QUANTIFIER ELIMINATION FOR PRESBURGER ARITHMETIC

SAV 2012 LECTURE NOTES

Minimalistic Language of Presburger Arithmetic. Consider $L = \{+\}$ and consider as $\mathcal{I}$ the set of all interpretations with domain $\mathbb{N} = \{0, 1, 2, \ldots\}$ where $+$ is interpreted as addition of natural numbers (these interpretations differ only in values for free variables). This is one definition of Presburger arithmetic over natural numbers.
This theory is very simple to describe, but it is very far from allowing quantifier-elimination. For example, it does not have a name for zero, so we cannot express $\forall x. x + y = x$. It also does not have a way to express $\exists y. x + y = z$.

Language of Presburger Arithmetic that Admits QE. We look at the theory of integers with addition.

- introduce constant for each integer constant
- to be able to restrict values to natural numbers when needed, we introduce $<$
- introduce not only addition but also subtraction
- to conveniently express certain expressions, introduce function m_K for each $K \in \mathcal{Z}$, to be interpreted as multiplication by a constant, $m_K(x) = K \cdot x$. We write m_K as $K \cdot x$
- to enable quantifier elimination from $\exists x. y = K \cdot x$ introduce for each K predicate $K|y$ (divisibility by constant)

The resulting language:

$$L = \{+, -, <\} \cup \{K \mid K \in \mathbb{Z}\} \cup \{(K \cdot \_) \mid K \in \mathbb{Z}\} \cup \{(K|\_) \mid K \in \mathbb{Z}\}$$

Normalizing Conjunctions of Literals. We consider elimination of a quantifier from a conjunction of literals (because QE from conjunction of literals suffices).
Running example:

$$\exists y. 3y - 2w + 1 > -w \wedge 2y - 6 < z \wedge 4 \mid 5y + 1$$

We first show that we can bring conjunction of literals into a simpler form.

Normal Form of Terms. All terms are built from $K, +, -, K \cdot \_$, so using standard transformations they can be represented as: $K_0 + \sum_{i=1}^{n} K_i x_i$ We call such term a linear term.

Normal Form for Literals. Relation symbols: $=$, $<$, $K|\_$.

$$\begin{aligned}
\neg(t_1 < t_2) &\text{ becomes } t_2 < t_1 + 1 \\
\neg(t_1 = t_2) &\text{ becomes } t_1 < t_2 \vee t_2 < t_1 \\
t_1 = t_2 &\text{ becomes } t_1 < t_2 + 1 \wedge t_2 < t_1 + 1 \\
\neg(K \mid t) &\text{ becomes } \bigvee_{i=1}^{K-1} K \mid t + i \\
t_1 < t_2 &\text{ becomes } 0 < t_2 - t_1
\end{aligned}$$

We obtain a disjunction of conjunctions of literals of the form $0 < t$ and $K \mid t$ where t are of the form $K_0 + \sum_{i=1}^{n} K_i \cdot x_i$

Running example:

$$\exists y. 0 < 1 - w + 3y \wedge 0 < 6 - 2y + z \wedge 4 \mid 5y + 1$$

Exposing the Variable to Eliminate. By previous transformations, we are eliminating y from conjunction $F(y)$ of $0 < t$ and $K \mid t$ where t is a linear term. To eliminate $\exists y$ from such conjunction, we wish to ensure that the coefficient next to y is one or minus one. Observation: $0 < t$ is equivalent to $0 < ct$ and $K \mid t$ is equivalent to $cK \mid ct$ for c a positive integer.

If $K_1, \ldots, K_n$ are all coefficients next to y in the formula, let M be a positive integer such that $K_i \mid M$ for all i, $1 \leq i \leq n$ (for example, let M be the least common multiple of $K_1, \ldots, K_n$). Multiply each literal where y occurs in subterm $K_i y$ by constant $M/|K_i|$.

What is the coefficient next to y in the resulting formula? either M or $-M$

We obtain a formula of the form $\exists y.F(My)$. Letting $x = My$, we conclude the formula is equivalent to $\exists x.F(x) \wedge (M \mid x)$.

What is the coefficient next to y in the resulting formula? either 1 or -1

Running example:

$$\begin{aligned}
&M = lcm(3, 2, 5) = 30 \\
&\exists y. 0 < 10 - 10w + 30y \wedge 0 < 90 - 30y + 15z \wedge 24 \mid 30y + 6 \\
&\exists x. 0 < 10 - 10w + x \wedge 0 < 90 - x + 15z \wedge 24 \mid x + 6 \wedge 30|x
\end{aligned}$$

Lower and upper bounds: Consider the coefficient next to x in $0 < t$. If it is -1, move the term to left side. If it is 1, move the remaining terms to the left side. We obtain formula $F_1(x)$ of the form

$$\bigwedge_{i=1}^{L} a_i < x \wedge \bigwedge_{j=1}^{U} x < b_j \wedge \bigwedge_{i=1}^{D} K_i \mid (x + t_i)$$

If there are no divisibility constraints ($D = 0$), what is the formula equivalent to?

$$\max_i a_i + 1 \leq \min_j b_j - 1 \text{ which is equivalent to } \bigwedge_{ij} a_i + 1 < b_j$$

Replacing variable by test terms: There is a an alternative way to express the above condition by replacing $F_1(x)$ with $\bigvee_k F_1(t_k)$ where t_k do not contain x. This is a common technique in quantifier elimination. Note that if $F_1(t_k)$ holds then certainly $\exists x. F_1(x)$. What are example terms t_i when $D = 0$ and $L > 0$? Hint: ensure that at least one of them evaluates to $\max a_i + 1$.

$$\bigvee_{k=1}^{L} F_1(a_k + 1)$$

What if $D > 0$ i.e. we have additional divisibility constraints?

$$\bigvee_{k=1}^{L} \bigvee_{i=1}^{N} F_1(a_k + i)$$

What is N? least common multiple of $K_1, \ldots, K_D$
Note that if $F_1(u)$ holds then also $F_1(u - N)$ holds. That's it for $L > 0$.
Running example:

$F_1 : \exists x. -10 + 10w < x \wedge x < 90 + 15z \wedge 24 \mid x + 6 \wedge 30|x$

$$\bigvee_{i=1}^{120} 10w - 10 + i < 90 + 15z \wedge 10w - 10 < 10w - 10 + i \wedge 24 \mid 10w - 10 + i + 6 \wedge 30|10w - 10 + i$$

$$\bigvee_{i=1}^{120} 10w + i < 100 + 15z \wedge 0 < i \wedge 24 \mid 10w - 4 + i \wedge 30|10w - 10 + i$$

What if $L = 0$? We first drop all constraints except divisibility, obtaining $F_2(x)$

$$\bigwedge_{i=1}^{D} K_i \mid (x + t_i)$$

and then eliminate quantifier as

$$\bigvee_{i=1}^{N} F_2(i)$$

That's it!

Example. Consider verification condition from Symbolic Execution for Example Integer Program. How can we prove such verification condition? The invariant of this code example is : $F = (res + 2i = 2x \wedge i' = i - 1 \wedge res' = res + 2) \rightarrow res' + 2i' = 2x$ We have to find out if $\neg F$ is satisfiable, i.e.

$$\exists res, res', i, i'. \neg F$$

We can eliminate quantifiers with equalities: $i' = i - 1$ and $res' = res + 2$. Then $res' + 2i'$ becomes $res + 2 + 2(i - 1)$, and $\exists i', res'$ can be removed. Finally :

$$\begin{aligned}
&\exists res, i.\neg(res + 2i = 2x \rightarrow res + 2 + 2(i - 1) = 2x) \\
&\exists res, i.\neg(res + 2i = 2x \rightarrow res + 2i = 2x) \\
&\exists res, i.\neg true \\
&\qquad false
\end{aligned}$$

Some Improvements. Avoid transforming to conjunctions of literals: work directly on negation-normal form. The technique is similar to what we described for conjunctive normal form.

This is the Cooper's algorithm:

- Reddy, Loveland: Presburger Arithmetic with Bounded Quantifier Alternation. (Gives a slight improvement of the original Cooper's algorithm.)
- Section 7.2 of the Calculus of Computation Textbook