Decision Procedures
An Algorithmic Point of View

Deciding Combined Theories

So far we know how to…

- Decide Equality Logic with Uninterpreted Functions:
  \((x_1 = x_2) \lor \neg(f(x_2) = x_3) \land \ldots\)

- Decide Disjunctive Linear arithmetic:
  \(3x_1 + 5x_2 \leq 2x_3 \land x_2 \leq 4x_4 \ldots\)

- What about a combined formula?
  \((x_2 \geq x_1) \land (x_1 - x_2 \geq x_2) \land (x_2 \geq 0) \land f(f(x_1) - f(x_2)) \neq f(x_3)\)

More combination examples:

- Combining lists, arithmetic and Uninterpreted Functions:
  \((x_1 \leq x_2) \land (x_2 \leq (x_1 + \text{car(cons}(0, x_1)))) \land p(h(x_1) - h(x_2)) \land \neg p(0)\)

- Combining Arrays and Arithmetic:
  \(x = \text{store}(v, i, e)[j] \land y = v[j] \land x > e \land x > y\)

Combining theories

- Approach #1: Reduce all theories to a common logic, if possible (e.g. Propositional Logic).
  - All un-quantified theories we saw so far are in NP.
  - We saw their direct translation to SAT (i.e. not through a Turing-machine).

- Approach #2: Combine decision procedures of the individual theories.
  - How? we will learn the Nelson-Oppen method*

- * Greg Nelson and Derek Oppen, *simplification by cooperating decision procedures*, 1979
Reminders: theories and signatures

- First order logic –
  - Variables
  - Logical symbols (boolean connectives, quantifiers).
  - Syntax (wffs)
  - “Logical” axioms and inference rules
- First order theories –
  - Non-logical symbols (theory-specific function and predicate symbols)
  - Additional axioms characterizing the theory
  - The signature $\Sigma$ of a theory $T$ holds the set of functions and predicates of the theory.
- “First order quantifier-free theories with equality” – the equality predicate must be part of the signature.

The Theory-Combination problem

- Given theories $T_1$ and $T_2$ with signatures $\Sigma_1$ and $\Sigma_2$, the combined theory $T_1 \oplus T_2$
  - has signature $\Sigma_1 \cup \Sigma_2$ and
  - the union of their axioms.
- Let $\phi$ be a $\Sigma_1 \cup \Sigma_2$ formula.
- The problem: Does $T_1 \oplus T_2 \vdash \phi$?

The problem

- The Theory-Combination problem is undecidable (even when the individual theories are decidable).
- Under certain restrictions, it becomes decidable.
- We will assume the following restrictions:
  - $T_1$ and $T_2$ are decidable, quantifier-free first-order theories with equality.
  - Disjoint signatures (other than equality): $\Sigma_1 \cap \Sigma_2 = \emptyset$
  - More restrictions to follow…
- There are extensions to the basic algorithm that partially overcomes each of these restrictions.

The Nelson-Oppen method (1)

- Purification: validity-preserving transformation of the formula after which predicates from different theories are not mixed.
  1. Replace an ‘alien’ sub-expression $\phi$ with a new auxiliary variable $a$
  2. Constrain the formula with $a = \phi$
     Transform $x_1 \geq f(x_1)$
     … into $x_1 \geq a_1 \land a_1 = f(x_1)$

Uninterpreted Functions

Pure expressions, shared variables
The Nelson-Oppen method (2)

- After purification we are left with several sets of pure expressions $F_1, \ldots, F_n$ such that:
  - $F_i$ belongs to some ‘pure’ theory, which we can decide.
  - Shared variables are allowed, i.e. it is possible that $\text{vars}(F_i) \cap \text{vars}(F_j) \neq \emptyset$, for some $i \neq j$.
  - $\phi$ is satisfiable iff $F_1 \land \ldots \land F_n$ is satisfiable.

Example (1)

$$(x_1 \leq x_2) \land (x_2 \leq (x_1 + \text{car}(\text{cons}(0, x_1)))) \land p(h(x_1) - h(x_2)) \land \lnot p(0)$$

- Purification:
  $$(x_1 \leq x_2) \land (x_2 \leq x_1 + a_1) \land p(a_2) \land \lnot p(a_3) \land\]
  $$a_1 = \text{car}(\text{cons}(a_3, x_1)) \land\]
  $$a_2 = a_3 - a_4 \land\]
  $$a_3 = h(x_1) \land\]
  $$a_4 = h(x_2) \land\]
  $$a_5 = 0$$

Example (1), cont’d

<table>
<thead>
<tr>
<th>L: Arithmetic</th>
<th>EUF</th>
<th>Lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1 \leq x_2$</td>
<td>$a_3 = h(x_1)$</td>
<td>$a_1 = \text{car}(\text{cons}(a_3, x_1))$</td>
</tr>
<tr>
<td>$x_2 \leq x_1 + a_1$</td>
<td>$a_3 = h(x_2)$</td>
<td>$a_1 = a_3$</td>
</tr>
<tr>
<td>$a_2 = a_3 - a_4$</td>
<td>$p(a_2)$</td>
<td>$a_1 = a_3$</td>
</tr>
<tr>
<td>$a_3 = 0$</td>
<td>$\lnot p(a_3)$</td>
<td>$x_1 = x_2$</td>
</tr>
<tr>
<td>$a_1 = a_5$</td>
<td>$a_1 = a_5$</td>
<td>$x_1 = x_2$</td>
</tr>
<tr>
<td>$a_1 = a_5$</td>
<td>$a_3 = a_4$</td>
<td>$a_3 = a_4$</td>
</tr>
<tr>
<td>$a_3 = a_4$</td>
<td>$a_3 = a_4$</td>
<td>$a_2 = a_5$</td>
</tr>
<tr>
<td>$a_1 = a_5$</td>
<td>$\lnot p(a_3)$</td>
<td>False</td>
</tr>
</tbody>
</table>

Equality Propagation

* So far only for ‘non-convex’ theories – to be explained.
Example (2)

\[(x_2 \geq x_1) \land (x_1 - x_2 \geq x_2) \land (x_3 \geq 0) \land f(x_1) \neq f(x_2) \land a_1 = a_2 - a_3 \land a_2 = f(x_1) \land a_3 = f(x_2)\]

Purification:

\[(x_2 \geq x_1) \land (x_1 - x_3 \geq x_2) \land (x_3 \geq 0) \land f(a_1) \neq f(x_3) \land a_1 = a_2 - a_3 \land a_2 = f(x_1) \land a_3 = f(x_2)\]

Wait, it’s not so simple…

- Consider: \[1 \leq x \land x \leq 2 \land p(x) \land \neg p(1) \land \neg p(2)\], where \[x \in \mathbb{Z}\]

<table>
<thead>
<tr>
<th>Arithmetic over (\mathbb{Z})</th>
<th>Uninterpreted predicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1 \leq x]</td>
<td>(p(x))</td>
</tr>
<tr>
<td>[x \leq 2]</td>
<td>(\neg p(1))</td>
</tr>
<tr>
<td></td>
<td>(\neg p(2))</td>
</tr>
</tbody>
</table>

- Neither theory implies an equality, and both are satisfiable.
- But the conjunction is unsatisfiable!

Example (2) – cont’d

<table>
<thead>
<tr>
<th>L. Arithmetic</th>
<th>EUF</th>
</tr>
</thead>
<tbody>
<tr>
<td>[x_2 \geq x_1]</td>
<td>(f(a_1) \neq f(x_2))</td>
</tr>
<tr>
<td>[x_1 - x_3 \geq x_2]</td>
<td>(a_2 = f(x_1))</td>
</tr>
<tr>
<td>[x_3 \geq 0]</td>
<td>(a_3 = f(x_2))</td>
</tr>
<tr>
<td>[a_1 = a_2 - a_3]</td>
<td>(x_3 = 0)</td>
</tr>
<tr>
<td>[a_2 = f(x_1)]</td>
<td>(x_1 = x_2)</td>
</tr>
<tr>
<td>[a_3 = f(x_2)]</td>
<td>(a_1 = 0)</td>
</tr>
<tr>
<td>[x_3 = 0]</td>
<td>False</td>
</tr>
</tbody>
</table>

Some theories have it, some don’t

- Definition: A theory \(T\) is convex if for all conjunction of literals, \(\phi\), it holds that:
  - If \(\phi \Rightarrow \bigvee_{i=1..n} x_i = y_i\), where \(n > 1\), then
    - \(\phi \Rightarrow x_j = y_j\), for some \(j \in \{1..n\}\)
  - where \(x_j, y_j, x_i, y_i \in \text{vars}(\phi)\).

  - Convex: Linear Arithmetic over \(\mathbb{R}\), EUF
  - Non-convex: Almost anything else…
Convexity: examples

- Linear arithmetic over $\mathbb{R}$ is convex
  $\phi$: $x_1 \leq 1 \land x_1 \geq 0$ implies an infinite disjunction?
  $\phi$: $x_1 \leq 1 \land x_1 \geq 1$ implies a singleton
  $\phi$: $x_1 \leq 1 \land x_1 \geq 2$ implies everything

- Linear arithmetic over $\mathbb{Z}$ is not convex
  $\phi$: $1 \geq x_1 \land x_1 \geq 2$
  Although $\phi \Rightarrow (x_1 = 1 \lor x_1 = 2)$
  It is not the case that $\phi \Rightarrow x_1 = 1 \lor \phi \Rightarrow x_1 = 2$

Decision Procedures
An algorithmic point of view

So why is convexity important?

- Recall: $1 \leq x \land x \leq 2 \land p(x) \land \neg p(1) \land \neg p(2)$, where $x \in \mathbb{Z}$

| $1 \leq x$ | $p(x)$ |
| $x \leq 2$ | $\neg p(1)$ |
| $\neg p(2)$ |

<table>
<thead>
<tr>
<th>Arithmetic over $\mathbb{Z}$</th>
<th>Uninterpreted predicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \leq x$</td>
<td>$p(x)$</td>
</tr>
<tr>
<td>$x \leq 2$</td>
<td>$\neg p(1)$</td>
</tr>
<tr>
<td>$\neg p(2)$</td>
<td></td>
</tr>
</tbody>
</table>

Neither theory implies an equality, and both pure subformulas are satisfiable.

Decision Procedures
An algorithmic point of view

So why is convexity important? (cont’d)

- But: $1 \leq x \land x \leq 2$ imply the disjunction $1 = x \lor x = 2$
- Since the theory is non-convex we cannot propagate either $x = 1$ or $x = 2$.
- We can only propagate the disjunction itself.

| $1 \leq x$ | $p(x)$ |
| $x \leq 2$ | $\neg p(1)$ |
| $\neg p(2)$ |

<table>
<thead>
<tr>
<th>Arithmetic over $\mathbb{Z}$</th>
<th>Uninterpreted predicates</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \leq x$</td>
<td>$p(x)$</td>
</tr>
<tr>
<td>$x \leq 2$</td>
<td>$\neg p(1)$</td>
</tr>
<tr>
<td>$\neg p(2)$</td>
<td></td>
</tr>
</tbody>
</table>

$\neg p(1)$ $\lor \neg p(2)$

Split!

| $x = 1$ | $x = 2$ |
| False | False |

Decision Procedures
An algorithmic point of view
So why is convexity important? (cont’d)

- Conclusion: when the theory is non-convex, we must case-split.
- This adds a splitting step in Nelson-Oppen.
- As a result:
  - Convex theories: Polynomial
  - Non-Convex theories: Exponential

The (full) Nelson-Oppen method

1. Purify \( \phi \) into \( \phi' : F_1 \wedge \ldots \wedge F_n \).
2. If \( \exists i \) such that \( F_i \) is unsatisfiable, return 'unsatisfiable'.
3. If \( \exists i, j \) such that \( F_i \) implies an equality not implied by \( F_j \), add it to \( F_j \) and go to step 2.
4. If \( \exists i \) such that \( F_i \Rightarrow (x_1 = y_1 \lor \ldots \lor x_k = y_k) \) but \( F_i \wedge x_j \neq y_j \) is satisfiable, for \( j = 1, \ldots, k \), apply method recursively to \( \phi' \wedge x_1 = y_1, \ldots, \phi' \wedge x_k = y_k \). If any of them is satisfiable, return 'satisfiable'. Otherwise return 'unsatisfiable'.
5. Return 'satisfiable'.

Correctness is hard to prove…

- Theorem: N.O. returns unsatisfiable if and only if its input formula \( \phi \) is unsatisfiable.
- We will prove this theorem for the case of combining two convex theories. The generalization is not hard. The proof is based on [NO79].

Correctness is hard to prove…

- (\( \Rightarrow \)) N.O. returns 'unsatisfiable' implies \( \phi \) is unsatisfiable. (The simple direction)
  - Assume \( \phi \) is satisfiable and let \( \alpha \) be a satisfying assignment of \( \phi \).
  - Let \( A = \{a_1, \ldots, a_n\} \) be the purification (auxiliary) variables.
  - Claim: there exists an assignment to the \( A \) variables such that \( \alpha \) extended with this assignment satisfies \( F_1 \wedge F_2 \).
  - Let \( \alpha' \) be this extended assignment.
  - For each equality \( eq \) added in line 3, \( \exists i, F_i \Rightarrow eq \).
  - Since \( \alpha' \vDash F_1 \) then also \( \alpha' \vDash eq \).
  - Hence for all \( j \in \{1,2\}, \alpha' \vDash F_j \wedge eq \).
  - Thus, N.O. does not return unsat in this case.
  - In other words, if N.O. returns unsat, then \( \phi \) is unsat.
Stopped here – to finish this out, consult the nelson, oppen paper. (in the Readings directory)