Logic Proofs - Through Ideal Inclusions

R. Germundsson
Department of Electrical Engineering
Linköping University
S-581 83 Linköping
SWEDEN

Email: roger@isy.liu.se

1991-09-07

Abstract: The two approaches of propositional logic (semantic and proof theoretic) are found to have equivelent formulations in commutative algebra over finite fields. In particular the semantic approach correspond to an algebro geometric formulation and the proof theoretic correspond to an ideal theoretic framework. Based on this correspondence a new completeness proof is given. An implementation of this proof system in Mathematica is also given.

Keywords: Sematics, Proof Theory, Ideal, Algebraic Geometry, Variety, Gröbner Bases

1 Introduction

This section presents the major results and the disposition of the report.

The basic idea of this report is that you can use algebraic methods to do (propositional) logic proofs. More specifically logic propositions may be mapped to polynomials in such a way that truth values are preserved. By using this representation one can simulate logic proofs in either of two modes:

Semantic: The semantic approach to logic proofs basically examines truth values and reasons from that standpoint. This approach is closely related to algebraic geometry and the exact relation is given in theorem 4.2

Proof Theoretic: The proof theoretic approach essentially means using proof rules of the form $\phi \rightarrow \psi$ and ϕ then we can conclude ψ. Even though it is far less obvious, this approach is closely related to determining wether or not an ideal is included in another ideal. See theorem 5.1 for the exact relationship. The ideal inclusion decision is then really determined through polynomial remainder calculation, see section 7 for exact results and examples.

By using the equivalence stated above and a purely algebraic result, namely theorem 2.2, we can prove that the two modes are really equivalent, commonly known as *completeness* within logic. Pictorially this report may be presented as the following result:

$$\begin{array}{rcl} \textbf{Semantic} & & \textbf{Proof Theoretic} \\ \Updownarrow & & \Updownarrow \\ \textbf{Algebraic Geometry} & \Leftrightarrow & \textbf{Ideal Theoretic} \end{array}$$

In the notations of logic and commutative algebra (defined later) the exact result is: Let $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$ be a set of propositions and let ϕ be single proposition in all cases using proposition symbols $x_1, \ldots, x_n$. Let $g_i = P(\gamma_i)$ and $f = P(\phi)$ where P maps propositions to polynomials in such a way that truth values are preserved, see definition 4.2, then:

$$\begin{array}{rcl} \Gamma \models \phi & & \Gamma \vdash \phi \\ \Updownarrow & & \Updownarrow \\ V(g_1 \ldots g_m) \supseteq V(f) & \Leftrightarrow & (g_1 \cdots g_m) \subseteq (f) \end{array}$$

The disposition of the report is as follows:

Algebraic Results contains the algebraic results needed in the subsequent sections, such as notation and the connection between ideals and varieties.

Logic Concepts introduces the most fundamental logical concepts, e.g. formulas and induction over formulas.

Semantics makes the precise connection between algebraic geometry and logical semantics.

Proof Theory makes the corresponding connection between proof theory (of logic) and ideal theory.

Completeness ties together the algebraic geometry and ideal theory and because of the previous sections this means that semantics and proof theory are connected.

Implementation contains an example implementation in Mathematica of both the geometric and ideal theoretic approaches. It also contains examples.

Future Work describes some paths to follow after this these results.

Appendix contains full Mathematica code used for this paper.

2 Algebraic Results

In this section we will introduce the algebraic results needed in subsequent sections.

Basically we will introduce the concepts of *polynomial ring, ideal* and *variety.* More extensive studies of these may be found in [6, 8, 1, 9, 12, 14, 10]. The main results of this section – an extension to Hilbert's Nullstellen Satz to algebraically nonclosed fields of finite characteristic – has so far not been found in any of the references.

The main objects that we work with - in actual computations - are polynomials in several variables. Suppose we work with polynomials in variables $x_1, \ldots, x_n$, then a *monomial* is something of the form:

$$x_1^{\alpha_1} \cdots x_n^{\alpha_n} = x^\alpha, \ \alpha = [\alpha_1, \ldots, \alpha_n] \in N^n$$

Using the multiindex notation above we can easily write a multivariable *polynomial* p as:

$$p = \sum_{\alpha \in N^n} c_\alpha x^\alpha$$

where only finitely many c_α are nonzero. We have not yet specified where our coefficients lie, but in this article they will always come from a field K (i.e. we can sum, multiply and divide). The set of all such polynomials is denoted by $K[x_1, \ldots, x_n]$, i.e.

$$K[x_1, \ldots, x_n] = \{ \sum_{\alpha \in N^n} c_\alpha x^\alpha : c_\alpha \in K \text{ finitely many } c_\alpha \neq 0\}$$

Definition 2.1 *Let* $p_1, \ldots, p_l \in K[x_1, \ldots, x_n]$.

Ideal *The* ideal *a generated by* $p_1, \ldots, p_l$ *is*

$$a = \{ \sum_{1 \leq i \leq l} f_i p_i : f_i \in K[x_1, \ldots, x_n]\}$$

Variety *The* variety *A of* $p_1, \ldots, p_l$ *is the set:*

$$A = \{u \in K^n : p_i(u) = 0, \ i = 1, \ldots, l\}$$

□

Denote the set of all ideals in $K[x_1, \ldots, x_n]$ by $\text{Ideals}(K[x_1, \ldots, x_n])$. The ideal generated by $p_1, \ldots, p_l$ is usually denoted by $(\{p_1, \ldots, p_l\})$ or $(p_1, \ldots, p_l)$.

The variety associated with $p_1, \ldots, p_l$ is the same as the variety associated with $(p_1, \ldots, p_n)$. This variety is usually denoted by $V((p_1, \ldots, p_n))$, i.e.

$$V : \text{Ideals}(K[x_1, \ldots, x_n]) \to \text{Pow}(K^n)$$

where $\mathrm{Pow}(K^n)$ is the set of subsets of K^n. Symmetrically we can consider

$$I : \mathrm{Pow}(K^n) \to \mathrm{Ideals}(K[x_1, \ldots, x_n])$$

defined by:

$$I(A) = \{p \in K[x_1, \ldots, x_n] : p(u) = 0 \ \forall u \in A\}$$

and it is readily verified that $I(A)$ is an ideal.

The following are more or less direct consequences of these definitions:

Theorem 2.1 *Let $a_1, a_2 \in \mathrm{Ideals}(K[x_1, \ldots, x_n])$ then*

$$\begin{aligned}
V((0)) &= K^n && (1)\\
V((1)) &= \emptyset && (2)\\
V(a_1 \cap a_2) &= V(a_1) \cup V(a_2) && (3)\\
V(a_1 + a_2) &= V(a_1) \cap V(a_2) && (4)\\
a_1 \subseteq a_2 &\Rightarrow V(a_1) \supseteq V(a_2) && (5)
\end{aligned}$$

Let $A_1, A_2 \in \mathrm{Pow}(K^n)$ then

$$\begin{aligned}
I(K^n) &= (0) && (6)\\
I(\emptyset) &= K[x_1, \ldots, x_n] && (7)\\
A_1 \subseteq A_2 &\Rightarrow I(A_1) \supseteq I(A_2) && (8)
\end{aligned}$$

And combining these we have: ($a \in \mathrm{Ideals}(K[x_1, \ldots, x_n])$, $A \in \mathrm{Pow}(K^n)$)

$$\begin{aligned}
I(V(a)) &\supseteq a && (9)\\
V(I(A)) &\supseteq A && (10)\\
I(V(I(A))) &= I(A) && (11)\\
V(I(V(a))) &= V(a) && (12)
\end{aligned}$$

□

These results may be found in [12]. It is worth noting that

$$(1) = K[x_1, \ldots, x_n]$$

In this article we will utilize $K = Z_2$, i.e. the integers modulo 2.

Theorem 2.2 (Main Theorem) *Let p be prime then*

$$\begin{aligned}
\forall A \in \mathrm{Pow}(Z_p^n) &\qquad V(I(A)) = A && (13)\\
\forall a \in \mathrm{Ideals}(Z_p[x_1, \ldots, x_n]) &\qquad I(V(a)) = a + (x_1^p - x_1, \ldots, x_n^p - x_n) && (14)
\end{aligned}$$

□

Proof 2.1 *See [4] for the proof.*

In words equation (13) states that every subset of Z_p^n can be represented as a variety of some ideal or equivalently that every set is the set of common zeros to a set of polynomials. See example 2.1 for how you can constructively compute such a set of polynomials.

Similarly equation (14) states that we can almost uniquely recover our original ideal from its variety. However we get some extra elements, namely

$$(x_1^p - x_1, \ldots, x_n^p - x_n)$$

Intuitively this can be realized from the fact that

$$x^{(p-1)} = 1 \bmod p$$

by Fermat's little theorem, i.e.

$$x^p - x = 0 \; \forall x \in Z_p$$

So if we calculate the variety of $(x_1^p - x_1, \ldots, x_n^p - x_n)$ we get Z_p^n. It can also be shown that $(x_1^p - x_1, \ldots, x_n^p - x_n)$ is the largest ideal with this property, see [4] for details.

Let

$$R_p[x_1, \ldots, x_n] = Z_p[x_1, \ldots, x_n]/(x_1^p - x_1, \ldots, x_n^p - x_n) \tag{15}$$

(see [1] for a definition of this). We will use the notations R_p^n, $R_p[x]$ and $R_p^n[x]$ to denote $R_p[x_1, \ldots, x_n]$. We can now restate equation (14) in terms of ideals in R_p^n as

$$\forall a \in \text{Ideals}(R_p^n) \;\; I(V(a)) = a \tag{16}$$

This is just a compact restatement of equation (14). It is interesting to note that theorem (2.2) is an extension of the famous Hilbert Nullstellen Satz, see [12].

We need to know how to compute polynomials with prescribed variety, or more generally we need to know how to compute a polynomial $q \in Z_p[x_1, \ldots, x_n]$ with prescribed values at certain points. Let

$$L_i(x) = \frac{x(x-1)\cdots(x-(i-1))(x-(i+1))\cdots(x-(p-1))}{(i-0)(i-1)\cdots(i-(i-1))(i-(i+1))\cdots(i-(p-1))} \in Z_p[x]$$

then

$$L_i(x) = \begin{cases} 1 & , x = i \\ 0 & , x \neq i \end{cases}$$

From this we can define

$$L_\alpha(x_1, \ldots, x_n) = L_{\alpha_1}(x_1) \cdots L_{\alpha_n}(x_n), \;\; \alpha \in Z_p^n$$

with the properties:

$$L_\alpha(x_1, \ldots, x_n) = \begin{cases} 1 & , [x_1, \ldots, x_n] = \alpha \\ 0 & , [x_1, \ldots, x_n] \neq \alpha \end{cases}$$

Suppose we are given a function $f : Z_p^n \to Z_p$ we can then synthesize a polynomial q such that $q(\alpha) = f(\alpha)\ \forall \alpha \in Z_p^n$. This polynomial q is given by:

$$q = \sum_{\alpha \in Z_p^n} L_\alpha(x_1, \ldots, x_n) f(\alpha) \tag{17}$$

This approach of generating a polynomial with prescribed values at certain points is usually called *Lagrange interpolation*. This is illustrated in the following example:

Example 2.1 *Perform this in the ring $Z_3[x, y]$, i.e. $p = 3, n = 3$. Let*

$$A = \{(0, 0), (1, 0), (0, 1), (2, 2), (1, 2), (0, 2)\}$$

we then want to find a polynomial q such that

$$q(x, y) = \begin{cases} 0 & (x, y) \in A \\ 1 & (x, y) \notin A \end{cases}$$

By using equation (17) we obtain:

$$q = x + 2x^2 + 2x^2y + 2xy^2$$

We also have $V(q) = A$.

3 Logic Concepts

This section introduces some fundamental concepts from propositional logic such as formulas as well as inductions and recursions over formulas. The presentation follows van Dalen [16]. We recursively define all formulas through:

Definition 3.1 *The set $PROP[x_1, \ldots, x_n]$ is the smallest set X with the properties.*

$$x_1, \ldots, x_n \in X \tag{18}$$

$$\perp \in X \tag{19}$$

$$\phi, \psi \in X \quad \Rightarrow \quad (\phi \wedge \psi), (\phi \vee \psi), (\phi \to \psi), (\phi \leftrightarrow \psi) \in X \tag{20}$$

$$\phi \in X \quad \Rightarrow \quad (\neg \phi) \in X \tag{21}$$

$\square$

Van Dalen [16] make use of proposition symbols $p_0, p_1, \ldots$, but that is an unfortunate notation since we want to map this to polynomials typically denoted by p_i and also we really don't need infinitely many of them in any specific case. The notation $PROP[x_1, \ldots, x_n]$ thus means the set of propositional logic formulas in the variables (or proposition symbols) $x_1, \ldots, x_n$. The symbols $\{\wedge, \vee, \rightarrow, \neg, \leftrightarrow, \perp\}$ are usually called logical *connectives*.

Just for the sake of it, it might be a good idea to write down the counterpart of definition 3.1 for polynomial rings:

The set $R[x_1, \ldots, x_n]$ is the smallest set X with the properties.

$$\begin{aligned} & & x_1, \ldots, x_n \in X \quad (22) \\ r \in R & \Rightarrow & r \in X \quad (23) \\ p, q \in X & \Rightarrow & (p+q), (p*q) \in X \quad (24) \end{aligned}$$

The point of writing down such a recursive definition as 3.1 is that you may prove certain properties by induction over the set of formulas as well as define functions recursively over formulas (as we will do). In particular we have:

Theorem 3.1 *Let A be a property, $A(\phi)$ holds for all $\phi \in PROP[x_1, \ldots, x_n]$ if*

- $A(x_i),\ i = 1, \ldots, n$
- $A(\perp)$
- $A(\phi), A(\psi) \Rightarrow A(\phi \Box \psi)$ *where* $\Box \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$
- $A(\phi) \Rightarrow A(\neg\phi)$

□

Proof 3.1 *Let $X = \{\phi \in PROP[x_1, \ldots, x_n] : A(\phi)\}$ so since the cases of the theorem exactly match the cases of definition 3.1 we have:*

$$X = PROP[x_1, \ldots, x_n]$$

4 Semantics

Semantics of logical expressions is the intended meaning. More precisely this is a calculus that tells us how the meaning of composite expressions is built of from the meaning of subexpressions. This is pretty much what you would do when writing down a truth table for a composite expression. In our case we have the truth tables:

x	y	$\neg x$	$x \wedge y$	$x \vee y$	$x \rightarrow y$	$x \leftrightarrow y$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Having this truth table in mind we define a *valuation*.

Definition 4.1 *A mapping* $v : PROP[x_1, \ldots, x_n] \to \{0, 1\}$ *is a* valuation *if for* $\phi, \psi \in PROP[x_1, \ldots, x_n]$

$$\begin{aligned}
v(\phi \wedge \psi) &= \min(v(\phi), v(\psi)) \\
v(\phi \vee \psi) &= \max(v(\phi), v(\psi)) \\
v(\phi \to \psi) &= v(\neg\phi \vee \psi) \\
v(\phi \leftrightarrow \psi) &= v((\phi \to \psi) \wedge (\psi \to \phi)) \\
v(\neg\phi) &= 1 - v(\phi) \\
v(\bot) &= 0 \\
v(x_i) &\in \{0, 1\}
\end{aligned}$$

□

These rules are expressed in ordinary arithmetic and basically describe how you recursively calculate the meaning (or truth value) of a logic formula given the truth values of the elementary proposition symbols.

As was shown in example 2.1 any function $f : Z_p^n \to Z_p$ (p prime) can be realized as a polynomial (by use of Lagrange interpolating polynomial). In particular this is true if $p = 2$ and hence we should be able to construct polynomials $p_\wedge, p_\vee, p_\to, p_\leftrightarrow, p_\neg$ having the same truth tables as $\wedge, \vee, \to, \leftrightarrow, \neg$. This computation yields:

$$p_\neg(x) = 1 + x \tag{25}$$

$$p_\wedge(x, y) = xy \tag{26}$$

$$p_\vee(x, y) = x + y + xy \tag{27}$$

$$p_\to(x, y) = 1 + x + xy \tag{28}$$

$$p_\leftrightarrow(x, y) = 1 + x + y \tag{29}$$

In particular this is a means by which we can translate logic formulas to polynomials, but preserving truth values. This is of course something we do recursively:

Definition 4.2 *Let the mapping* $P : PROP[x_1, \ldots, x_n] \to Z_2[x_1, \ldots, x_n]$ *be defined through:*

$$\begin{aligned}
P(x_i) &= x_i, \; i = 1, \ldots, n \\
P(\phi \Box \psi) &= p_\Box(P(\phi), P(\psi)), \; \Box \in \{\wedge, \vee, \to, \leftrightarrow\} \\
P(\neg\phi) &= p_\neg(P(\phi)) \\
P(\bot) &= 0
\end{aligned}$$

□

The following theorem justifies our mapping of propositions to polynomials.

Theorem 4.1 *Let $v : PROP[x_1, \ldots, x_n] \to \{0,1\}$ be valuation such that $v(x_i) = \alpha_i \in \{0,1\}$, then we have:*

$$\forall \phi \in PROP[x_1, \ldots, x_n] \;\; v(\phi) = P(\phi)(\alpha_1, \ldots, \alpha_n)$$

$\square$

Proof 4.1

- **Basis:** *We have $v(\bot) = 0 = P(\bot)$ and also $v(x_i) = \alpha_i = P(x_i)(\alpha)$ for $i = 1, \ldots, n$.*
- **Induction:** *Assume that we have $v(\phi) = P(\phi)(\alpha)$ and $v(\psi) = P(\psi)(\alpha)$ then we have:*

  $$v(\phi \Box \psi) = P(\phi \Box \psi)(\alpha), \quad \Box \in \{\wedge, \vee, \to, \leftrightarrow\}$$

  and also

  $$v(\neg\phi) = P(\neg\phi)(\alpha)$$

  Which in all cases is easily verified by truthtables for $p_\wedge, p_\vee, p_\to, p_\leftrightarrow, p_\neg$.

The map P maps propositions to polynomials. It is interesting to exhibit the reverse map

$$B : Z[x_1, \ldots, x_n] \to PROP[x_1, \ldots, x_n]$$

There are several choices of such a map, but we will make use of only $\bot$, $\to$ and $\wedge$ since we in the next section will concentrate on those.

$$\begin{aligned} B(0) &= \bot && (30) \\ B(1) &= \bot \to \bot = \top && (31) \\ B(pq) &= B(p) \wedge B(q) && (32) \\ B(p+q) &= ((B(p) \to B(q)) \wedge (B(q) \to B(p))) \to \bot && (33) \end{aligned}$$

See section 7 for examples of these functions.

Another logic concept is *semantic consequence* $\models$.

Definition 4.3 *Let* $\mathrm{Mod}(\phi) = \{v : v(\phi) = 1\}$ *be the* model set *of ϕ and* $\mathrm{Mod}(\{\phi_1, \ldots, \phi_m\}) = \mathrm{Mod}(\phi_1) \cap \cdots \cap \mathrm{Mod}(\phi_m)$. *Let $\Gamma \subset PROP[x_1, \ldots, x_n]$ and $\phi \in PROP[x_1, \ldots, x_n]$ then we have that:*

- *ϕ is a* semantic consequence *of Γ, denoted by $\Gamma \models \phi$, iff*

  $$\mathrm{Mod}(\Gamma) \subseteq \mathrm{Mod}(\phi)$$

- *ϕ is a* tautology *iff $\emptyset \models \phi$, denoted by $\models \phi$.*

□

By now it is pretty clear that a valuation is fully defined by how it evaluates elementary propositions. In particular one could think of a valuation as a point in $\{0,1\}^n$. With this view in mind it is natural to think of $\mathrm{Mod}(\{\phi_1,\ldots,\phi_m\})$ as the set of points for which all propositions $\phi_1,\ldots,\phi_m$ are true (the modelset). In particular a tautology has $\{0,1\}^n$ as modelset.

Explained in this manner the following is pretty natural:

Theorem 4.2 *Let $\gamma_1,\ldots,\gamma_m,\phi \in PROP[x_1,\ldots,x_n]$ then*

$$\{\gamma_1,\ldots,\gamma_m\} \models \phi \Leftrightarrow V(P(\gamma_1)) \cup \cdots \cup V(P(\gamma_m)) \supseteq V(P(\phi))$$

□

Proof 4.2

$$\begin{array}{rcc} \{\gamma_1,\ldots,\gamma_m\} \models \phi & & \\ \text{def} & \Leftrightarrow & \mathrm{Mod}(\gamma_1) \cap \cdots \cap \mathrm{Mod}(\gamma_m) \subseteq \mathrm{Mod}(\phi) \\ \text{by thm 4.1} & \Leftrightarrow & V(P(\gamma_1)+1) \cap \cdots \cap V(P(\gamma_m)+1) \subseteq V(P(\phi)+1) \\ \textit{def of } V & \Leftrightarrow & \overline{V(P(\gamma_1))} \cap \cdots \cap \overline{V(P(\gamma_m))} \subseteq \overline{V(P(\phi))} \\ & \Leftrightarrow & V(P(\gamma_1)) \cup \cdots \cup V(P(\gamma_m)) \supseteq V(P(\phi)) \end{array}$$

where $V(\{p_1,\ldots,p_n\}) = \{\alpha \in Z_2^n : p_i(\alpha) = 0\}$ is the variety of $\{p_1,\ldots,p_n\}$.

From this theorem it is obvious that any proof of a statement $\Gamma \models \phi$ could equally well be carried out in a polynomial context. Let's illustrate this by an example:

Example 4.1 *Suppose we want to prove*

$$\models \neg(x \wedge y) \leftrightarrow (\neg x \vee \neg y)$$

by computing its truth table we have:

x	y	$\neg(x \wedge y) \leftrightarrow (\neg x \vee \neg y)$
0	*0*	*1*
0	*1*	*1*
1	*0*	*1*
1	*1*	*1*

Hence $\mathrm{Mod}(\emptyset) = \{0,1\}^2 = \mathrm{Mod}(\neg(x \wedge y) \leftrightarrow (\neg x \vee \neg y))$.

To do the proof using polynomials we first map our expression to $Z_2[x,y]$. We get

$$P(\neg(x \wedge y) \leftrightarrow (\neg x \vee \neg y)) = 3 + x + xy + x(1+y) = 1 \text{ in } \mathrm{Z_2[x,y]}$$

and hence $V(1) = \emptyset$, thus $\emptyset \supseteq V(1)$.

5 Proof Theory

There is another view of logic not based on its semantics (i.e. truth tables), but rather based on inference rules. As an example, take *modus ponens*, i.e. if we know ϕ and $\phi \rightarrow \psi$ then we can conclude ψ. We could write this rule as:

$$\frac{\phi \quad \phi \rightarrow \psi}{\psi}$$

or as

$$\{\phi, \phi \rightarrow \psi\} \vdash \psi$$

If we only consider only $\{\wedge, \rightarrow, \perp\}$ the following rulebase (by Gentzen) is enough for our purposes (to be made precise later).

R1

$$\frac{\phi \quad \psi}{\phi \wedge \psi} \quad \Leftrightarrow \quad \{\phi, \psi\} \vdash \phi \wedge \psi$$

R2

$$\frac{\phi \wedge \psi}{\phi} \; \frac{\phi \wedge \psi}{\psi} \quad \Leftrightarrow \quad \{\phi \wedge \psi\} \vdash \psi \text{ and } \{\phi \wedge \psi\} \vdash \phi$$

R3

$$\frac{\begin{array}{c}\phi \\ \vdots \\ \psi\end{array}}{\phi \rightarrow \psi} \quad \Leftrightarrow \quad \{\phi, \ldots, \psi\} \vdash \phi \rightarrow \psi$$

R4

$$\frac{\phi \; \phi \rightarrow \psi}{\psi} \quad \Leftrightarrow \quad \{\phi, \phi \rightarrow \psi\} \vdash \psi$$

R5

$$\frac{\perp}{\phi} \quad \Leftrightarrow \quad \{\perp\} \vdash \phi$$

R6

$$\frac{\begin{array}{c}\neg\phi \\ \vdots \\ \perp\end{array}}{\phi} \quad \Leftrightarrow \quad \{\neg\phi, \ldots, \perp\} \vdash \phi$$

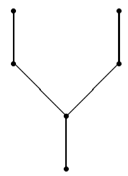
The elements of this rulebase go by several different names, e.g. R4 is called modus ponens and R6 is usually called reducti ad absurdum.

Now new propositions are deduced by repeatedly applying these rules and an example will readily illustrate this:

Example 5.1 *Suppose our initial hypothesis is $\phi \wedge \psi$ so if we first apply R2 twice to get ϕ and ψ then use R1 to obtain $\psi \wedge \phi$. If we then take our initial hypothesis $\phi \wedge \psi$ and $\psi \wedge \phi$ and apply R3 we obtain $\phi \wedge \psi \rightarrow \psi \wedge \phi$. All of this can be captured in the following which is called a* derivation.

$$\dfrac{\dfrac{\dfrac{\phi\wedge\psi}{\psi} \quad \dfrac{\phi\wedge\psi}{\phi}}{\psi\wedge\phi}}{\phi\wedge\psi\rightarrow\psi\wedge\phi}$$

If we try to extract the structure of the formula above substituting horizontal lines with vertical lines and formulas with nodes we obtain a tree. In the above case the derivation tree:



We need to somehow capture all these derivations possible w.r.t. our rulebase we can do this in the set of *derivations*:

Definition 5.1 *The set of derivations is the smallest set X such that:*

$$\forall\phi \in PROP[x_1, \ldots, x_n] \quad \Rightarrow \quad \phi \in X \tag{34}$$

$$\begin{array}{c}D\\ \phi\end{array}, \begin{array}{c}D'\\ \phi'\end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{cc}D & D'\\ \phi & \phi'\end{array}}{\phi \wedge \phi'} \in X \tag{35}$$

$$\begin{array}{c}D\\ \phi\wedge\psi\end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{c}D\\ \phi\wedge\psi\end{array}}{\phi}, \dfrac{\begin{array}{c}D\\ \phi\wedge\psi\end{array}}{\psi} \in X \tag{36}$$

$$\begin{array}{c}\phi\\ D\\ \psi\end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{c}\phi\\ D\\ \psi\end{array}}{\phi\rightarrow\psi} \in X \tag{37}$$

$$\begin{array}{c}D\\ \phi\end{array}, \begin{array}{c}D'\\ \phi\rightarrow\psi\end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{cc}D & D'\\ \phi & \phi\rightarrow\psi\end{array}}{\psi} \in X \tag{38}$$

$$\begin{array}{c} D \\ \perp \end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{c} D \\ \perp \end{array}}{\phi} \in X \tag{39}$$

$$\begin{array}{c} \neg\phi \\ D \\ \perp \end{array} \in X \quad \Rightarrow \quad \dfrac{\begin{array}{c} \neg\phi \\ D \\ \perp \end{array}}{\phi} \in X \tag{40}$$

The top formulas are called hypotheses *and the bottom formula is called* conclusion. □

Since the set of derivations is recursively defined we can again use induction over all the elementary recursion steps in definition 5.1 to show a property to hold for all derivations.

Definition 5.2 *Let $\Gamma \subset PROP[x_1, \ldots, x_n]$ and $\phi \in PROP[x_1, \ldots, x_n]$ then we have*

- *ϕ is* derivable *from Γ, denoted by $\Gamma \vdash \phi$ if there is a derivation D with hypotheses in Γ and conclusion ϕ.*
- *ϕ is a* theorem *iff $\emptyset \vdash \phi$, denoted by $\vdash \phi$.*

□

Indeed this offers us a nice dynamic system interpretation as follows: Think of $\Gamma = \Gamma_0 \subset PROP[x_1, \ldots, x_n]$ as an initial state. Then by applying any of the rules R1-R6 (that match) we obtain Γ_1, i.e. another set of formulas. Then $\Gamma \vdash \phi$ is equivalent to that for some $N \geq 0$ we have $\phi \in \Gamma_N$. Formally:

$$\begin{aligned} \Gamma_{n+1} &= f(\Gamma_n, R) \\ \Gamma_0 &= \Gamma \end{aligned}$$

where $R \in \{R1, \ldots, R6\}$. Strictly speaking there is some nondeterminism in the choice of which formulas we match when a rule is applied above, but essentially this interpretation translates provability to reachability.

One might wonder how derivability can be connected to polynomials, but with an eye toward completeness we obtain the following:

Theorem 5.1 *Let $\gamma_1, \ldots, \gamma_m, \phi \in PROP[x_1, \ldots, x_n]$ then*

$$\Gamma = \{\gamma_1, \ldots, \gamma_m\} \vdash \phi \Leftrightarrow (P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi))$$

where (p) is the ideal generated by $p \in R_2^n$. □

Proof 5.1 *First we prove*

$$\{\gamma_1, \ldots, \gamma_m\} \vdash \phi \Rightarrow (P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi))$$

which is done by induction over the set of derivations following definition 5.1.

- **Basis:** *$\forall\phi \in PROP[x_1,\dots,x_n]$ the one element tree ϕ is a proper derivation, i.e. $\{\phi\} \vdash \phi$. Of course we trivially have $(P(\phi)) \subseteq (P(\phi))$.*
- **Induction:** *Let us introduce some notation to simplify this part:*

$$\begin{aligned} \Phi &= \{\phi_1,\dots,\phi_m\} & \Psi &= \{\psi_1,\dots,\psi_n\} \\ f_i &= P(\phi_i)\ i = 1,\dots,m & g_j &= P(\psi_j)\ j = 1,\dots,n \\ F &= (f_1)\cap\cdots\cap(f_m) & G &= (g_1)\cap\cdots\cap(g_n) \\ f &= P(\phi) & g &= P(\psi) \end{aligned}$$

  - **Eq. 35** *can be rewritten as*

    $$\Phi \vdash \phi \text{ and } \Psi \vdash \psi \Rightarrow \Phi \cup \Psi \vdash \phi \wedge \psi$$

    In R_2^n we have:

    $$\begin{aligned} F \subseteq (f) \text{ and } G \subseteq (g) &\quad\Leftrightarrow \\ f_1\cdots f_m = fp \text{ and } g_1\cdots g_n = gq &\quad\Rightarrow \\ f_1\cdots f_m g_1\cdots g_n = fpgq = (gf)(pq) &\quad\Leftrightarrow \\ F \cap G \subseteq (fg) & \end{aligned}$$

  - **Eq. 36** *is equivalent to:*

    $$\Phi \vdash \phi \wedge \psi \Rightarrow \Phi \vdash \phi \text{ and } \Phi \vdash \psi$$

    In R_2^n we get:

    $$\begin{aligned} F \subseteq (fg) &\quad\Leftrightarrow \\ f_1\cdots f_m = (fg)p = f(gp) = g(fp) &\quad\Leftrightarrow \\ F \subseteq (f) \text{ and } F \subseteq (g) & \end{aligned}$$

  - **Eq. 37** *is:*

    $$\phi \in \Phi \vdash \psi \Rightarrow \Phi \vdash \phi \to \psi$$

    In R_2^n:

    $$\phi \in \Phi \Leftrightarrow f = f_i\ \exists i = 1,\dots,m$$

    This trivially means $F \subseteq (f)$ and we have $F \subseteq (g)$ by assumption. Together this implies:

    $$F \subseteq (fg) \tag{41}$$

    Since $P(\phi \to \psi) = 1 + f + fg$ and

    $$f(1+f+fg) = f + f^2 + f^2 g = //f^2 + f = 0, f^2 = f// = fg$$

    we have

    $$(fg) \subseteq (1 + f + fg) \tag{42}$$

    Using equations (41) and (42) we thus obtain

    $$F \subseteq (fg) \subseteq (1 + f + fg)$$

– **Eq. 38** *we rewrite as:*

$$\Phi \vdash \phi \text{ and } \Psi \vdash \phi \to \psi \Rightarrow \Phi \cup \Psi \vdash \psi$$

In R_2^n *we have:*

$$\begin{aligned} F \subseteq (f) \text{ and } G \subseteq (1+f+fg) &\Leftrightarrow \\ f_1 \cdots f_m = fp \text{ and } g_1 \cdots g_n = (1+f+fg)q &\Rightarrow \\ f_1 \cdots f_m g_1 \cdots g_n = f(1+f+fg)pq = g(fpq) &\Leftrightarrow \\ F \cap G \subseteq (g) & \end{aligned}$$

– **Eq. 39** *is:*

$$\Phi \vdash \perp \Rightarrow \Phi \vdash \phi$$

In R_2^n *we have:*

$$\begin{aligned} F \subseteq (0) &\Rightarrow \\ F = (0) \subseteq (f) & \end{aligned}$$

– **Eq. 40** *is:*

$$\neg\phi \in \Phi \vdash \perp \Rightarrow \Phi \vdash \phi$$

In R_2^n *we obtain:*

$$F \subseteq (0) \subseteq (f)$$

Hence we have proved

$$\Phi \vdash \phi \Rightarrow F \subseteq (f)$$

Now for the reverse implication it is enough to study ideal inclusions of the form $(g) \subseteq (f)$*, i.e. principal ones. We then have*

$$(g) \subseteq (f) \Leftrightarrow g = fp$$

but this is equivalent to $\{\psi = \phi \wedge \omega\} \vdash \phi$ *where* $P(\psi) = g$*,* $P(\phi) = f$ *and* $P(\omega) = p$ *which we know is a valid proof rule. The only concern is if there exist* ψ*,* ϕ *and* ω *that map to* g*,* f *and* p *respectively. In fact we can chose* $\psi = B(g)$*,* $\phi = B(f)$ *and* $\omega = B(p)$ *where* B *is defined in equations (30-33).*

Corollary 5.1 ϕ *is a theorem, i.e.* $\vdash \phi$*, iff* $(P(\phi)) = (1)$*.* □

This theorem says something very exciting: That to test wether or not $\Gamma \vdash \phi$ we can either try apply our set of proof rules or test $(P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi))$. This is fairly straightforward to test and indeed from section 7 we know

$$(P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi)) \Leftrightarrow \mathrm{Rem}(P(\phi), \mathrm{GB}(P(\gamma_1) \cdots P(\gamma_m))) = 0$$

Although we don't really need that machinery here.

Example 5.2

- $\vdash (\phi \to (\psi \to \sigma)) \to ((\phi \wedge \psi) \to \sigma)$
  Suppose we have $P(\phi) = x_1$, $P(\psi) = x_2$ and $P(\sigma) = x_3$ then we get

$$\begin{aligned} & P((\phi \to (\psi \to \sigma)) \to ((\phi \wedge \psi) \to \sigma)) \\ = \;& 2 + x_1 + x_1(1 + x_2 + x_2x_3) + \\ & (1 + x_1x_2 + x_1x_2x_3)(1 + x_1 + x_1(1 + x_2 + x_2x_3)) \\ //\text{in } R_2^3// \; = \;& 1 \end{aligned}$$

- $\vdash \phi \to ((\phi \to \bot) \to \bot)$
  Suppose we have $P(\phi) = x_1$ then:

$$\begin{aligned} P(\phi \to ((\phi \to \bot) \to \bot)) &= 1 + x_1 + x_1^2 \\ //\text{in } R_2^1// &= 1 \end{aligned}$$

- $\{\phi \wedge \psi\} \vdash \psi \wedge \phi$
  Again suppose $P(\phi) = x_1$ and $P(\psi) = x_2$ then this statement is equivalent to

$$(x_1x_2) \subseteq (x_2x_1)$$

  which is trivially true.

6 Completeness

The completeness theorem for propositional logic states that you may use either semantic or proof theoretic methods to do logic proofs since result will be the same. For more complicated logics, i.e. predicate logic completeness still holds (proved by Gödel). In those cases the semantic approach isn't even an option since one usually have variables that belong to infinite sets, e.g. integers.

In the propositional logic case we get completeness from a nice result in algebraic geometry [4].

Theorem 6.1 Completeness: *Let $\Gamma = \{\gamma_1, \ldots, \gamma_m\}$ and $\gamma_1, \ldots, \gamma_m, \phi \in PROP[x_1, \ldots, x_n]$ then we have*

$$\begin{aligned} \Gamma \models \phi \quad & \Leftrightarrow \\ V(P(\gamma_1)) \cup \cdots \cup V(P(\gamma_m)) \supseteq V(P(\phi)) \quad & \Leftrightarrow \\ (P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi)) \quad & \Leftrightarrow \\ \Gamma \vdash \phi \quad & \end{aligned}$$

□

Proof 6.1 *The first equivalence is from theorem 4.2 and the last is from theorem 5.1. For the middle equivalence we have:*

$\Rightarrow$

$$\begin{array}{rcl} V(P(\gamma_1)) \cup \cdots \cup V(P(\gamma_m)) \supseteq V(P(\phi)) & \Rightarrow & //\text{apply } I// \\ I(V(P(\gamma_1))) \cap \cdots \cap I(V(P(\gamma_m))) \subseteq I(V(P(\phi))) & \Rightarrow & //\text{by thm } 2.2// \\ (P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi)) & & //\text{in } R_2^n// \end{array}$$

$\Leftarrow$

$$\begin{array}{rcl} (P(\gamma_1)) \cap \cdots \cap (P(\gamma_m)) \subseteq (P(\phi)) & \Rightarrow & //\text{apply } V// \\ V(P(\gamma_1)) \cup \cdots \cup V(P(\gamma_m)) \supseteq V(P(\phi)) & & \end{array}$$

So in both directions we make use theorem 2.2.

7 Implementation

In this section it is described how you easily implement a proof system based on the results of this report. This implementation is in terms Gröbner basis computations and remainder calculations and may be directly performed on any of the major computer algebra systems around, e.g. *Mathematica* [17], *Maple* [3], *Macsyma* [15] and *Reduce* [5]. These examples are computed using Mathematica.

First we need to be able to map propositions to polynomials, i.e. the map P of definition 4.2. As a matter of fact you can pretty much copy this down directly into Mathematica by first noting the following notational differences from the rest of this report:

Report	**Mathematica**
$\perp$	`False`
$\top$	`True`
$a \wedge b$	`And[a,b] = a && b`
$a \vee b$	`Or[a,b] = a || b`
$a \rightarrow b$	`Implies[a,b]`
$a \leftrightarrow b$	`Equal[a,b] = a == b`
$\neg a$	`Not[a] = ! a`

So in Mathematica we get:

```
P[ True             ] := 1
P[ False            ] := 0
P[ e_Symbol         ] := e
P[ Not[e_]          ] := 1 + P[e]
P[ Implies[e1_,e2_] ] := 1 + P[e1] + P[e1] P[e2]
P[ And[e1_,e2_]     ] := P[e1] P[e2]
P[ Or[e1_,e2_]      ] := P[e1] + P[e2] + P[e1] P[e2]
P[ Equal[e1_,e2_]   ] := 1 + P[e1] + P[e2]
```

We do similarly with the map B that maps polynomials to propositions:

```
B[ 1                  ] := True
B[ 0                  ] := False
B[ e_Symbol           ] := e
B[ Times[e1_,e2_]     ] := B[e1] && B[e2]
B[ Plus[e1_,e2_]      ] := (! B[e1] && B[e2]) || (B[e1] && ! B[e2])
B[ Power[e_,n_]       ] := B[ e ] B[e^(n-1)]   /; n > 1
B[ Power[e_,1]        ] := B[ e ]
```

We can try this out in an example:

Example 7.1 *Let $\phi = x_1 \wedge (x_2 \vee (\neg x_3))$ then in Mathematica we obtain:*

```
Mathematica (sun4) 1.2 (June 13, 1990) [With pre-loaded data]
by S. Wolfram, D. Grayson, R. Maeder, H. Cejtin,
   S. Omohundro, D. Ballman and J. Keiper
with I. Rivin and D. Withoff
Copyright 1988,1989,1990 Wolfram Research Inc.
 -- X11 windows graphics initialized --

In[1]:= <<PropositionalCalc.m

In[2]:= f = x1 && (x2 || !x3)

Out[2]= x1 && (x2 || !x3)

In[3]:= q = P[f]

Out[3]= x1 (1 + x2 + x3 + x2 (1 + x3))

In[4]:= Mod[ Expand[q], 2 ]

Out[4]= x1 + x1 x3 + x1 x2 x3

In[5]:= g = B[ q ]

Out[5]= x1 && !(!x2 && (!x3 && x2 && !x3 || x3 && !(x2 && !x3)) ||

>       x2 && !(!x3 && x2 && !x3 || x3 && !(x2 && !x3)))
```

It is interesting to compute the truth tables of `f, b, g` *above and if everything works out they should be the same. Except for the first line below that contains information regarding the expression whose truth table has been computed we see by inspection that these expressions really define the same function. (The notation* `expr /. {True->1,False->0}` *only substitutes* `1` *for* `True` *and* `0` *for* `False` *in the table.)*

```
In[6]:= ftab = BTable[ f, {x1,x2,x3}] /. {True->1, False->0}

Out[6]= {{x1, x2, x3, x1 && (x2 || !x3)}, {0, 0, 0, 0}, {0, 0, 1, 0},

>    {0, 1, 0, 0}, {0, 1, 1, 0}, {1, 0, 0, 1}, {1, 0, 1, 0}, {1, 1, 0, 1},

>    {1, 1, 1, 1}}
```

```
In[7]:= gtab = BTable[ g, {x1,x2,x3}] /. {True->1, False->0}

Out[7]= {{x1, x2, x3, x1 && !(!x2 &&

>          (!x3 && x2 && !x3 || x3 && !(x2 && !x3)) ||

>         x2 && !(!x3 && x2 && !x3 || x3 && !(x2 && !x3)))}, {0, 0, 0, 0},

>    {0, 0, 1, 0}, {0, 1, 0, 0}, {0, 1, 1, 0}, {1, 0, 0, 1}, {1, 0, 1, 0},

>    {1, 1, 0, 1}, {1, 1, 1, 1}}

In[8]:= qtab = ZTable[ q, {x1,x2,x3} ]

Out[8]= {{x1, x2, x3, x1 (1 + x2 + x3 + x2 (1 + x3))}, {0, 0, 0, 0},

>    {0, 0, 1, 0}, {0, 1, 0, 0}, {0, 1, 1, 0}, {1, 0, 0, 1}, {1, 0, 1, 0},

>    {1, 1, 0, 1}, {1, 1, 1, 1}}
```

As can be seen from these computations the truth tables are indeed identical. We can build a proof system based on $\models$ by comparing truth tables, this is done at the end of this section.

This is quite enough for the semantic parts, but the really interesting part is of course the ideal inclusion part. Basically we want to test $(g) \subseteq (f)$ in R_2^n. This can equivalently be stated as

$$(g) + (x_1^2 + x_1, \dots, x_n^2 + x_n) \subseteq (f) + (x_1^2 + x_1, \dots, x_n^2 + x_n)$$

in $Z_2[x_1, \dots, x_n]$. The basic tool for testing ideal inclusion och ideal membership is Gröbner bases, see [2, 13, 11] for detailed account of this.

The basic result is this:

Theorem 7.1 *Given $I = (F)$ where $F = \{f_1, \dots, f_m\}$ and $f_i \in K[x_1, \dots, x_k]$ then we can compute a Gröbner basis $G = \{g_1, \dots, g_n\}$ such that $I = (G)$ and*

- $q \in I \Leftrightarrow \mathrm{Rem}(q, G) = 0$
- $\forall q \in K[x_1, \dots, x_n]$ $\mathrm{Rem}(q, G)$ *is unique.*
- $(\{h_1, \dots, h_l\}) \subseteq I \Leftrightarrow h_i \in I \; i = 1, \dots, l$

□

The proof can be found in [2].

The second statement implies that we can compute in R_2^n pretty much as we compute in Z_p, e.g. to multiply two elements of Z_p we can do the following:

$$a, b \in Z_p \quad a \odot b = \mathrm{mod}(ab, p)$$

In R_p^n we do the following:

$$q, r \in R_p^n \quad q \odot r = \mathrm{Rem}(qr, \{x_1^p - x_1, \ldots, x_n^p - x_n\})$$

By using first statement of the theorem we see that

$$(g) \subseteq (f) \text{ in } R_p^n \Leftrightarrow \mathrm{Rem}(g, G) = 0$$

where $G = \mathrm{GB}(\{f, x_1^p - x_1, \ldots, x_n^p - x_n\})$.

Mathematica also has Gröbner basis routines, although the documentation of these far from complete. There is an explicit command,

```
GroebnerBasis[ {p1, ... , pn}, {x1, ... , xn} ]
```

but if we want to compute remainders the following command does the right thing:

```
AlgebraicRules[ {p1, ... , pn}, {x1, ... , xn} ]
```

The remainder is computed through:

```
Rem(q, G) = q /. AlgebraicRules[ G, X ]
```

where $G = \{g_1, \ldots, g_k\}$ and $X = \{x_1, \ldots, x_n\}$.

The following example will illustrate the computation more clearly:

Example 7.2 *Suppose we want to prove*

$$\theta = (\phi \rightarrow \psi) \rightarrow (\phi \rightarrow \sigma) \vdash \phi \rightarrow (\psi \rightarrow \sigma) = \omega$$

According to theorem 5.1 this equivalent to

$$(P(\theta)) \subseteq (P(\omega)) \text{ in } R_2^n$$

which translates to the computation:

$$\mathrm{Rem}(P(\omega), \mathrm{GB}(\{P(\theta), x_1^2 + x_1, \ldots, x_n^2 + x_n\}) = 0 \tag{43}$$

In Mathematica we get:

```
In[1]:= (* phi -> x1, psi -> x2, sigma -> x3 *)

In[2]:= G = AlgebraicRules[{    x1^2 + x1 == 0,
                                x2^2 + x2 == 0,
                                x3^2 + x3 == 0,
                                Modulus   == 2,
                                P[Implies[x1,Implies[x2,x3]]]==0},
                           {    x1, x2, x3                       },
                           Mode -> Modular                            ]

Out[2]= {-2 -> 0, x3 -> 0, x2 -> -1, x1 -> -1}

In[3]:= P[ Implies[ Implies[ x1,x2 ], Implies[ x1,x3 ] ] ] /. G

Out[3]= 0
```

where `In[3]` *does the computation of equation (43).*

Finally we should of course package the two approaches into single commands:

$$\Gamma \models \phi \Leftrightarrow \texttt{ModelsQ}[\Gamma, \phi, \texttt{x}]$$

and

$$\Gamma \vdash \phi \Leftrightarrow \texttt{EntailsQ}[\Gamma, \phi, \texttt{x}]$$

Example 7.3 *This is just using the previous functions so we try some arbitrary statements The calculation below proves:*

$$\begin{array}{rcl} \phi & \vdash & \neg(\neg\phi \wedge \neg\psi) \\ \{\neg(\phi \wedge \neg\psi), \phi\} & \vdash & \psi \\ \neg\phi & \vdash & (\phi \rightarrow \psi) \leftrightarrow \neg\phi \end{array}$$

```
In[1]:= EntailsQ[ { phi }, ! ( ! phi && ! psi), {phi, psi} ]

Out[1]= True

In[2]:= EntailsQ[ { ! ( phi && ! psi ), phi }, psi, {phi, psi} ]

Out[2]= True

In[3]:= EntailsQ[ { ! phi }, Implies[ phi, psi ] == ! phi, {phi, psi} ]

Out[3]= True
```

In a similar vein we can prove the following:

$$\begin{array}{rcl} \phi & \models & \neg(\neg\phi \wedge \neg\psi) \\ \{\neg(\phi \wedge \neg\psi), \phi\} & \models & \psi \\ \neg\phi & \models & (\phi \rightarrow \psi) \leftrightarrow \neg\phi \end{array}$$

```
In[4]:= ModelsQ[ { phi }, ! ( ! phi && ! psi), {phi, psi} ]

Out[4]= True

In[5]:= ModelsQ[ { ! ( phi && ! psi ), phi }, psi, {phi, psi} ]

Out[5]= True

In[6]:= ModelsQ[ { ! phi }, Implies[ phi, psi ] == ! phi, {phi, psi} ]

Out[6]= True
```

8 Future Work

This report basically presents a new method to do logic proofs, namely through ideal inclusions. This in turn translates to Gröbner basis and polynomial remainder calculations. It is certainly interesting from a complexity point of view to further examine this method. Indeed it is shown in Hopcroft & Ullman [7] that the *satisfiability problem*, i.e. wether or not there exist a valuation such that a logic formula is true, is *NP*-complete. The satisfiability problem is thus equivalent to wether or not

$$V(q(x_1, \ldots, x_n)) = Z_p^n$$

which is equivalent to wether or not

$$(q) = (0)$$

Since the values used as truth values really doesn't matter this problem is also equivant to

$$V(q) = \emptyset \Leftrightarrow (q) = (1)$$

This complexity result provides lower bounds on the complexity of the method (assuming $P \neq NP$ of course). So what really matters then is performance on real problems.

Another interesting path to follow would be to extend this to multivalued logic. This is probably a very simple exercise by mimicing the methods used in this report. For instance 3 valued logic should be mapped to Z_3^n etc.

A Mathematica Code

This is a complete listing of the functions used in the examples. These may be obtained through e-mail upon request to the author (`roger@isy.liu.se`).

Translating between formats, i.e. polynomials and propositional logic.

```
(* Boolean Expr To Polynomial *)

P[ True               ] := 1
P[ False              ] := 0
P[ e_Symbol           ] := e
P[ Not[e_]            ] := 1 + P[e]
P[ Implies[e1_,e2_]   ] := 1 + P[e1] + P[e1] P[e2]
P[ And[e1_,e2_]       ] := P[e1] P[e2]
P[ Or[e1_,e2_]        ] := P[e1] + P[e2] + P[e1] P[e2]
P[ Equal[e1_,e2_]     ] := 1 + P[e1] + P[e2]

(* Polynomial Expr To Boolean *)

B[ 1                  ] := True
```

```
B[ 0                 ] := False
B[ e_Symbol          ] := e
B[ Times[e1_,e2_]    ] := B[e1] && B[e2]
B[ Plus[e1_,e2_]     ] := (! B[e1] && B[e2]) || (B[e1] && ! B[e2])
B[ Power[e_,n_]      ] := B[ e B[e^(n-1)] ]  /; n > 1
B[ Power[e_,1]       ] := B[ e ]
```

Computing truth tables:

```
(* Truth tables *)

VRule[ v_, w_ ] := Map[ Rule[#[[1]],#[[2]]]&, Transpose[{v,w}] ]

(* Boolean *)

T[1] = {{False},{True}};
T[n_] := T[n] =
        Join[
                Map[Prepend[#,False]&,T[n-1]],
                Map[Prepend[#,True]&,T[n-1]]
        ]

BTable[ e_, v_, set_ ] :=
        Prepend[
                        Map[    Append[#, e /. VRule[v,#]]&, set ],
                        Append[v,e]
        ]

BTable[ e_, v_ ] := BTable[ e,v, T[Length[v]] ]

BTable[ {e1_, e2__}, v_ ] :=
        Transpose[ Append[
                        Transpose[ BTable[{e2},v] ],
                        Last[ Transpose[ BTable[e1,v] ] ]
        ]]

BTable[ {e_}, v_ ] := BTable[ e, v ]

(* Mod 2 *)

Z[1] = {{0},{1}};
Z[n_] := Z[n] =
        Join[
                Map[Prepend[#,0]&,Z[n-1]],
                Map[Prepend[#,1]&,Z[n-1]]
        ]

ZTable[e_,v_,set_] :=
        Prepend[
                        Map[    Append[#, Mod[ e /. VRule[v,#], 2 ]]&, set ],
                        Append[v,e]
        ]
```

```
ZTable[e_,v_] := ZTable[e,v,Z[Length[v]]]

ZTable[{e1_,e2__}, v_] :=
        Transpose[ Append[
                Transpose[ZTable[e2,v]],
                Last[Transpose[ ZTable[e1,v] ] ]
        ]]

ZTable[ {e_}, v_] := ZTable[ e, v ]

BToZ[ tab_ ] := tab /. {True->1, False->0}
```

Proof system based on either geometry or ideals:

```
ModelsQ[ G_, f_, x_] :=
        Block[{ gtab, ftab, gp, fp },
                gp = P[ And @@ G ];
                fp = P[ f ];
                gtab = Rest @ (Last /@ ZTable[ gp, x ]);
                ftab = Rest @ (Last /@ ZTable[ fp, x ]);
                gpos = Flatten @ Position[gtab, 1];
                fpos = Flatten @ Position[ftab, 1];
                SubsetQ[ gpos, fpos ]
        ]

SubsetQ[ a_List, b_List ] := Intersection[ a, b ] == a

EntailsQ[ G_, f_, x_ ] :=
        Block[{ gp, fp, gp0, fp0, R, Rn2},
                Rn2 = (#^2 + # == 0)& /@ x;
                R   = AlgebraicRules[ Join[     Rn2,
                                                {Modulus == 2,
                                                P[ f ] == 0}    ],
                                        x, Mode->Modular ];
                (P[ And @@ G ] /. R) === 0
        ]

EntailsQ[ {}, f_, x_ ] :=
        Block[{ gp, fp, gp0, fp0, R },
                R = Ring[2,x];
                (P[ f ] /. R) === 1
        ]

TheoremQ[ f_, x_ ] := EntailsQ[ {}, f, x ]

Ring[ p_, x_ ] :=
        AlgebraicRules[ Append[ (#^p - # == 0)& /@ x, Modulus == p ],
                        x, Mode->Modular                                ]
```

References

[1] M.F. Atiyah and I.G. MacDonald. *Introduction to Commutative Algebra.* Addison-Wesley, 1969.

[2] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N.K. Bose, editor, *Multidimensional Systems Theory*, pages 184–232. Dordrecht Reidel, 1985.

[3] B. Char, K.O. Geddes, G.H. Gonnet, M.B. Monagan, and S.M. Watt. *Maple Reference Manual.* Symbolic Computation Group, Univ. of Waterloo, fifth edition, March 1988.

[4] R. Germundsson. Basic results on ideals and varieties in finite fields. Report, Dept of EE, 1991. To be published.

[5] A.C. Hearn. *Reduce User's Manual.* Rand Corp., Santa Monica, 1987.

[6] I.N. Herstein. *Topics in Algebra.* Wiley, 1974.

[7] J.D. Ullman J.E. Hopcroft. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[8] S. Lang. *Algebra.* Addison-Wesley, 1984.

[9] H. Matsumura. *Commutative Ring Theory.* Cambridge Studies in Advanced Mathematics 8. Cambridge University Press, 1986.

[10] D. Mumford. *The Red Book of Varieties and Schemes*, volume 1388 of *Lect. Notes in Math.* Springer, 1988.

[11] F. Pauer and M. Pfeifhofer. The theory of Gröbner bases. *L'Enseignement Mathématique*, 34:215–232, 1988.

[12] M. Reid. *Undergraduate Algebraic Geometry*, volume 12 of *London Mathematical Society Student Texts.* Cambridge University Press, 1988.

[13] L. Robbiano. Gröbner bases: a foundation for commutative algebra. Internal. Email: robbiano@igecuniv (bitnet).

[14] I.R. Shafarevich. *Basic Algebraic Geometry.* Springer, 1974.

[15] Symbolics Inc. *Macsyma Reference Manual*, November 1988. Version 13.

[16] D. van Dalen. *Logic and Structure.* Universitext. Springer Verlag, 1980.

[17] S. Wolfram. *Mathematica. A System for Doing Mathematics by Computer.* Addison-Wesley, second edition, 1991.