Simplification modulo equational axioms using Gröbner-bases

Visa Nummelin

November 1, 2019

The problem to solve

- Goal: Make tactic like *ring* which takes local equations into account
- E.g. prove $xy = 0$ given that $x + y = z$ and $x^2 + y^2 = z^2$

The problem to solve

- Goal: Make tactic like *ring* which takes local equations into account
- E.g. prove $xy = 0$ given that $x + y = z$ and $x^2 + y^2 = z^2$
- Normal forms: as with *ring*—like one writes down a polynomial—modulo axioms
- Handling of assumptions: Gröbner-basis will allow to find the normal form
- Terms are polynomials whose indeterminates are the variables in the problem (e.g. x, y, z above)

Formulation by ideals

Definitions

Let R be a ring. An ideal J of R is a subgroup of $(R,+)$ closed under multiplication by elements of R. In other words, non-empty $J \subseteq R$ is an ideal iff $RJ + JR = J$.
An ideal $\langle S \rangle$ generated by $S \subseteq R$ is the smallest one containing S.
Explicitly when R commutes

$$\langle S \rangle = \left\{ \sum_{j=1}^{n} r_j s_j \,\middle|\, r_j \in R,\ s_j \in S,\ 1 \leq j \leq n \in \mathbb{N} \right\}$$

- Ideals represent equations: $\langle x - y \rangle$ is the set of 0-elements, if $x = y$ were to hold
- E.g. in the language of ideals one asks:
  Does $xy \in \langle x + y - z,\ x^2 + y^2 - z^2 \rangle$?
- Naturally this is a question in ring $R[x,y,z]$ where R comes from the problem
- The use of Gröbner-bases requires R to be a field

Idea: simplify

- Example problem: $xy \in \langle x + y - z,\ x^2 + y^2 - z^2 \rangle$

Idea: simplify

- Example problem: $xy \in \langle x + y - z,\ x^2 + y^2 - z^2 \rangle$
- Let y and z be simpler than x
  $\Longrightarrow$ $x + y - z$ becomes a simplification rule $x \to z - y$

Idea: simplify

- Example problem: $xy \in \langle x + y - z,\ x^2 + y^2 - z^2 \rangle$
- Let y and z be simpler than x
  $\implies$ $x + y - z$ becomes a simplification rule $x \to z - y$
- $xy = (z - y)y = zy - y^2$ and
  $x^2 + y^2 - z^2 = z^2 - 2yz + y^2 + y^2 - z^2 = 2y^2 - 2yz$

Idea: simplify

- Example problem: $xy \in \langle x + y - z,\ x^2 + y^2 - z^2 \rangle$
- Let y and z be simpler than x
  $\implies$ $x + y - z$ becomes a simplification rule $x \to z - y$
- $xy = (z - y)y = zy - y^2$ and
  $x^2 + y^2 - z^2 = z^2 - 2yz + y^2 + y^2 - z^2 = 2y^2 - 2yz$
- Result: $zy - y^2 \in \langle \ldots,\ 2y^2 - 2yz \rangle$

Idea: simplify

- Example problem: $xy \in \langle x+y-z,\ x^2+y^2-z^2 \rangle$
- Let y and z be simpler than x
  $\Longrightarrow$ $x+y-z$ becomes a simplification rule $x \to z-y$
- $xy = (z-y)y = zy - y^2$ and
  $x^2+y^2-z^2 = z^2-2yz+y^2+y^2-z^2 = 2y^2-2yz$
- Result: $zy - y^2 \in \langle \ldots,\ 2y^2-2yz \rangle$
- Let z be simpler than y
  $\Longrightarrow$ $2y^2-2yz$ becomes a simplification rule $2y^2 \to 2yz$

Idea: simplify

- Example problem: $xy \in \langle x+y-z,\ x^2+y^2-z^2 \rangle$
- Let y and z be simpler than x
  $\Longrightarrow$ $x+y-z$ becomes a simplification rule $x \to z-y$
- $xy = (z-y)y = zy - y^2$ and
  $x^2+y^2-z^2 = z^2-2yz+y^2+y^2-z^2 = 2y^2-2yz$
- Result: $zy - y^2 \in \langle \ldots,\ 2y^2-2yz \rangle$
- Let z be simpler than y
  $\Longrightarrow$ $2y^2-2yz$ becomes a simplification rule $2y^2 \to 2yz$
- $zy - y^2 = zy - {}^1\!/_2(2zy) = 0$

Idea: simplify

- Example problem: $xy \in \langle x+y-z,\ x^2+y^2-z^2 \rangle$
- Let y and z be simpler than x
  $\implies$ $x+y-z$ becomes a simplification rule $x \to z-y$
- $xy = (z-y)y = zy - y^2$ and
  $x^2+y^2-z^2 = z^2-2yz+y^2+y^2-z^2 = 2y^2-2yz$
- Result: $zy - y^2 \in \langle \ldots,\ 2y^2-2yz \rangle$
- Let z be simpler than y
  $\implies$ $2y^2-2yz$ becomes a simplification rule $2y^2 \to 2yz$
- $zy - y^2 = zy - {}^1\!/_2(2zy) = 0$
- Result: $0 \in \langle \ldots \rangle$—certainly true!

Simplicity order

- Let M be the least simple monomial of P
  $\Longrightarrow$ P becomes a simplification rule $M \to M - P$
- Some properties are required to admit an order as simplifying

Definition

An order $\leq$ on monomials is admissible if

1. it is linear,
2. $M_1 \leq M_2 \Longrightarrow NM_1 \leq NM_2$ for all monomials N, M_1, M_2, and
3. it is well-founded.

- Given 1. and 2., well-foundedness $\Leftrightarrow$ monomial 1 is the bottom
- Well-foundedness ensures that normal forms exist

Common orderings

- Monomials of ring $K[X_1, ..., X_n]$ are isomorphic to $\mathbb{N}^n$ (consider exponent sequences; $\times$ becomes $+$)

Common orderings

- Monomials of ring $K[X_1, ..., X_n]$ are isomorphic to $\mathbb{N}^n$ (consider exponent sequences; $\times$ becomes $+$)
- Lexicographic order
- Reverse lexicographic order and other permutations

Common orderings

- Monomials of ring $K[X_1, \ldots, X_n]$ are isomorphic to $\mathbb{N}^n$ (consider exponent sequences; $\times$ becomes $+$)
- Lexicographic order (informative)
- Reverse lexicographic order and other permutations
- Total degree lex. order (economic)
  - primarily compare total degree $\deg \prod_j X_j^{e_j} = \sum_j e_j$
  - secondarily compare lexicographically

Common orderings

- Monomials of ring $K[X_1, \ldots, X_n]$ are isomorphic to $\mathbb{N}^n$ (consider exponent sequences; $\times$ becomes $+$)
- Lexicographic order (informative)
- Reverse lexicographic order and other permutations
- Total degree lex. order (economic)
  - primarily compare total degree $\deg \prod_j X_j^{e_j} = \sum_j e_j$
  - secondarily compare lexicographically
- Orders built in stages
  - e.g. lexicographically X_1, then deg.lex. $(X_2, \ldots, X_n)$
- Instead of exponent sum, use max, min, weighted sum, power sum, ...

Common orderings

- Monomials of ring $K[X_1, \ldots, X_n]$ are isomorphic to $\mathbb{N}^n$ (consider exponent sequences; $\times$ becomes $+$)
- Lexicographic order (informative)
- Reverse lexicographic order and other permutations
- Total degree lex. order (economic)
  - primarily compare total degree $\deg \prod_j X_j^{e_j} = \sum_j e_j$
  - secondarily compare lexicographically
- Orders built in stages
  - e.g. lexicographically X_1, then deg.lex. $(X_2, \ldots, X_n)$
- Instead of exponent sum, use max, min, weighted sum, power sum, ...
- From now on, consider a fixed admissible order

Gröbner-bases

- Just simplifying is not enough: consider
  $x^2 - y \in \langle xy^2 - x - y,\ x^2y - x - 1 \rangle$

Definitions

Fix $B \subseteq K[X_1, \ldots, X_n]$. An associated rewrite/simplification relation is defined by

$$P + NM \to P - N(R - M)$$

if P lacks multiple of monomial NM and M is least simple in $R \in B$.
Let $\overset{*}{\to}$ be transitive-reflexive- and $\overset{*}{\leftrightarrow}$ be equivalence-closure of $\to$.

Lemma

$P = R \mod \langle B \rangle$ *iff* $P \overset{*}{\leftrightarrow} R$.

Especially $P \in \langle B \rangle$ *iff* $P \overset{*}{\leftrightarrow} 0$.

- We want to only check that $P \overset{*}{\to} 0$.

Definitions

Rewriting $\to$ is confluent if $P \overset{*}{\leftarrow} \circ \overset{*}{\to} R$ implies $P \overset{*}{\to} \circ \overset{*}{\leftarrow} R$.
B is Gröbner-basis of $\langle B \rangle$ if $\to$ is confluent.

Recognising Gröbner-basis

Theorem
As a terminating rewrite relation $\to$ is confluent if $P \leftarrow \circ \to R$ implies $P \overset{*}{\to} \circ \overset{*}{\leftarrow} R$.

- A further pinpointing of possible failure leads to S-polynomials
- Let M be least simple (aka leading) monomial of $M+P$ and likewise for N and $N+R$.
- $-NP \leftarrow NM \to -MR$ as well as after dividing by $\gcd\{M,N\}$
- $S(M+P,N+R) := \dfrac{NP-MR}{\gcd\{M,N\}}$

Theorem
If $S(P,R) \overset{*}{\to} 0$ for all $P,R \in B$, then B is a Gröbner-basis.

Buchberger's algorithm

1. Given B, find a Gröbner-basis of $\langle B \rangle$
2. Compute S-polynomials between pairs of polynomials in B
3. Simplify them
4. Add to B the non-zero simplified ones
5. Repeat from step 2 until B stabilizes
6. Simplify elements in B by each others

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}:\ -x - 1 - y(-y) = y^2 - x - 1$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}:\ -x - 1 - y(-y) = y^2 - x - 1$
- Next and final round: 7 new S-polynomials reduce to 0

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- S : $x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- S' : $x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}$: $-x - 1 - y(-y) = y^2 - x - 1$
- Next and final round: 7 new S-polynomials reduce to 0

- With simplification after S: $\{xy^2 - x - y,\ x^2y - x - 1,\ x^2 - y\}$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}:\ -x - 1 - y(-y) = y^2 - x - 1$
- Next and final round: 7 new S-polynomials reduce to 0

- With simplification after S: $\{xy^2 - x - y,\ x^2y - x - 1,\ x^2 - y\}$
- $x^2y - x - 1 \to y^2 - x - 1$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}:\ -x - 1 - y(-y) = y^2 - x - 1$
- Next and final round: 7 new S-polynomials reduce to 0

- With simplification after S: $\{xy^2 - x - y,\ x^2y - x - 1,\ x^2 - y\}$
- $x^2y - x - 1 \to y^2 - x - 1$
- $xy^2 - x - y \to (x^2 + x) - x - y \to y - y = 0$

Example run

- Given $\{xy^2 - x - y,\ x^2y - x - 1\}$, use deg.lex. with $x > y$
- $S:\ x(-x-y) - y(-x-1) = -(x^2 - y)$
- $\{xy^2 - x - y,\ \underline{x^2y} - x - 1,\ \underline{x^2} - y\}$
- $S':\ x(-x-y) - y^2(-y) \to y^3 - xy - y$
- $\underline{S}:\ -x - 1 - y(-y) = y^2 - x - 1$
- Next and final round: 7 new S-polynomials reduce to 0

- With simplification after S: $\{xy^2 - x - y,\ x^2y - x - 1,\ x^2 - y\}$
- $x^2y - x - 1 \to y^2 - x - 1$
- $xy^2 - x - y \to (x^2 + x) - x - y \to y - y = 0$

- Result: $\{x^2 - y,\ y^2 - x - 1\}$

Representing polynomials in Lean

- The *mv_polynomial* might be converted to noncomputable
- Hence must implement own custom data type

Representing polynomials in Lean

- The *mv_polynomial* might be converted to noncomputable
- Hence must implement own custom data type
- Operations on monomials needed to compute Gröbner-bases:
  - order comparison
  - gcd, lcm
  - division
- List of exponents

- Mapping from indeterminates to exponents

Representing polynomials in Lean

- The *mv_polynomial* might be converted to noncomputable
- Hence must implement own custom data type
- Operations on monomials needed to compute Gröbner-bases:
  - order comparison
  - gcd, lcm
  - division
- List of exponents
  simpler and often more (space) efficient
- Mapping from indeterminates to exponents
  allows idiomatically to parameterise by type of indeterminates

Representing polynomials in Lean

- The *mv_polynomial* might be converted to noncomputable
- Hence must implement own custom data type
- Operations on monomials needed to compute Gröbner-bases:
  - order comparison
  - gcd, lcm
  - division
- List of exponents—my bad choice
  simpler and often more (space) efficient

- Mapping from indeterminates to exponents
  allows idiomatically to parameterise by type of indeterminates

Representing polynomials in Lean

- Operations of polynomials needed to compute Gröbner-bases:
  - addition
  - multiplication (by monomial)
  - determine leading (least simple) monomial
  - test for 0

Representing polynomials in Lean

- Operations of polynomials needed to compute Gröbner-bases:
  - addition
  - multiplication (by monomial)
  - determine leading (least simple) monomial
  - test for 0
- Tensor / nD-matrix of coefficients
- List of (monomial, coefficient)-pairs
- Tree of (monomial, coefficient)-pairs
- Hash map from monomials to coefficients
- Single variate recursive via $K[\vec{X}, Y] \cong K[\vec{X}][Y]$

Representing polynomials in Lean

- Operations of polynomials needed to compute Gröbner-bases:
  - addition
  - multiplication (by monomial)
  - determine leading (least simple) monomial
  - test for 0
- Tensor / nD-matrix of coefficients
  wastes space: $\sum_{j=1}^{n} X_j^2$ needs exponential $\mathcal{O}(2^n)$ memory
- List of (monomial, coefficient)-pairs
  efficient multiplication demands extra temporary structures
- Tree of (monomial, coefficient)-pairs
  most constant overhead
- Hash map from monomials to coefficients
  leading term is expensive to find
- Single variate recursive via $K[\vec{X}, Y] \cong K[\vec{X}][Y]$
  tricky in typed setting

Representing polynomials in Lean

- Operations of polynomials needed to compute Gröbner-bases:
  - addition
  - multiplication (by monomial)
  - determine leading (least simple) monomial
  - test for 0
- ~~Tensor / nD-matrix of coefficients~~
  wastes space: $\sum_{j=1}^{n} X_j^2$ needs exponential $\mathcal{O}(2^n)$ memory
- List of (monomial, coefficient)-pairs—state of art
  efficient multiplication demands extra temporary structures
- Tree of (monomial, coefficient)-pairs—my choice
  most constant overhead
- Hash map from monomials to coefficients
  leading term is expensive to find
- Single variate recursive via $K[\vec{X}, Y] \cong K[\vec{X}][Y]$
  tricky in typed setting

Take a look at the code

- I used *rbtree* which is a non-meta variant of *rb_tree*
- Meta could have been used through out the code, but in the beginning it was undecided wether to also formalize the theory of Gröbner-bases

Reconstructing the proof

- Let B be the initial basis and G the derived Gröbner-basis
- We want to know that $P \xrightarrow[G]{*} R \implies P - R \in \langle B \rangle$, and $G \subseteq \langle B \rangle$

Reconstructing the proof

- Let B be the initial basis and G the derived Gröbner-basis
- We want to know that $P \xrightarrow[G]{*} R \implies P - R \in \langle B \rangle$, and $G \subseteq \langle B \rangle$
- Ideal membership can be certified by producing the coefficient polynomials $\vec{c}$
- $P - R = \vec{c} \bullet B$ can be conveniently checked by the *ring* tactic
- $\vec{c} \bullet B = 0$, given each $b \in B$ is 0, can often be checked by *simp*
  - Not failsafe
  - Proof is built manually by repeatedly applying lemma $x = y \implies O = 0 \implies O - c(x - y) = 0$
- Finally lemma $P - R = O \implies O = 0 \implies P = R$ finishes

Using the tactic

- Requiring terms to have field type would be too restrictive
- Ask for a ring and a vector space over a coefficient field
- One can test if type of an expression is ring or vector space over known field by *apply_instance*
- Testing vector spaceness over to-be-determined field is not possible this way
  $\Longrightarrow$ field must be given as a paramter to the tactic

Using the tactic

- Requiring terms to have field type would be too restrictive
- Ask for a ring and a vector space over a coefficient field
- One can test if type of an expression is ring or vector space over known field by *apply_instance*
- Testing vector spaceness over to-be-determined field is not possible this way
  $\Longrightarrow$ field must be given as a paramter to the tactic

- Good to find normal forms other than 0 too
- E.g. $x = y \vdash xF(2x^2 - y^2) - yF(xy) = 0$
- Bottom-up simplification strategy (or repeat until stabilisation)