Virtual Substitution

A more efficient way to eliminate quantifiers compared to FM, Section 6.2.1, in linear rational arithmetic was developed by R. Loos and V. Weispfenning (1993).

The method is also known as *test point method* or *virtual substitution method*. In contrast to FM, the method does not require CNF/DNF transformations of a prenex formula $\{\exists, \forall\} x_1 \ldots \{\exists, \forall\} x_n . \phi$.

Let $\phi[x, \vec{y}]$ be a quantifier-free formula of linear arithmetic in negation normal form containing the free variables $x, \vec{y}$ where all negation symbols are removed. Any quantifier free formula ϕ can be effectively and equivalently transformed in this form, see Section 6.2.1 and for the removal of the operator $\neg$ rule ElimNeg.

The linear inequations in ϕ can be transformed such that x is either isolated or does not occur in the inequation: $x \circ_i s_i(\vec{y})$ and $0 \circ_j s'_j(\vec{y})$ with $\circ_i, \circ_j \in \{\approx, \not\approx, <, \leq, >, \geq\}$, that is, ϕ us a formula built from linear inequations, $\wedge$ and $\vee$.

The goal of the virtual substitution method is to identify a a finite set T of "test points", i.e., LA terms such that

$$\{\forall, \exists\}\vec{y}.\exists x.\phi[x, \vec{y}] \quad \text{iff} \quad \{\forall, \exists\}\vec{y}. \bigvee_{t \in T} \phi[x, \vec{y}]\,\{x \mapsto t\}.$$

Semantically, an existential quantifier represents an infinite disjunction over $\mathbb{Q}$. The goal of virtual substitution is to replace this infinite disjunction by a finite disjunction.

If the values of the variables $\vec{y}$ are determined by some arbitrary but fixed assignment β for the $\vec{y}$, then ϕ can be considered as a function $\phi_\beta : \mathbb{Q} \mapsto \{0, 1\}$ by

$$\phi_\beta(d) := \mathcal{A}_{\mathsf{LRA}}(\beta[x \mapsto d])(\phi)$$

for any $d \in \mathbb{Q}$. The value of each of the atoms $x \circ_i s_i[\vec{y}]$ changes only at $\mathcal{A}_{\mathsf{LRA}}(\beta)(s_i[\vec{y}])$, and the value of ϕ can only change if the value of one of its atoms changes. So ϕ_β is a piecewise constant function.

More precisely, the set of all $d \in \mathbb{Q}$ with $\phi_\beta(d) = 1$ is a finite union of intervals. The union may be empty, the individual intervals may be finite or infinite and open or closed.

Let

$$\begin{aligned}\mathsf{dist}(\phi, x, \beta) = \quad & \min\{\, |\mathcal{A}_{\mathsf{LRA}}(\beta)(s_i[\vec{y}]) - \mathcal{A}_{\mathsf{LRA}}(\beta)(s_j[\vec{y}])| \\ & \text{where } \mathcal{A}_{\mathsf{LRA}}(\beta)(s_i[\vec{y}]) \neq \mathcal{A}_{\mathsf{LRA}}(\beta)(s_j[\vec{y}]) \,\}\end{aligned}$$

the minimal distance between two differently interpreted terms of atoms $x \circ_i s_i[\vec{y}]$, $x \circ_j s_j[\vec{y}]$ in ϕ under β. Then each of the intervals has either length 0, i.e., it consists of one point, or its length is at least $\mathsf{dist}(\phi, x, \beta)$.

The set of all values $d \in \mathbb{Q}$ of $\phi_\beta(d)$ can be considered either by traversing $\mathbb{Q}$ from $-\infty$ to $+\infty$ or the other way round. In the case of traversing from $-\infty$ to $+\infty$ if the set of all d for which $\phi_\beta(d) = 1$ is non-empty, then

(i) $\phi_\beta(d) = 1$ for all $d \circ \mathcal{A}_{\mathsf{LRA}}(\beta)(r[\vec{y}])$ for some $x \circ r[\vec{y}]$ occurring in ϕ, $\circ \in \{<, \leq\}$ or

(ii) there is some value $d \in \mathbb{Q}$ where the value of $\phi_\beta(d)$ switches from 0 to 1 when traversing from $-\infty$ to $+\infty$.

This observation can be used to construct a set of test points symbolically without considering β explicitly. It is sufficient to keep in mind that the values for the $\vec{y}$ are fixed and to use then the terms from ϕ as representatives for the values from $\mathbb{Q}$.

The start is a "sufficiently small" test point $r[\vec{y}]$ to take care of case (i). For case (ii), $\phi[x, \vec{y}]$ can only switch from 0 to 1 if one of the atoms switches from 0 to 1. Recall that after the initial transformations on ϕ, only positive boolean combinations of atoms and $\wedge$ and $\vee$ are left, which are monotonic with respect to truth values.

Atoms of the form $x \leq s_i[\vec{y}]$ and $x < s_i[\vec{y}]$ do not switch from 0 to 1 when x grows.

Atoms of the form $x \geq s_i[\vec{y}]$ and $x \approx s_i[\vec{y}]$ switch from 0 to 1 at $s_i[\vec{y}]$ hence $s_i[\vec{y}]$ is a test point.

Atoms of the form $x > s_i[\vec{y}]$ and $x \not\approx s_i[\vec{y}]$ switch from 0 to 1 "right after" $s_i[\vec{y}]$, hence $s_i[\vec{y}] + \varepsilon$ for some $0 < \varepsilon < \delta(\vec{y})$ is a test point.

If $r[\vec{y}]$ is sufficiently small and $0 < \varepsilon < \delta(\vec{y})$, then

$$T := \{r[\vec{y}]\} \cup \{\, s_i[\vec{y}] \mid \circ_i \in \{\geq, =\}\,\} \cup \{\, s_i[\vec{y}] + \varepsilon \mid \circ_i \in \{>, \neq\}\,\}.$$

is a set of test points for atoms $x \circ_i s_i[\vec{y}]$.

However, it is not known how small $r[\vec{y}]$ has to be for case (i), and $\delta(\vec{y})$ for case (ii) is not known as well, because it is not effectively possible to consider all, infinitely many β explicitly.

The idea out the problem is to extend the LA language by further symbols ∞, and ε with the obvious intended meanings. Now it is straightforward to define T independently of β.

$$\begin{aligned} T := \{-\infty\} \cup \{\, s_i[\vec{y}] \quad\quad\quad\quad \mid \circ_i \in \{\geq, =\}\,\} \\ \cup \{\, s_i[\vec{y}] + \varepsilon \mid \circ_i \in \{>, \neq\}\,\}. \end{aligned}$$

But the semantics of LA is not defined with respect to the infinitesimals ∞, ε and all considerations leading to the above set T do not hold anymore, if ϕ contains occurrences of ∞ or ε.

Fortunately, the infinitesimals ∞ and ε vanish when substituted for some variable x.

$$
\begin{aligned}
(x < s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r < s(\vec{y})) = \top \\
(x \leq s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r \leq s(\vec{y})) = \top \\
(x > s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r > s(\vec{y})) = \bot \\
(x \geq s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r \geq s(\vec{y})) = \bot \\
(x \approx s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r \approx s(\vec{y})) = \bot \\
(x \not\approx s(\vec{y}))\{x \mapsto -\infty\} &:= \lim_{r \to -\infty} (r \not\approx s(\vec{y})) = \top
\end{aligned}
$$

$$
\begin{aligned}
(x < s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon < s(\vec{y})) = (u < s(\vec{y})) \\
(x \leq s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon \leq s(\vec{y})) = (u < s(\vec{y})) \\
(x > s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon > s(\vec{y})) = (u \geq s(\vec{y})) \\
(x \geq s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon \geq s(\vec{y})) = (u \geq s(\vec{y})) \\
(x \approx s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon \approx s(\vec{y})) = \bot \\
(x \not\approx s(\vec{y}))\,\{x \mapsto u + \varepsilon\} &:= \lim_{\varepsilon \to 0}(u + \varepsilon \not\approx s(\vec{y})) = \top
\end{aligned}
$$

The above test point set is constructed by considering a traversal of possible values for x from $-\infty$ to $+\infty$. Alternatively, x can be traversed from $+\infty$ to $-\infty$. In this case, the test points are

$$\begin{aligned} T' := \{+\infty\} \cup \{\, s_i[\vec{y}] \quad\quad\quad\quad \mid \circ_i \in \{\leq, =\} \,\} \\ \cup \{\, s_i[\vec{y}] - \varepsilon \mid \circ_i \in \{<, \neq\} \,\}. \end{aligned}$$

Infinitesimals are eliminated in a similar way as before.

In practice, both sets T and T' and eventually the smaller formula after substitution and simplification is considered. Similar to the FM decision procedure for formulas, a universally quantified formula $\forall x.\phi$, is replaced by $\neg\exists x.\neg\phi$. Then the inner negation is pushed downwards, and then the test point procedure is applied as in the case of an existential quantifier.

Note that in contrast to the FM procedure, no CNF/DNF transformation is required. Loos-Weispfenning quantifier elimination works on arbitrary positive formulas. So the CNF/DNF conversion blow up caused in FM quantifier elimination does not happen for virtual substitution. Therefore, the worst-case complexity of Loos-Weispfenning quantifier elimination significantly improves upon the worst-case complexity of FM.

However, the cost of computing a negation normal form remain.

Virtual Substitution Complexity

The number of test points is at most half of the number of atoms for some formula ϕ with $|\phi| = n$, so the formula resulting from the elimination of one variable, independent from the type of the quantifier, is at most quadratic, therefore $O(n^2)$ runtime.
A sequence of m quantifiers of the same kind, results in a multiplication of the formula size with n in each step, therefore $O(n^{m+1})$ runtime. This is the result of distributing existential quantifiers over disjunctions.

$$
\begin{aligned}
& \exists x_2\, \exists x_1.\, \phi[x_1, x_2, \vec{y}] \\
\leftrightarrow\quad & \exists x_2.\, \left(\bigvee_{t_1 \in T_1} \phi[x_1, x_2, \vec{y}]\, \{x_1 \mapsto t_1\} \right) \\
\leftrightarrow\quad & \bigvee_{t_1 \in T_1} \left(\exists x_2.\, \phi[x_1, x_2, \vec{y}]\, \{x_1 \mapsto t_1\} \right) \\
\leftrightarrow\quad & \bigvee_{t_1 \in T_1} \bigvee_{t_2 \in T_2} \left(\phi[x_1, x_2, \vec{y}]\, \{x_1 \mapsto t_1\}\, \{x_2 \mapsto t_2\} \right)
\end{aligned}
$$

A sequence of m quantifier alternations $\exists\forall\exists\forall\ldots\exists$ turns the top-level disjunction after moving the inner negation into a top-level conjunction. An existential quantifier does not distribute over a conjunction, so the procedure needs $O(n^2)$ runtime for each step, therefore doubly exponential runtime in sum, $O(n^{2^m})$.

Cooper's Algorithm for LIA

Cooper' algorithm is a quantifier elimination algorithm for linear integer arithmetic (LIA), similar to virtual substitution [Cooper 72]. Actually, one can view virtual substitution as an application of Cooper's ideas to linear rational arithmetic.

A first consequence of the integer domain is that strict and non-strict inequations can be easily exchanged, e.g., $t \leq c$ iff $t < c + 1$.

6.2.12 Definition (LIA Syntax)

The syntax of LIA is

$$\Sigma_{\mathsf{LIA}} = (\{\mathsf{LIA}\}, \{0, 1, +, -\} \cup \mathbb{Z}, \{\leq, <, \mid, \nmid, >, \geq\})$$

where $-$ is unitary and all other symbols have the usual arities. The bar $\mid$ is the devisability operator between a positive integer constant d and a term t, i.e., it generates atoms of the form $d \mid t$.

6.2.13 Definition (Linear Integer Arithmetic Standard Semantics)

The Σ_{LIA} algebra $\mathcal{A}_{\mathsf{LIA}}$ is defined by $\mathsf{LA}^{\mathcal{A}_{\mathsf{LIA}}} = \mathbb{Z}$ and all other signature symbols are assigned the standard interpretations over the integers.

Cooper showed that given two strict inequations $x < t$ and $x > s$, and a divisibility constraint $d \mid x$ the variable x can be eliminated in the following way:

$$\exists x.(x < t \wedge x > s \wedge d \mid x) \quad \text{iff} \quad \bigvee_{i=1}^{i \leq |d|} (s + i < t \wedge d \mid s + i)$$

This needs to be further generalized to cope with $\not|$, multiple inequations, and divisibility constraints for some variable x.

Let $\exists x.\phi$ be a formula of LIA, where ϕ is in negation normal form, ϕ does not contain any quantifiers nor negation symbols, and the LIA relations occurring in ϕ are $\{<, >, \mid, \nmid\}$. Any LIA formula can be a a effectively transformed into this form. Furthermore, for all inequations $cx \circ t$ and divisibility atoms $a \circ' bx + s$, $\circ \in \{<, >\}$, $\circ' \in \{\mid, \nmid\}$, I assume $c = 1$, $b = 1$.

If c is negative for some inequation it is multiplied by -1 and then transformed into its strict form. If b is negative, for divisibility atoms it is sufficient to multiply the right hand side by -1.

If there are atoms

$$\begin{array}{rcl} c_i x & \circ_i & t_i \\ a_j & \circ'_j & b_j x + s_j \end{array}$$

in ϕ with $c_i > 1$ or $b_j > 1$ for some i, j, $\circ_i \in \{<, >\}$, $\circ'_j \in \{\mid, \nmid\}$, then the lcm d of the c_i, b_j is computed.

The atoms are first replaced by

$$\begin{array}{lll} dx & \circ_i & \frac{d}{c_i} t_i \\ \frac{d}{b_j} a_j & \circ'_j & dx + \frac{d}{b_j} s_j \end{array}$$

respectively, and finally they are replaced by

$$\begin{array}{rcl} x & \circ_i & \frac{d}{c_i} t_i \\ \frac{d}{b_j} a_j & \circ'_j & x + \frac{d}{b_j} s_j \\ d & \mid & x \end{array}$$

respectively, where the divisibility constraint $d \mid x$ is added conjunctively to ϕ.

Similar to the arguments for composing the virtual substitution test points, solutions for $\exists x.\phi$ can be considered from $-\infty$ to ∞ or the other way round. I explain the former, the latter is then a standard exercise. Let $x < t_i$, $x > s_j$, $a_k \mid x + r_k$, $b_h \nmid x + l_h$ be all atoms in ϕ containing x where the t_i, s_j, r_k, l_h do not contain x. Let $p_1, \ldots, p_n$ be the positions of the atoms $x < t_i$ in ϕ and $q_1, \ldots, q_o$ be the positions of the atoms $x > s_j$ in ϕ. Let d be the lcm of the a_k, b_h. Then

$$\mathcal{A}_{\mathsf{LIA}}(\beta) \models \exists x.\phi$$

iff

$$\mathcal{A}_{\mathsf{LIA}}(\beta) \models \bigvee_{m=1}^{m \leq d} \phi[\top]_{p_1,\ldots,p_n}[\bot]_{q_1,\ldots,q_o}\{x \mapsto m\} \vee \bigvee_{m=1}^{m \leq d} \bigvee_{s_j} \phi\{x \mapsto s_j + m\}$$

Complexity of Cooper's Algorithm

The worst-case complexity of Cooper's variable elimination procedure is immense. The lcm d computed for each eliminated variable x grows worst-case exponentially in the size of ϕ. Thus also the formula after eliminating x is exponentially larger then ϕ. The overall runtime is again non-elementary, similar to FM quantifier elimination. The formulas resulting from the test points $-\infty$ and $s_j + m$ contain a lot of redundancy that can be eliminated afterwards. However, even if the formula size can be drastically reduced through redundancy elimination, in each step the exponentially growing coefficients m remain.
Due to its inherent complexity, Cooper's elimination procedure is rarely used in practice. It mainly serves as a theoretical background for more practical procedures.