a) If $i = 0$ or $p \in \mathbb{Z}$, then $pop(i, p) = p$.

(b) If $p =$ `(POP j q)`*, then $pop(i, p) =$* `(POP` $i + j$ `q)`*.*

(c) Otherwise, $pop(i, p) =$ `(POP i p)`*.*

Definition 8 *Let $i \in \mathbb{N} - \{0\}$, $p \in \mathcal{H}$, and $q \in \mathcal{H}$.*

(a) If $p = 0$, then $pow(i, p, q) = pop(1, q)$.

(b) If $p =$ `(POW j r 0)`, then $pow(i, p, q) =$ `(POW` $i + j$ `r q)`.

(c) Otherwise, $pow(i, p, q) =$ `(POW i p q)`.

The following properties of these functions are immediate consequences of Definitions 5, 7, and 8:

Lemma 1 *If $i \in \mathbb{N}$, $p \in \mathcal{H}$, and N is a list of integers, then $pop(i, p) \in \mathcal{H}$ and*

$$evalh(pop(i, p), N) = evalh(\texttt{(POP } i \texttt{ } p\texttt{)}, N).$$

Lemma 2 *If $i \in \mathbb{N} - \{0\}$, $p \in \mathcal{H}$, $q \in \mathcal{H}$, and N is a list of integers, then $pow(i, p, q) \in \mathcal{H}$ and*

$$evalh(pow(i, p, q), N) = evalh(\texttt{(POW } i \texttt{ } p \texttt{ } q\texttt{)}, N).$$

Suppose we have have computed the SHNFs for polynomial terms x and y. The following function then computes the SHNF for the term `(+ x y)`.

Definition 9 *If $x \in \mathcal{H}$ and $y \in \mathcal{H}$, then $x \oplus y$ is defined as follows:*

(1) If $x \in \mathbb{Z}$, then

(a) $y \in \mathbb{Z} \Rightarrow x \oplus y = x + y$.

(b) $y =$ `(POP i p)` $\Rightarrow x \oplus y =$ `(POP i` $x \oplus p$`)`.

(c) $y =$ `(POW i p q)` $\Rightarrow x \oplus y =$ `(POW i p` $x \oplus q$`)`.

(2) If $y \in \mathbb{Z}$, then $x \oplus y = y \oplus x$.

(3) If $x =$ `(POP i p)` and $y =$ `(POP j q)`, then

(a) $i = j \Rightarrow x \oplus y = pop(i, p \oplus q)$.

(b) $i > j \Rightarrow x \oplus y = pop(j,$ `(POP` $i - j$ `p)` $\oplus q)$.

(c) $i < j \Rightarrow x \oplus y = pop(i,$ `(POP` $j - i$ `q)` $\oplus p)$.

(4) If $x =$ `(POP i p)` and $y =$ `(POW j q r)`, then

(a) $i = 1 \Rightarrow x \oplus y =$ `(POW j q` $r \oplus p$`)`.

(b) $i > 1 \Rightarrow x \oplus y =$ `(POW j q` $r \oplus$ `(POP` $i - 1$ `p))`.

(5) If $y =$ `(POP i p)` and $x =$ `(POW j q r)`, then $x \oplus y = y \oplus x$.

(6) If $x =$ `(POW i p q)` and $y =$ `(POW j r s)`, then

(a) $i = j \Rightarrow x \oplus y = pow(i, p \oplus r, q \oplus s)$.

(b) $i > j \Rightarrow x \oplus y = pow(j,$ `(POW` $i - j$ `p 0)` $\oplus r, q \oplus s)$.

(c) $i < j \Rightarrow x \oplus y = pow(i,$ `(POW` $j - i$ `r 0)` $\oplus p, s \oplus q)$.

The following is easily proved by induction, as are the analogous properties of negation, multiplication, and exponentiation, as defined below.

Lemma 3 *If $x \in \mathcal{H}$, $y \in \mathcal{H}$, and N is a list of integers, then $x \oplus y \in \mathcal{H}$ and*

$$evalh(x \oplus y, N) = evalh(x, N) + evalh(y, N).$$

Definition 10 *If $x \in \mathcal{H}$, then $\ominus x$ is defined as follows:*

(1) *If* $x \in \mathbb{Z}$, *then* $\ominus x = -x$.

(2) *If* $x =$ `(POP i p)`, *then* $\ominus x =$ `(POP i` $\ominus p$`)`.

(3) *If* $x =$ `(POW i p q)`, *then* $\ominus x =$ `(POW i` $\ominus p$ $\ominus q$`)`.

Lemma 4 *If* $x \in \mathcal{H}$ *and* N *is a list of integers, then* $\ominus x \in \mathcal{H}$ *and*

$$evalh(\ominus x, N) = -evalh(x, N).$$

Definition 11 *If* $x \in \mathcal{H}$ *and* $y \in \mathcal{H}$, *then* $x \otimes y$ *is defined as follows:*

(1) *If* $x \in \mathbb{Z}$, *then*

- *(a)* $y \in \mathbb{Z} \Rightarrow x \otimes y = xy$.
- *(b)* $y =$ `(POP i p)` $\Rightarrow x \otimes y = pop(i, x \otimes p)$.
- *(c)* $y =$ `(POW i p q)` $\Rightarrow x \otimes y = pow(i, x \otimes p, x \otimes q)$.

(2) *If* $y \in \mathbb{Z}$, *then* $x \otimes y = y \otimes x$.

(3) *If* $x =$ `(POP i p)` *and* $y =$ `(POP j q)`, *then*

- *(a)* $i = j \Rightarrow x \otimes y = pop(i, p \otimes q)$.
- *(b)* $i > j \Rightarrow x \otimes y = pop(j,$ `(POP` $i - j$ `p)` $\otimes q)$.
- *(c)* $i < j \Rightarrow x \otimes y = pop(i,$ `(POP` $j - i$ `q)` $\otimes p)$.

(4) *If* $x =$ `(POP i p)` *and* $y =$ `(POW j q r)`, *then*

- *(a)* $i = 1 \Rightarrow x \otimes y =$ `(POW j` $x \otimes q$ $p \otimes r$`)`.
- *(b)* $i > 1 \Rightarrow x \otimes y =$ `(POW j` $x \otimes q$ `(POP` $i - 1$ `p)` $\otimes r$`)`.

(5) *If* $y =$ `(POP i p)` *and* $x =$ `(POW j q r)`, *then* $x \otimes y = y \otimes x$.

(6) *If* $x =$ `(POW i p q)` *and* $y =$ `(POW j r s)`, *then*

$$x \otimes y = (pow(i + j, p \otimes r, q \otimes s) \oplus pow(i, p \otimes pop(1, s), 0)) \oplus pow(i, r \otimes pop(1, q), 0).$$

Lemma 5 *If* $x \in \mathcal{H}$, $y \in \mathcal{H}$, *and* N *is a list of integers, then* $x \otimes y \in \mathcal{H}$ *and*

$$evalh(x \otimes y, N) = evalh(x, N) \cdot evalh(y, N).$$

Definition 12 *If* $x \in \mathcal{H}$ *and* $k \in \mathbb{N}$, *then*

$$x^k = \begin{cases} 1 & \text{if } k = 0 \\ x \otimes x^{k-1} & \text{if } k > 0. \end{cases}$$

Lemma 6 *If* $x \in \mathcal{H}$, $k \in \mathbb{N}$, *and* N *is a list of integers, then* $x^k \in \mathcal{H}$ *and*

$$evalh(x^k, N) = evalh(x, N)^k.$$

We can now define the normalization procedure:

Definition 13 *Let* $x \in \mathcal{T}(V)$, *where* $V =$ `(`$v_0 \ldots v_{k-1}$`)` *is a list of distinct symbols.*

(a) $x \in \mathbb{Z} \Rightarrow norm(x, V) = x$.
(b) $x = v_i$, $0 \le i < k \Rightarrow norm(x, V) = pop(i,$ `(POW 1 1 0)`$)$.
(c) $x =$ `(-` y`)` $\Rightarrow norm(x, V) = \ominus norm(y, V)$.
(d) $x =$ `(+` y z`)` $\Rightarrow norm(x, V) = norm(y, V) \oplus norm(z, V)$.
(e) $x =$ `(-` y z`)` $\Rightarrow norm(x, V) = norm(y, V) \oplus (\ominus norm(z, V))$.
(f) $x =$ `(*` y z`)` $\Rightarrow norm(x, V) = norm(y, V) \otimes norm(z, V)$.
(g) $x =$ `(EXPT` y k`)` $\Rightarrow norm(x, V) = norm(y, V)^k$.

The reader may wish to check that the SHNF for the polynomial $-z+x^3(z+x-3y)$ with respect to the variable list `(x y z)` is once again

```
(POW 3 (POW 1 (POP 1 (POW 2 4 0)) 3)
       (POP 1 (POW 4 2 5))).
```

Lemma 7 *Let $f \in \mathcal{T}(V)$, where $V =$ `(`$v_0 \ldots v_{k-1}$`)` is a list of distinct symbols. Let $N =$ `(`$n_0 \ldots n_{\ell-1}$`)` be a list of integers with $\ell \geq k$ and*

$$A = \texttt{((}v_0\ \ n_0\texttt{)} \ldots \texttt{(}v_{k-1}\ \ n_{k-1}\texttt{))},$$

Then $norm(f, V) \in \mathcal{H}$ and

$$evalh(norm(f, V), N) = evalp(f, A).$$

PROOF: The case $f = v_i$ follows from Definitions 3 and 5 and Lemma 1; the other cases follow from Definitions 3 and 5, induction, and Lemmas 3, 4, 5, and 6. □

Lemma 7 implies that if two polynomials produce the same normal form, then they are equivalent. The converse follows from the next two lemmas.

Lemma 8 *Let $x \in \mathcal{H}$. If $x \neq 0$, then there exists a list of integers N such that $evalh(x, N) \neq 0$.*

PROOF: We shall prove, by induction on the structure of x, the following stronger statement: If $x \in \mathcal{H}$, $x \neq 0$, and $y \in \mathcal{H}$, then there exists a list of integers N such that $evalh(x, N) \neq 0$ and if $x =$ `(POW i p q)`, then $head(N) > 0$ and $|evalh(x, N)| > |evalh(y, N^{(1)})$.

Case 1: $x \in \mathbb{Z}$.
For any N, $evalh(x, N) = x \neq 0$.

Case 2: $x =$ `(POP i p)`.
By induction, there exists a list M such that $evalh(p, M) \neq 0$. We need only choose N so that $N^{(i)} = M$.

Case 3: $x =$ `(POW i p q)`, where $p \in \mathbb{Z}$ or $p =$ `(POP j r)`.
By induction, there exists a list N such that $evalh(p, N) \neq 0$. Since this value is independent of $head(N)$, we may choose

$$head(N) = n > |evalh(q, N^{(1)})| + |evalh(y, N^{(1)})|,$$

which implies

$$|evalh(x, N)| = |n^i \cdot evalh(p, N) + evalh(q, N^{(1)})| \geq n - |evalh(q, N^{(1)})| > |evalh(y, N^{(1)})|.$$

Case 4: $x =$ `(POW i p q)`, where $p =$ `(POW j r s)`.
By induction, we may choose N so that $head(N) = n > 0$ and

$$|evalh(p, N)| > |evalh((q \otimes q) \oplus (y \otimes y), N^{(1)})|.$$

It follows that

$$\begin{aligned}
|evalh(x, N)| &= |n^i \cdot evalh(p, N) + evalh(q, N^{(i)})| \\
&\geq |evalh(p, N)| - |evalh(q, N^{(1)})| \\
&> |evalh((q \otimes q) \oplus (y \otimes y), N^{(1)})| - |evalh(q, N^{(1)})| \\
&= evalh(q, N^{(1)})^2 + evalh(y, N^{(1)})^2 - |evalh(q, N^{(1)})| \\
&\geq |evalh(y, N^{(1)})| \ \square
\end{aligned}$$

Lemma 9 *Let $x \in \mathcal{H}$ and $y \in \mathcal{H}$. If $x \oplus y = 0$, then $x = \ominus y$.*

PROOF: First note that it follows from Definitions 7 and 8 that (a) if $p \in \mathcal{H}$ and $pop(i,p) = 0$, then $p = 0$, and (b) if $p \in \mathcal{H}$, $q \in \mathcal{H}$, and $pow(i,p,q) = 0$, then $p = q = 0$.

The proof is by induction and a case analysis based on Definition 9. Suppose $x \oplus y = 0$

Case 1: $x \in \mathbb{Z}$ or $y \in \mathbb{Z}$.

It must be that both $x \in \mathbb{Z}$ and $y \in \mathbb{Z}$ and $x \oplus y = x + y = 0$, which implies $y = -x = \ominus x$.

Case 2: $x = \texttt{(POP}\ i\ p\texttt{)}$.

Since $x \oplus y = 0$, $y = \texttt{(POP}\ j\ q\texttt{)}$. If $i = j$, then

$$x \oplus y = pop(i, p \oplus q) = 0,$$

which implies $p \oplus q = 0$. By inductive hypothesis, $p = \ominus q$, and hence $x = \ominus y$.

If $i > j$, then

$$x \oplus y = pop(j, \texttt{(POP}\ i-j\ p\texttt{)} \oplus q) = 0$$

implies $\texttt{(POP}\ i-j\ p\texttt{)} \oplus q = 0$, and by inductive hypothesis,

$$q = \ominus\texttt{(POP}\ i-j\ p\texttt{)} = \texttt{(POP}\ i-j\ \ominus p\texttt{)}$$

and $y = \texttt{(POP}\ j\ \texttt{(POP}\ i-j\ \ominus p\texttt{)}$, contradicting Definition 6.

The case $i < j$ similarly leads to a contradiction.

Case 3: $x = \texttt{(POW}\ i\ p\ q\texttt{)}$.

Since $x \oplus y = 0$, $y = \texttt{(POW}\ j\ r\ s\texttt{)}$.

If $i = j$, then $x \oplus y = pow(i, p \oplus r, q \oplus s)$, which implies $p \oplus r = q \oplus s = 0$. By induction, $p = \ominus r$ and $q = \ominus s$, and hence $x = \ominus y$.

If $i > j$, then

$$x \oplus y = pow(j, \texttt{(POW}\ i-j\ p\ 0\texttt{)} \oplus r, q \oplus s) = 0$$

implies $\texttt{(POW}\ i-j\ p\ 0\texttt{)} \oplus r = 0$, and by inductive hypothesis,

$$r = \ominus\texttt{(POW}\ i-j\ p\ 0\texttt{)} = \texttt{(POW}\ i-j\ \ominus p\ 0\texttt{)}$$

and $y = \texttt{(POW}\ j\ \texttt{(POW}\ i-j\ \ominus p\ 0\texttt{)}\ s\texttt{)}$, contradicting Definition 6.

The case $i < j$ similarly leads to a contradiction. □

Theorem 1 *If f and g are polynomial terms over a variable list $V = \texttt{(}v_0 \ldots v_{k-1}\texttt{)}$, then $norm(f,V) = norm(g,V)$ if and only if for every list of values $N = \texttt{(}n_0 \ldots n_{k-1}\texttt{)}$, $evalp(f,A) = evalp(g,A)$, where $A = \texttt{((}v_0\ n_0\texttt{)} \ldots \texttt{(}v_{k-1}\ n_{k-1}\texttt{))}$.*

PROOF: Let $x = norm(f,V)$ and $y = norm(g,V)$. If $x = y$, then by Lemma 7,

$$evalp(f,A) = evalh(x,N) = evalh(y,N) = evalp(g,A).$$

On the other hand, suppose $x \neq y$. Then by Lemma 9, since $y = \ominus(\ominus y)$, $x \oplus (\ominus y) \neq 0$, and consequently, by Lemmas 3, 4, and 8, there exists $N' = \texttt{(}n_0 \ldots n_{\ell-1}\texttt{)}$ such that

$$evalh(x \oplus (\ominus y), N') = evalh(x,N') - evalh(y,N') \neq 0.$$

If $\ell \geq k$, then let N be the initial segment of N' of length k, $N = \texttt{(}n_0 \ldots n_{k-1}\texttt{)}$. If $\ell < k$, then let $N = \texttt{(}m_0 \ldots m_{k-1}\texttt{)}$

$$m_i = \begin{cases} n_i & \text{if } i < \ell \\ 0 & \text{if } \ell \leq i < k. \end{cases}$$

In the latter case, it follows from Definition 5 that $evalh(z,N) = evalh(z,N')$ for every SHF z. Thus, in either case, $evalh(x,N) \neq evalh(y,N)$, and by Lemma 7,

$$evalp(f,A) = evalh(x,N) \neq evalh(y,N) = evalp(f,A).\ \square$$

References

[1] ACL2 home page, `www.cs.utexas.edu/users/moore/acl2/.`

[2] Gregoire, Benjamin and Mahboubi, Assia.: Proving Equalities in a Coomutative Ring Done Right in Coq. In: Proceedings of the 18th International Conference on Theorem Proving in Higher Order Logics. Springer-Verlag (2005)

[3] Russinoff, David M.: A Computationally Surveyable Proof of the Curve25519 Group Axioms. Unpublished manuscript, `www.russinoff.com/papers/group.pdf`.