Notes on Satisfiability-Based Problem Solving

Solving by Grounding

David Mitchell
mitchell@cs.sfu.ca
February 8, 2020

This is a preliminary draft. Please do not distribute. Corrections and suggestions welcome.

We review our formalized problem solving context, and describe a scheme for problem solving in this context based on "grounding" or instantiation. Grounding-based schemes work as follows. A pair consisting of a problem specification, which contains quantified variables, and an instance, are mapped on to a quantifier-free representation, which is then solved by a "ground solver". Grounding-based schemes are the basis of most current, competitive systems for solving search, decision and optimization problems from declarative specifications.

For simplicity and clarity, we will focus on a narrowly defined setting in which we consider only problems in the complexity class NP (or search problems whose decision variants are in NP) , specified in ∃SO with no second order function variables, and solved by grounding (a kind of automated reduction) to SAT and then application of a SAT solver. However, the scheme generalizes to many other settings, including higher complexity classes, richer specification languages, and ground solvers for languages other than SAT.

1 Problem Solving Context

The specification-based problem solving task we consider, which is abstracted from real problem solving systems and formalized in logic, is as follows. We have:

1. A class $\mathcal{K}$ of finite structures for some vocabulary $\mathcal{L}_\mathcal{I}$. The $\mathcal{I}$ in $\mathcal{L}_\mathcal{I}$ is to indicate that this is the vocabulary used to describe our problem instances, which are finite $\mathcal{L}_\mathcal{I}$-structures. For decision problems, the task is to decide if a given $\mathcal{L}_\mathcal{I}$-structure is in $\mathcal{K}$ or not. For search problems, instances with solutions are in $\mathcal{K}$, and those which are not in $\mathcal{K}$ do not have solutions.
2. An $\mathcal{L}_\mathcal{I}$-formula $\phi_\mathcal{K}$ of ∃SO, that defines the class $\mathcal{K}$ of structures (i.e., $Mod[\phi_\mathcal{K}] =$

$\mathcal{K}$). $\phi_{\mathcal{K}}$ is of the form

$$\exists \overline{R} \phi$$

where $\overline{R}$ is a tuple of second order variables and ϕ is a FO formula with vocabulary $\mathcal{L} = \mathcal{L}_{\mathcal{I}} \cup \overline{R}$ (the instance vocabulary plus the symbols in $\overline{R}$). Intuitively, we think of the symbols in $\overline{R}$ as being the vocabulary in which we describe solutions (or witnesses) for instances. Thus, $Mod[\phi]$ contains exactly those finite $\mathcal{L}$-structures that consist of a problem instance together with one of its solutions (or witnesses).

3. Finding a solution to problem instance $\mathcal{A}$ amounts to finding structure $\mathcal{A}'$ that is an expansion of $\mathcal{A}$ to vocabulary $\mathcal{L}$ such that $\mathcal{A}' \models \phi$. (For decision problems, we need only decide if such an expansion exists, but generally speaking we do not know how to do that without constructing the expansion. Cases to the contrary tend to be interesting special cases.)

Remark 1. Intuitively, we think of the symbols in $\overline{R}$ as being the vocabulary in which we describe the solutions. However, strictly speaking this may not be precise. In practice, $\overline{R}$ describes a witness or certificate for a given instance, but may not correspond exactly to our intuitive notion of a solution for our problem, and may contain more symbols than needed to describe these solutions, because using them is convenient for the specification writer or the solver.

2 Grounding

Given a FO sentence ϕ, with quantifiers, and a finite set S, the goal of grounding is, roughly speaking, to produce a ground formula ϕ_S that is equivalent to ϕ on structures with universe S. We may obtain such a formula by repeatedly re-writing a sub-formula involving quantification over variable x as a conjunction or disjunction of $|S|$ formulas, with occurrences of x replaced by constant symbols denoting elements of S. For this, we need to have a unique constant symbol for each element of S.

For any set S, let $\tilde{S}$ denote a set of $|S|$ constant symbols, one for each element of S. We will write $\tilde{a}$ for the constant symbol corresponding to element $a \in S$. For any $\mathcal{L}$-structure $\mathcal{A}$, we write $\tilde{A}$ for a $\mathcal{L} \cup \tilde{S}$-structure that expands $\mathcal{A}$ with interpretations of the constant symbols of $\tilde{A}$. (That is, for each constant symbol $\tilde{a} \in \tilde{A}$, the interpretation of $\tilde{a}$ in structure $\tilde{\mathcal{A}}$ is the universe element $a \in A$.)

Definition 1. Let ϕ be a FO sentence of vocabulary $\mathcal{L}$ and S a finite set. Formula ϕ_S is a grounding of ϕ over S if:

1. ϕ_S is a ground formula over vocabulary $\mathcal{L} \cup \tilde{S}$;
2. For any $\mathcal{L}$-structure $\mathcal{A}$ with universe $A = S$, it holds that $\mathcal{A} \models \phi$ iff $\tilde{\mathcal{A}} \models \phi_S$;

To obtain a grounding, we substitute constant symbols for variables in a formula. We write $\psi\langle x \to c\rangle$ for the formula obtained by replacing each free occurrence of variable x in formula ψ with the constant symbol c.

A particular grounding of ϕ over S can be constructed by recursion on the structure of ϕ, mirroring the recursion in the definition of the satisfaction relation, and in the basic algorithm for FO model checking. The following function $Gnd(\phi, S)$ performs this construction.

$$Gnd(\phi, S) = \begin{cases} \phi & \text{if } \phi \text{ is an atom} \\ (Gnd(\psi_1, S) \vee Gnd(\psi_2, S)) & \text{if } \phi \text{ is } (\psi_1 \vee \psi_2) \\ (Gnd(\psi_1, S) \wedge Gnd(\psi_2, S)) & \text{if } \phi \text{ is } (\psi_1 \wedge \psi_2) \\ \neg Gnd(\psi, S) & \text{if } \phi \text{ is } \neg\psi \\ (\bigwedge_{s \in S} Gnd(\psi\langle x \to \tilde{s}\rangle,\ S)) & \text{if } \phi \text{ is } \forall x\psi \\ (\bigvee_{s \in S} Gnd(\psi\langle x \to \tilde{s}\rangle,\ S)) & \text{if } \phi \text{ is } \exists x\psi \end{cases}$$

Example 1. *Let $\phi = (\forall x \forall y(Exy \to \neg(Gx \vee Gy)))$ and $S = \{a, b, c\}$. ϕ is part of the axiomatization for 3-Colouring, and says that no edge has both of its incident vertices coloured green. We have:*

$$
\begin{aligned}
Gnd(\phi, S) &= (\wedge_{s \in S}\, Gnd([\forall y(Exy \rightarrow \neg(Gx \vee Gy)]\langle x \rightarrow \tilde{s}\rangle, S)) \\
&= (Gnd([\forall y(Exy \rightarrow \neg(Gx \vee Gy))]\langle x \rightarrow \tilde{a}\rangle, S) \\
&\quad \wedge\ Gnd([\forall y(Exy \rightarrow \neg(Gx \vee Gy))]\langle x \rightarrow \tilde{b}\rangle, S) \\
&\quad \wedge\ Gnd([\forall y(Exy \rightarrow \neg(Gx \vee Gy))]\langle x \rightarrow \tilde{c}\rangle, S) \\
&= (Gnd(\forall y(E\tilde{a}y \rightarrow \neg(G\tilde{a} \vee Gy)), S) \\
&\quad \wedge\ Gnd(\forall y(E\tilde{b}y \rightarrow \neg(G\tilde{b} \vee Gy)), S) \\
&\quad \wedge\ Gnd(\forall y(E\tilde{c}y \rightarrow \neg(G\tilde{c} \vee Gy)), S)) \\
&= ([\wedge_{s \in S}\, Gnd((E\tilde{a}y \rightarrow \neg(G\tilde{a} \vee Gy))\langle y \rightarrow \tilde{s}\rangle, S)] \\
&\quad \wedge\ [\wedge_{s \in S}\, Gnd((E\tilde{b}y \rightarrow \neg(G\tilde{b} \vee Gy))\langle y \rightarrow \tilde{s}\rangle, S)] \\
&\quad \wedge\ [\wedge_{s \in S}\, Gnd((E\tilde{c}y \rightarrow \neg(G\tilde{c} \vee Gy))\langle y \rightarrow \tilde{s}\rangle, S)]) \\
&= ([Gnd(E\tilde{a}\tilde{a} \rightarrow \neg(G\tilde{a} \vee G\tilde{a})), S) \wedge Gnd(E\tilde{a}\tilde{b} \rightarrow \neg(G\tilde{a} \vee G\tilde{b})), S) \\
&\qquad \wedge Gnd(E\tilde{a}\tilde{c} \rightarrow \neg(G\tilde{a} \vee G\tilde{c})), S)] \\
&\quad \wedge\ [Gnd((E\tilde{b}\tilde{a} \rightarrow \neg(G\tilde{b} \vee G\tilde{a})), S) \wedge Gnd((E\tilde{b}\tilde{b} \rightarrow \neg(G\tilde{b} \vee G\tilde{b})), S) \\
&\qquad \wedge Gnd((E\tilde{b}\tilde{c} \rightarrow \neg(G\tilde{b} \vee G\tilde{c})), S)] \\
&\quad \wedge\ [Gnd((E\tilde{c}\tilde{a} \rightarrow \neg(G\tilde{c} \vee G\tilde{a})), S) \wedge Gnd((E\tilde{c}\tilde{b} \rightarrow \neg(G\tilde{c} \vee G\tilde{b})), S) \\
&\qquad \wedge Gnd((E\tilde{c}\tilde{c} \rightarrow \neg(G\tilde{c} \vee G\tilde{c})), S)]) \\
&= ([(E\tilde{a}\tilde{a} \rightarrow \neg(G\tilde{a} \vee G\tilde{a})) \wedge (E\tilde{a}\tilde{b} \rightarrow \neg(G\tilde{a} \vee G\tilde{b})) \wedge (E\tilde{a}\tilde{c} \rightarrow \neg(G\tilde{a} \vee G\tilde{c}))] \\
&\quad \wedge\ [(E\tilde{b}\tilde{a} \rightarrow \neg(G\tilde{b} \vee G\tilde{a})) \wedge (E\tilde{b}\tilde{b} \rightarrow \neg(G\tilde{b} \vee G\tilde{b})) \wedge (E\tilde{b}\tilde{c} \rightarrow \neg(G\tilde{b} \vee G\tilde{c}))] \\
&\quad \wedge\ [(E\tilde{c}\tilde{a} \rightarrow \neg(G\tilde{c} \vee G\tilde{a})) \wedge (E\tilde{c}\tilde{b} \rightarrow \neg(G\tilde{c} \vee G\tilde{b})) \wedge (E\tilde{c}\tilde{c} \rightarrow \neg(G\tilde{c} \vee G\tilde{c}))])
\end{aligned}
$$

In the above, we have shown the steps eliminating quantification in detail, but our last step combines several applications of Gnd to various sub-formulas.

3 Grounding with respect to a Structure

The function $Gnd(\phi, S)$ produces a ground formula based only on the set S. Typically, we have a structure $\mathcal{A}$ for part of the vocabulary of ϕ (e.g., the instance vocabulary $\mathcal{L}_{\mathcal{I}}$), and not only the universe. $\mathcal{A}$ gives interpretations to the symbols in $\mathcal{L}_{\mathcal{I}}$, and thus makes each ground atom for the symbols in $\mathcal{L}_{\mathcal{I}}$ that appears in $Gnd(\phi, S)$ either true or false. It follows that we can simplify the formula $Gnd(\phi, S)$ by evaluating out the atoms for

which we know the truth value.

Example 2. *Let $\mathcal{A} = (A, E^{\mathcal{A}}) = (\{a, b, c\}, \{\langle a, b\rangle, \langle b, c\rangle\})$ be a directed graph, and ϕ be as in Example 1. $\mathcal{A}$ gives us the interpretation for E, so we know exactly which ground atoms of the form Exy are true in $\mathcal{A}$. If we replace these atoms, in $Gnd(\phi, A)$, with their truth values, and then simplify the formula to eliminate occurrences of True and False (for example, we replace $(True \wedge P)$ with (P)), we obtain the much smaller formula*

$$(\neg(G\tilde{a} \wedge G\tilde{b}) \wedge \neg(G\tilde{b} \wedge G\tilde{c}))$$

which contains no occurrences of the symbol E.

Based on this observation, we extend the function *Gnd* to the case where the second argument is a structure. Then $Gnd(\phi, \mathcal{A})$ is the formula obtained by:

1. Compute the formula $\psi = Gnd(\phi, A)$;
2. Simplify ψ based on $\mathcal{A}$.

Intuitively, $Gnd(\phi, \mathcal{A})$ defines the structures with universe A that are solutions to (or witnesses for) the instance $\mathcal{A}$.

Proposition 1. *Let $\mathcal{A}'$ be an expansion of $\mathcal{A}$ to $\mathcal{L}$. Then $\mathcal{A}' \models \phi$ iff and only if $\tilde{\mathcal{A}}' \models Gnd(\phi, \mathcal{A})$.*

Proposition 2. *$Gnd(\phi, \mathcal{A})$ can be computed in time $O(|A|^{|\phi|})$.*

4 Solving by Grounding

$Gnd(\phi, \mathcal{A})$ is a FO formula, but it has no variables. Here, for simplicity, we also assume it has no function symbols. In this case, transformation to propositional logic is simple.

Let $\mathcal{F}$ be the set of all ground FO formulas with no function symbols, and let $\mathcal{CNF}$ be the set of all propositional formulas in conjunctive normal form. We define $ToSAT : \mathcal{F} \rightarrow \mathcal{CNF}$ to be the function that, for any ground function-free FO formula ϕ, is computed by:

1. For each distinct ground atom $P\bar{a}$ that appears in ψ, let $X_{P\bar{a}}$ be a distinct propositional atom.
2. Replace each occurrence of $P\bar{a}$ in ψ with $X_{P\bar{a}}$, to obtain a propositional formula.

3. Apply Tseitin's polynomial-time transformation to CNF.

Example 3. *$ToSAT((\neg(G\tilde{a} \wedge G\tilde{b}) \wedge \neg(G\tilde{b} \wedge G\tilde{c})))$ could be $((X_1) \wedge (\neg X_1 \vee \neg X_2 \vee \neg X_3) \wedge (\neg X_1 \vee \neg X_3 \vee \neg X_4))$*

Proposition 3. *If ψ is a ground, function free FO formula, then ψ is satisfiable if and only if $ToSAT(\psi)$ is.*

We may run a SAT solver with input $ToSAT(\psi)$ to determine if ψ is satisfiable. Moreover, given a satisfying assignment for $ToSAT(\psi)$, we can (assuming we recorded the mapping from ground atoms to propositional atoms), construct a model for ψ from a satisfying assignment for $ToSAT(\psi)$.

Finally, we may apply all of these steps to solve a problem instance from a specification.

Proposition 4. *There is an expansion $\mathcal{A}'$ of $\mathcal{A}$ to $\mathcal{L}$ such that $\mathcal{A}' \models \phi$ if and only if $ToSAT(Gnd(\phi, \mathcal{A}))$ is satisfiable.*

Moreover, we can construct an expansion of $\mathcal{A}$ that satisfies ϕ from a satisfying assignment to $ToSAT(Gnd(\phi, \mathcal{A}))$.

Proposition 5. *$\mathcal{A} \models \phi_K$ if and only if $ToSAT(\phi, \mathcal{A})$ is satisfiable.*

For a given $\exists$SO formula ψ of the form $\exists \overline{R} \phi$, where ϕ is a function-free FO sentence, we define the function $ToSAT_\psi : struct[\mathcal{L}] \to \mathcal{CNF}$ by

$$ToSAT_\psi(\mathcal{A}) = ToSAT(Gnd(\phi, \mathcal{A}))$$

.

Proposition 6. *If $\mathcal{K}$ is a class of structures defined by an $\exists$SO formula ψ, then $ToSAT_\psi$ is a polynomial-time many-one reduction from $\mathcal{K}$ to SAT.*

Thus, a program that implements the function $ToSAT(Gnd(\phi, \mathcal{A}))$ provides a universal reduction to SAT, and together with a SAT solver, provides a universal solver for problems in NP. That is, for any problem Π that is in NP, or is an NP search problem, there is a specification formula which makes the grounder implement a reduction from Π to SAT. The grounder (when ϕ is fixed) will run in polynomial time. The combination of grounder and SAT solver is a solver for Π.