Seminar Report

The Decision Problem: Quantifier Elimination

Martin Pfeifhofer
martin.pfeifhofer@uibk.ac.at

28 February 2017

Supervisor: Martin Avanzini, PhD

Abstract

The 'Decision Problem' is an important topic in mathematics and computer science. This paper is concerned with the Decision Problem and one of the many methods to solve it, which is named 'Quantifier Elimination'. At the beginning I give a short overview of the decision problem and what is necessary to know for its application. Then I will unfold the connection to quantifier elimination and give a concrete example how to solve such a problem. Further I will treat a specific theory to be solved by quantifier elimination and explain an algorithmic solution given by David C. Cooper in 1972.

Contents

1 Introduction

This is a report about Quantifier Elimination, which is a procedure used for solving the Decision Problem in certain cases. The main source for this report was the 5th Chapter 'Decidable Problems' of the Handbook of Practical Logic and Automated Reasoning [1] by John Harrison. Before we can dive into the topic of Quantifier Elimination, we first have to clarify why there is a need for such a computation and where we can apply it. The first section is concerned with the Decision Problem, the term of theories and their connection to the method of quantifier elimination for solving such a problem. In the second section I'm explaining a certain theory, which is called Presburger Arithmetic and an algorithm by David C. Cooper, that shows quantifier elimination for that theory.

2 Preliminaries

2.1 The Decision Problem

The Decision Problem [2] - or 'Entscheidungsproblem' - is basically a *yes* or *no* question on a finite input. Given two numbers x and y; does x evenly divide y? - would be a simple example. To solve such a problem we need to make a decision by evaluating wether the statement is true or not. In a first-order logic we could describe the Decision Problem as the following collection of three natural and closely connected problems:

1. Confirm that a logically valid formula is indeed valid; never confirm an invalid one.
2. Confirm that a logically invalid formula is indeed invalid; never confirm a valid one.
3. Test wether a formula is valid or invalid.

Why is this important? The Decision Problem is typically used for the question of decidability, which is to determine if a certain formula - in this case a first order logic formula - is valid or not. Being able to decide about such a formula is the motivation for the search of algorithmic solutions for the Decision Problem. There is no systematic procedure of doing so generally, but there are procedures which work for certain special cases of formulas and for validity in certain special classes of models. Some of them are going to be covered in this paper.

2.2 Theories

In search for a decidable procedure we turn our interest to the following questions, considering logic with equality:

- instead of validity in all interpretations, we consider validity in a particular class of interpretations M, i.e. wether $\models_M p$ in a class K;
- logical consequence from a set of axioms Σ, i.e. wether $\Sigma \models p$.

Anyway both of these two formulations are inconsequential, because the class K is defined to be exactly the collection of models of a set of axioms Σ:

$$\mathrm{Mod}(\Sigma) = \{M \mid \text{for all } \psi \in \Sigma, \models_M \psi\}.$$

Now by defining a theory of a class of interpretations K, which is the set of all sentences holding in all interpretations in the class K, we also define a kind of controverse to Mod.

Definition 2.1. *A theory is a set of formulas T closed under logical consequence, i.e. such that for any formula p we have $T \models p$ iff $p \in T$.*

When we are talking about the theory of a specific structure, the following terminology will be used:

$$\mathrm{Th}(K) = \{\psi \mid \text{for all } I \in \mathrm{K}, \models_I \psi\}.$$

To For example Th($\mathbb{R}$),with a slight abuse of notation, is called the theory of the real numbers and is defined to be exactly the set of first-order sentences that hold in the specific structure $\mathbb{R}$. To be more precise about such a theory we can bundle the list of functions and predicates, which are used e.g. Th($\mathbb{R}$,0,1,$-, +, <$). This would then be a purely additive theory of real numbers with $<$ as the only predicate besides equality. Theories are axiomatizable, meaning that there is a set of axioms Σ that describes the theory. If the set of axioms is finite, we call the theory *finitely axiomatizable*. Other important characteristics of theories are:

- Consistency − we never have both $T \models p$ and $T \models \neg p$.
- Completeness − for any sentence p, either $T \models p$ or $T \models \neg p$.
- Decidability − there is an algorithm that takes as input a formula p and decides whether $T \models p$.

Out of these properties we know that a complete finitely axiomatizable theory is automatically decidable. Why is that so? By completeness we know that for any sentence p there has to be a verification A where either $A \Rightarrow p$ or $A \Rightarrow \neg p$ terminates.

Definition 2.2. A theory is complete iff all its models are elementarily equivalent.

Two models of the same signature σ are elementarily equivalent if they satisfy the same first order -sentences. A signature contains function symbols and relation symbols or predicates.
What are theories now actually good for? Well, because there does not exist a general systematic procedure to determine if a formula is decidable, we have to find some procedure which is working on a certain theory.

2.3 Quantifier Elimination

Definition 2.3. *A theory T in a first-order language L admits quantifier elimination if for each formula p of L, there is a quantifier-free formula q with $FV(q) \subseteq FV(p)$ such that p and q are T-equivalent ($T \models p \Leftrightarrow q$).*

As stated in the above definition, if a theory admits quantifier elimination, then for every formula p of a language L there is an equivalent formula which is quantifier free. Why is this important at all? If a theory admits quantifier elimination, we can reduce many logical questions that seem difficult to the special case of quantifier-free formulas, where they can be much easier. The reason of that is, because we can simply evaluate it. The goal is now to create an algorithmic process for constructing a quantifier-free equivalent of formulas in first order logic.

In the case of arithmetical theories, quantifier elimination is a generalization of testing the solvability of equations, that are in the form of $\exists x\ .F[x] = 0$. The quantifier free T-equivalent of such a formula contains no variables at all. Many theories of practical interest have the same truth-values in all models and can be evaluated to true or false algorithmically. Any such theory that admits quantifier elimination is therefore *complete* and *decidable*, which are attributes we are particularly interested in. An effective decision procedure is to reduce a formula to its quantifier-free equivalent and evaluate it then. This suffices to demonstrate for formulas with the following form

$$\exists x.\ \alpha_1 \wedge \cdots \wedge \alpha_n$$

where each α_i is a literal (a negated or non-negated atomic formula) containing x. The reason why we can ignore universal quantifiers is due to the fact that we can transform every $\forall x.P[x]$ into a $\neg(\exists x.\neg P[x])$. This can be applied from the innermost quantifier to the outermost, putting the body into disjunctive normal form and distributing the existential quantifier over it. A short example is shown in the following subsection.

2.4 Dense Linear Orders

In 1927 Cooper Harold Langford showed that the theory of *dense linear orders without end points* (which I will just call 'DLOs') admits quantifier elimination. This section is concerned with the explicit algorithm for this theory by John Harrison, thus showing quantifier elimination on DLOs. The language contains the binary predicate '$<$' as well as equality, but no function symbols. It can be axiomatized as follows:

$$\begin{aligned}
&\forall x\ y.x = y \vee x < y \vee y < x,\\
&\forall x\ y\ z.x < y \wedge y < z \Rightarrow x < z,\\
&\forall x.\neg(x < x),\\
&\forall x\ y.x < y \Rightarrow \exists z.x < z \wedge z < y,\\
&\forall x.\ \exists y.x < y,\\
&\forall x.\ \exists y.y < x.
\end{aligned}$$

The first three axioms show that DLOs are an irreflexive, transitive total linear order. The 4th axiom asserts denseness and the last two show that there is no greatest or smallest element. Denseness means that between each pair of elements there is another element. A set of numbers that would not satisfy this axiom is $\mathbb{Z}$. Therefore $\mathbb{Z}$ is not a model of the DLO axioms. Since $\top$ and $\bot$ are the only ground formulas in the language, this theory is *complete* and *decidable*. By Definition 2.2 we can also see that all models of the DLO axioms are elementarily equivalent. This also means that no sentence in the first-order language considered here can distinguish two models of the theory, such as $\mathbb{R}$ and $\mathbb{Q}$.

2.4.1 Example

Until now we were concerned with the theoretical background of quantifier elimination. So here we will work out a small example for a better understanding of the actual procedure. Consider the following formula which is in $\mathbb{R}$ and in the DLOs:

$$\forall x.\exists y.\exists z.\ (y + 1 \leq x \wedge z + 1 \leq y \wedge 2x + 1 \leq z)$$

We are starting with the innermost quantifier, which is $\exists z$. We are eliminating this quantifier by removing every occurrence of z, while we are not changing the actual meaning of the formula. This is done first by changing $z + 1 \leq y \wedge 2x + 1 \leq z$ to $z \leq$ y $- 1 \wedge 2x + 1 \leq z$. Thus we can just substitute the rearranged terms with z.

$$\forall x.\exists \text{y.}\ (y + 1 \leq x \wedge 2x + 1 \leq y - 1)$$

We are continuing to eliminate the innermost quantifier, which now is $\exists y$. We are seperating the -1 and $+1$ from y by using addition and subtraction, which does not change the meaning of the formula at all.

$$\forall x.\ (2x + 2 \leq \text{x} - 1)$$

Now we continue by eliminating the last quantifier that is left, which is $\forall x$. This time we have an universal quantifier, which means we have to transform it into an existential quantifier. This can be done using the following equivalence $\forall x.P[x] \Leftrightarrow \neg(\exists x.\neg P[x])$.

$$\neg\exists x.\ \neg(2x + 2 \leq x - 1)$$

Before we continue with the quantifier elimination, we can simplify our formula to

$$\neg\exists x.\ \neg(x \leq -3)$$

So lets finally remove our last quantifier, which we transformed earlier into $\neg\exists x$. By doing that everything is left between the parentheses is a simple expression, that can be evaluated to a true, considering that we are in $\mathbb{R}$.

$$\neg \mathit{TRUE}$$

Then we are applying the negation and are getting our evaluated quantifier-free equivalent.

$$\mathit{FALSE}$$

3 Presburger Arithmetic

The next theory that is covered is named Presburger Arithmetic, after Mojzesz Presburger, who first showed quantifier elimination and decidability for it in 1930. Presburgers algorithm has additional historical significance, since the implementation by Davis (1957) was arguably the first logical decision procedure actually to be implemented on a computer.

The theory consists of linear integer arithmetic, which is the set of formulas true in $\mathbb{Z}$ without multiplication. In its most obvious formulation this theory does not admit quantifier elimination. This is why we have to add so called 'divisibility predicates' D_k for all integers $k \geq 2$. Because of the fact that ground instances of divisibility predicates are always decidable, quantifier eliminations also holds for this modified theory. The conventional notation for 'd divides x' is $d|x$. Thus we are fixing the following first order language containing infinitely many predicate symbols:

- constants 0 and 1
- functions of unary negation ($\neg$), addition ($+$) and subtraction ($-$)
- equality ($=$), inequalities ($\leq$, $<$, $\geq$, $>$) and unary predicates D_k ('is divisible by k') for all integers $k \geq 2$.

In contrast to DLOs, we will not spell out a set of axioms for the theory, but work directly with properties that hold in the usual model $\mathbb{Z}$. In the implementation of Harrison the language is a bit adjusted to express things like $x + x$, which could not expressed in the 'pure' version of the language.

- We allow positive and negative integer constants:
  We can rewrite -4 as $-(1 + 1 + 1 + 1)$.
- We allow multiplication between constants:
  We can rewrite $4 \cdot x$ as $x + x + x + x$.
- We use a divisibility predicate 'divides' where the LHS argument d has to be a positive integer constant: $d\,|x$.

3.1 Canonical Forms

Writing terms in canonical form makes the implementation of multiplication of numeral constants easier:

$$c_1 \cdot x_1 + \cdots + c_n \cdot x_n + k,$$

where $n \geq 0$, c_i and k are integer constants, and the x_i are distinct variables, with a fixed order. Here we insist that c_i are present even if they are 1, but that they are never 0, and that k is present even if it is zero. To work with terms in canonical form we first need operations. Our two basic operations are: multiplication by an integer constant

and addition. For multiplication we are multiplying up all the coefficients. Here, an expresson is canonical if it is of the form

$$n \cdot (c_1 \cdot x_1 + \cdots + c_n \cdot x_n + k) = (n \cdot c_1\ ) \cdot x_1 + \cdots + (n \cdot c_n\ ) \cdot x_n + (n \cdot k)$$

unless $n = 0$, then we can just return 0. For Addition we need to merge together our given sequences of variables, while maintaining the fixed order. This is done by pairwise summing up all the coefficients. Now after we have defined these basic functions we can use them to define negation and subtraction for canonical terms. We can also implement multiplication, with at least one of the two operands being a constant.
To convert any permissible term into canonical form, we need a linearization of terms of the form x into $1 \cdot x + 0$ and after that we need to expand this linearization to atomic formulas. So we force both equations and inequalities to have zero on the LHS, e.g. transforming $s = t$ to $0 = s - t$ and $s < t$ to $0 < t - s$. Then we take advantage of the fact, that the integers are a 'discrete' set of numbers, and are rewriting every atomic inequality formulas in terms of $<$, e.g. $s \leq t$ as $0 < (\ t + 1) - s$. And finally we also force the left-hand constants in divisibility predicates to be positive.

3.2 Coopers Algorithm

Even due to the fact that Presburgers original algorithm is pretty straightforward and follows the classic quantifier elimination pattern of dealing with the special case of an existentially quantified conjunction of literals, we are now dealing with a slightly optimized version. This algorithm was published by David C. Cooper in 1972 and has a significant difference: it allows us to eliminate an existential quantifier whose body is an arbitrary quantifier-free negation normal form formula. This avoids a possible blowup caused by the transformation into DNF. The following subsection is concerned with the implementation of Coopers algorithm by John Harrison.

3.2.1 Algorithmic Application

Lets consider the task of eliminating the existential quantifier from $\exists x.p$ where p is quantifier-free. We will assume that all the atoms have been maintained in the standard form with 0 on the left and a linearized term on the right, and only strict inequalities using '$<$' present. Further we assume that p is in negation normal form, which means that a quantifier free formula can contain conjunctions and disjunctions of literals which are in one of the following forms: $0 = t$, $\neg(0 = t)$, $0 < t$, $d \mid t$ or $\neg(d \mid t)$ and if a term t contains x, it is in the form $c \cdot x + s$. To correlate all instances of x multiplied by different coefficients we compute the positive least common multiple (LCM) l of all the coefficients, returning 1 if there doesn't occur any x. After that we can make any coefficient of x equal to $\pm l$, simply by taking each atomic formula whose right-hand argument is of the form $c \cdot x + z$, and consistently multiplying it through by an appropriate m. For inequalities $m = |l\ /\ c|$, because we cannot multiply by negative numbers without changing their sense, and for anything else by $m = l\ /\ c$.

So we transformed all the coefficients of x from $\pm l \cdot x$ to $\pm 1 \cdot x$. As next step we can just replace $l \cdot x$ with just x and add a new divisibility clause, which is justified by the following equivalence:

$$\exists x.P[l \cdot x]) \Leftrightarrow (\exists x.l \mid x \wedge P[x]).$$

After the initial transformations of the formula, we can now start with the main quantifier elimination step for $\exists x.\ P[x]$. Due to the fact that integers are a discrete set of numbers and that any set of integers has a minimal element we know that $\exists x.\ P[x]$ holds, iff

- either there are arbitrarily large and negative x such that $P[x]$ hold,
- or there is a minimal x such that $P[x]$ hold.

We will now separately consider how to find quantifier equivalents for the above two mentioned cases which are expressed at the right of this equivalence:

$$(\exists x.\ P[x]) \Leftrightarrow (\forall y.\ \exists x.\ x < y \wedge P[x]) \vee (\exists x.\ P[x] \wedge \forall y.\ y < x \Rightarrow \neg P[y]).$$

Arbitrarily large and negative x

The first case we are looking at is when there are arbitrarily large and negative x such that $P[x]$. Here we claim that $P[x]$ must be equivalent to $P_{-\infty}[x]$ and so we can replace atoms in formulas with the following form with $\top$ or $\bot$.

In $P[x]$	In $P_{-\infty}[x]$
$0 = x + a$	$\bot$
$0 < x + \mathrm{a}$	$\bot$
$0 < -x + \mathrm{a}$	$\top$

Lemma 3.1. *For sufficiently large and negative x, $P[x]$ and $P_{-\infty}[x]$ are equivalent, i.e. $\exists y.\ \forall x.\ x < y \Rightarrow (P[\mathrm{x}] \Leftrightarrow P_{-\infty}[x])$ holds.*

Now lets take a look at the divisibility predicates. All divisibility terms $d \mid \pm x + a$ are unchanged if x is altered by an integer multiple of d. So we are looking for the positive least common multiple D of all ds occurring in formulas of the form $d \mid c \cdot x + a$. At this stage we actually know that $\mathrm{c} = \pm 1$. After this step all divisibility atoms in the formula are invariant if x is changed to $x \pm kD$. The fact that in the case of $P_{-\infty}[x]$ divisibility atoms and other atoms not involving x are all that's left, implies that $P_{-\infty}[x \pm kD] \Leftrightarrow P_{-\infty}[x]$ always holds. That means we can simplify the equivalence of our target formula. For any $P[\mathrm{x}]$ which is quantifier free and in NNF we have

$$(\forall y.\ \exists x.\ x < y \wedge P[x]) \Leftrightarrow \bigvee_{i=1}^{D} P_{-\infty}[i].$$

Minimal x

Lets now look at the case of a minimal x satisfying $P[x]$. Here $P[x]$ holds, but $P[x - D]$ does not. As we know divisibility predicates do not change under translation by D, so the change from true to false must have come from another literal, which has changed from true to false in the step from x to a smaller value. For such a literal we can define a 'boundary point' b, such that it is false for $x = b$, but true for $x = b + 1$. In the following table there are all literals which change from true to false as x decreases by D.

Literal	Boundary Point
$0 = x + a$	$-(a + 1)$
$\neg(0 = x + a)$	$-a$
$0 < x + a$	$-a$
$0 < -x + a$	none
$\mathrm{d} \mid x + a$	none
$\neg(\mathrm{d} \mid x + a)$	none
literals without x	none

The collection of boundary points for literals is called a B-set of the considered formula.

Lemma 3.2. *If D is the LCM of all relevant divisors in a quantifier-free negation normal form formula $P[x]$ with no logically negated inequality literals and a B-set B, and $P[x]$ holds while $P[x - D]$ does not, then $x = b + j$ for some $b \in B$ and $1 \leq j \leq D$.*

Now after we know how to find quantifier-free equivalents of both cases we are now coming to the main definition justifying quantifier elimination, using our simplified equivalences.

Lemma 3.3. *If $P[x]$ is a formula in the subset being discussed with B-set B, and D is the positive lowest common multiple of all the relevant divisors, then the following equivalence holds:*

$$(\exists x.\ P[x]) \Leftrightarrow \bigvee_{j=1}^{D} (P_{-\infty}[j] \vee \bigvee_{b \in B} P[b + j]).$$

To be able to perform this, we first need to define a formula that allows us to substitute instances like $P[b + j]$ while retaining canonical form. The function replaces the top variable x in atoms by another term t, which is assumed not to involve x, and is therefore restoring canonicality. Now for the overall inner quantifier elimination step we can just apply the transformation as stated in Lemma 3.3. After the elimination of all quantifiers from an initially closed formula, the result contains no variables at all and we can now evaluate all atoms to either true or false.

There are some optimizations that can be done to increase the efficiency of this algorithm. One was already mentioned in Coopers own paper, which is to sometimes use dual expansion based on a 'plus infinity' variant of the formula and corresponding A-Sets instead of B-Sets. Another optimization was given by Reddy and Loveland in 1978, which slightly improved the treatment of the coefficient homogenization of Coopers algorithm.

4 Conclusion

First mentioned in 1928, The Decision Problem today is still an important topic in the field of mathematics and computer science. Even if there are some techniques to solve such a problem by considering a specific input, scientists are still searching for more efficient methods of computation. Quantifier Elimination is considered a helpful method for solving decidable problems. The theory of rationals and the theory of integers extended with divisibility predicate both admit quantifier elimination. Unfortunately many other theories do not, like the theory of equality. Even if Coopers algorithm is a nice way of solving this decidable problem and even due to the fact Presburger Arithmetic is complete and decidable, the worst case complexity of the algorithm is is known to be at least doubly exponential in the size of the formula (Fischer and Rabin 1974). So we see, that there can still a lot be done in the future and so we are looking forward to new scientific discoveries.

References

[1] John Harris. Handbook of Practical Logic and Automated Reasoning, Mar 2009.

[2] Wikipedia. Definition of the Decision Problem, Feb 2017.