Notes on Satisfiability-Based Problem Solving
Axiomatizing Problems

David Mitchell
mitchell@cs.sfu.ca

February 8, 2020

In this section we consider formulas as a way to specify computational problems. The key logical concept required is that of a formula defining a class of structures.

1 Defining a Class of Structures

If Φ is a set of formulas (of some logic L) for a vocabulary $\mathcal{L}$, then $Mod(\Phi)$ denotes the collection of all finite models of Φ. (For a single formula ϕ, we may write $Mod(\phi)$, rather than $Mod(\{\phi\})$.) That is, if we write $\text{Struct}[\mathcal{L}]$ for the class of all finite $\mathcal{L}$-structures, then

$$Mod(\Phi) = \{\mathcal{A} \mid \mathcal{A} \in \text{Struct}[\mathcal{L}] \text{ and for every } \phi \in \Phi, \mathcal{A} \models \phi\}.$$

Definition 1. Let $\mathcal{K}$ be a collection of $\mathcal{L}$-structures. We say that $\mathcal{K}$ is defined by a set Φ of $\mathcal{L}$-sentences iff $\mathcal{K} = Mod(\Phi)$.

We will almost always use vocabularies with =, so we henceforth assume implicity that vocabularies have =. If we wish to consider a vocabulary without equality we will state this explicitly.

Example 1. *Let$\mathcal{L} = (E)$, with E a binary relation symbol and $\Phi = \{\forall x \exists y Exy, \forall x \forall y \forall z Exy \land Exz \rightarrow y = z\}$. Then Φ defines the class of directed graphs in which every vertex has out-degree exactly one.*

Example 2. *Let $\mathcal{L} = (E, R, B, G)$, where E is a binary relation symbol and R, B, G are unary relation symbols. Let $\Phi = \{\forall v(Rv \lor Bv \lor Gv),\ \forall u \forall v[Euv \rightarrow \neg(Ru \land Rv) \land \neg(Bu \land Bv) \land \neg(Gu \land Gv)$. Then Φ defines the class of structures consisting of a graph (directed or undirected) and a labelling of its vertices (with labels R, B, G) so that no two adjacent vertices have the same label. In other words, the properly 3-coloured graphs (in which we allow a vertex to have more* *than one colour). Notice that, in the second formula, variables u and v might denote the same edge, so self-loops count as an edge with the same colour on both ends.*

Example 3. *Let $\mathcal{L} = (f)$, where f is a binary function symbol. For any $\mathcal{L}$-structure $\mathcal{M}$, $f^{\mathcal{M}}$ is a function $f^{\mathcal{M}} : M \times M \to M$. We can think of $f^{\mathcal{M}}$ as an $|M| \times |M|$ matrix with entries and indices all being elements of M. Now, let ϕ be*

$$((\forall r \forall c_1 \forall c_2 (c_1 \neq c_2 \to \ frc_1 \neq frc_2) \ \land \ (\forall c \forall r_1 \forall r_2 (r_1 \neq r_2 \to \ fr_1 c \neq fr_1 c)).$$

Then ϕ defines the class of Latin Squares.

Definition 2. A class $\mathcal{K}$ of structures if first order definable iff there exists a formula ϕ of FO such that $\mathcal{K} = Mod(\phi)$.

Any finite set Φ of sentences of FO is equivalent to a single formula consisting of the conjunction of the formulas in Φ, so we may take the ϕ in the definition to be a finite set of formulas.

Exercise 1. *Show that each of the following classes of graphs is FO definable, but writing a set of formulas that defines it.*

1. *The directed graphs containing at least one directed triangle.*
2. *The directed graphs consisting of exactly one directed triangle.*
3. *The directed graphs containing a directed cycle of length 5.*
4. *The directed graphs consisting of a union of disjoint directed triangles.*
5. *The undirected graphs consisting of a union of disjoint triangles.*
6. *The graphs consisting of a union of disjoint cycles. (Hint: don't think about describing cycles directly, think about simple local properties that a the vertices of simple cycles have.)*

Note that in normal practice, when we deal with undirected graphs, we view edges as sets. However, in our formalization, the set of edges is a set of 2-tuples, so ordered pairs. Here, a graph is undirected if E is symmetric. Sometimes the distinction will not matter, but in other cases it may. To require a graph be un-directed (as in 5), we must write a formula requiring E to be symmetric.

2 Decision Problems

A decision problem, as the term is used in computer science, is a problem that can be defined in the following form (which we have been using throughout these notes), where X and P are mathematically well-defined.

Given: X
Question: Does X have property P?

Example 4. *Simple examples include:*

1. *Given: graph G; Question: is G complete?*
2. *Given: graph G; Question: is G connected?*
3. *Given: a tuple $\langle G, R, B, Y$, where G is a graph and R, B, Y as sets of vertices of G (we can think of them as a labelling of the vertices of G); Question: Is R, B, Y a proper 3-colouring of G?*
4. *Given: graph G; Question: does G have a proper 3-colouring?*

Observe that the object X is always a finite structure. That is, it is a collection of finite sets, relations, and functions. Graphs are obvious examples. Labelled or weighted graphs consist of sets of vertices and edges, and one or more functions on vertices or edges giving the labels or weights. (The universe is the union of the elements in all the functions and relations.) In a job-shop scheduling problem, we have sets of machines, jobs, and times, and functions giving durations of jobs and possibly other information, such as precedences among jobs.

The question "Does X have property P" then, is the question of whether X belongs to the collection of all structures (of the appropriate sort) that have property P. As we have seen, we can define a class of structures having desired properties with a formula. Thus, if we let ϕ_P be a formula that defines the structures having our property P, then decision problems may be presented in the following manner.

Given: Finite structure $\mathcal{A}$
Question: Does $\mathcal{A} \models \phi_P$?

That is, decision problems correspond to model checking problems.

In the usual development of computability theory and computational complexity theory, a decision problem S corresponds to a set of strings over a finite alphabet, often $\{0, 1\}$. In this setting, a decision problem is of the form:

Given: String $s \in \{0,1\}^*$;
Question: Does s belong to S?;

For example, SAT is the problem: Given a propositional CNF formula, does it belong to the set of strings which are satisfiable CNF formulas? Most real problems are not naturally problems on strings, so we assume (or establish) a "reasonable" encoding of the real-world entities of interest as strings. Then "does graph G have property P" becomes "does the string encoding G belong to the set of strings encoding graphs with property P".

This formalization is often mathematically convenient, and is natural in light of the fact that real computers do operate on sequences of bits or bytes However, in many respects, the logical setting is more natural, in that we talk more directly about the objects of interest. A correspondence between the two settings is established easily by adopting a reasonable encoding of finite structures as strings. Then "$\mathcal{A} \models \phi_P$" is essentially the same as "the string encoding $\mathcal{A}$ belongs to the set S of strings encoding structures with property P", which is essentially the same as $\mathcal{A} \in \mathcal{P}$, where $\mathcal{P}$ denotes the structures having property P. The logical setting also leads to a fruitful study involving the expressiveness of the languages (logics) used for ϕ, since different languages can describe different classes of structures. In some cases, languages correspond exactly to computational complexity classes.

It is natural to ask whether *every* decision problem corresponds exactly to a model checking problem for some formula of some logic. For most practical purposes the answer is yes, although addressing the question with precision is beyond the scope of these notes.

There are however, many simple problems which are not FO definable.

Example 5. *The following classes of graphs are not FO definable.*

1. *The graphs of odd order.*
2. *The connected graphs.*
3. *The graphs consisting of exactly one simple cycle.*
4. *The graphs that are transitively closed.*
5. *The 3-colourable graphs.*

Fact 1.

1. *Every FO-definable class of structures is in the complexity class AC^0.*
2. *$AC^0 \subsetneq P$. That is, there are polytime-decidable sets which are known not to be in AC^0. Examples of problems which are known to be in P and to not be in AC^0 are Parity (is the size of a set even or odd) and Integer Multiplication (with standard representations of integers).*

We may conclude from the examples of structures which are not FO-definable that, for practical problems solving systems, we will want to use languages with greater expressive power than FO. We will will do so in the following section. Here we observe that for some classes which are not FO-definable, we can define a closely related class by introducing additional vocabulary symbols. For example, the class of 3-colourable graphs is not FO-definable, but the class of properly 3-coloured graphs is, as shown in Example 2.

3 Problems Definable in Existential Second Order Logic

Second order logic (SO) extends FO syntactically by allowing quantification over relations and functions. That is, in addition to quantification over first order variables, which range over elements of the universe, we have quantification over second order variables, which range over sets, relations and functions. The formula $(\forall x(Px))$ says that every universe element is in the set denoted by P. The formula $(\forall P(Pa))$ says that every set P of elements from the universe contains the element denoted by a. (This formula is unsatisfiable because the empty set is one of the possible values for the second order variable P.) The formula

$$\forall P\ \forall x\ (Px \lor \neg Px)$$

is a second order formula in which P is a universally quantified second order variable and x is a universally quantified first order variable. The formula states the principle of bivalence: For any set P, and any object x, either x is in P or x is not in P.

The existential fragment of SO, called existential second order logic, and denoted ∃SO (or sometimes SO∃), allows only existential quantification over second order variables. For most purposes, it is sufficient to consider only formulas of the form

$$\exists R_1 \exists R_2 \ldots \exists f_1 \exists f_2 \ldots \psi$$

where the R_i are relation variables, the f_i are function variables, and ψ is a formula of FO. Note that the second order variables (the function and relation variables) are not part of the vocabulary of the formula.

For brevity, we sometimes write $\exists SO$ formulas as $\exists\bar{R}\exists\bar{f}\psi$, where $\bar{R}, \bar{f}$ denote, respectively, a tuple of relation variables and a tuple of function variables.

We give the semantics of $\exists SO$, as we did for propositional logic and FO, by defining the satisfaction relation. Informally, a structure $\mathcal{M}$ for the vocabulary of $\exists SO$ formula $\phi = \exists\bar{R}\exists\bar{f}\psi$ satisfies ϕ if and only if there are choices for the second order variables that, together with $\mathcal{M}$, make ψ true. Formally, we use the notion of expansions of a structure.

Definition 3. Let $\mathcal{L}$ and $\mathcal{L}'$ be vocabularies, with $\mathcal{L} \subsetneq \mathcal{L}'$, and let $\mathcal{M}$ be an $\mathcal{L}$-structure. Then $\mathcal{L}'$-structure $\mathcal{M}'$ is an expansion of $\mathcal{M}$ to $\mathcal{L}'$ iff

1. $M' = M$,
2. For each relation symbol $R \in \mathcal{L}$, $R^{\mathcal{M}} = R^{\mathcal{M}'}$,
3. For each function symbol $f \in \mathcal{L}$, $f^{\mathcal{M}} = f^{\mathcal{M}'}$.

That is, $\mathcal{M}'$ is just like $\mathcal{M}$, except that it gives interpretations to all the symbols of $\mathcal{L}'$, not just those in $\mathcal{L}$.

Example 6. *Let $\mathcal{L} = (R, S, f, g)$ and $\mathcal{L}' = (R, S, f, g, P, h, c)$, and let $\mathcal{M}$ be a $\mathcal{L}$-structure and $\mathcal{M}'$ be an expansion of $\mathcal{M}$ to $\mathcal{L}'$. Then, schematically, we have:*

$$\begin{aligned} \mathcal{M} &= (M, \overbrace{R^{\mathcal{M}}, S^{\mathcal{M}}, f^{\mathcal{M}}, g^{\mathcal{M}}}^{\mathcal{L}}) \\ \mathcal{M}' &= (M, \underbrace{\overbrace{R^{\mathcal{M}}, S^{\mathcal{M}}, f^{\mathcal{M}}, g^{\mathcal{M}}}^{\mathcal{L}}, P^{\mathcal{M}'}, h^{\mathcal{M}'}, c^{\mathcal{M}'})}_{\mathcal{L}'} \end{aligned}$$

Example 7. *Let $\mathcal{L} = (E)$ and $\mathcal{L}' = (E, R, B, G)$. Then*

$$\mathcal{G} = \langle\{1,2,3\}, \{(1,2), (2,3), (3,1)\}\rangle$$

(consisting of a single triangle) is an $\mathcal{L}$-structure, and

$$\mathcal{G}' = \langle\{1,2,3\}, \{(1,2), (2,3), (3,1)\}, \{\}, \{1,2\}, \{3\}\rangle$$

is an expansion of $\mathcal{G}$ to $\mathcal{L}'$ (in which vertices 1,2 are "coloured blue", and vertex 3 is "coloured green").

Example 8. *Let $\mathcal{L} = (E)$, $\mathcal{L}' = (E, R, B, Y)$, and let A be the FO formula from Example 2 that defines the class of properly 3-coloured graphs. Then for every graph $G = \langle V, E^{\mathcal{G}} \rangle$, the expansions of $\mathcal{G}$ to $\mathcal{L}'$ that satisfy A are the properly 3-coloured "copies" of $\mathcal{G}$.*

Definition 4. ($\exists$SO Satisfaction) Let ϕ be an $\exists$SO formula $\exists \bar{R} \exists \bar{f} \psi$, where ψ is a FO formula. Further, let $\mathcal{L}$ be the vocabulary of ϕ and $\mathcal{L}' = \mathcal{L} \cup \{\bar{R}\} \cup \{\bar{f}\}$ the vocabulary of ψ. Then $\mathcal{L}$-structure $\mathcal{M}$ satisfies ϕ, written $\mathcal{L} \models \phi$, iff there is an expansion $\mathcal{M}'$ of $\mathcal{M}$ to $\mathcal{L}'$, such that $\mathcal{M}' \models \psi$.

The definitions of satisfiability, finite satisfiability, validity, and logical consequence, are all defined in terms of the satisfaction relation, exactly as for FO.

Definition 5. A class $\mathcal{K}$ of $\mathcal{L}$-structures is $\exists$SO-definable iff there is an $\exists$SO sentence ϕ such that $\mathcal{M} \in \mathcal{K}$ iff $\mathcal{M} \models \phi$.

The following definition is equivalent.

Definition 6. A class $\mathcal{K}$ of $\mathcal{L}$-structures is $\exists$SO-definable iff there is a FO sentence A such that $\mathcal{M} \in \mathcal{K}$ iff there is an expansion $\mathcal{M}'$ of $\mathcal{M}$ such that $\mathcal{M}' \models A$.

Example 9. *Let $\phi = \exists R \exists B \exists G \psi$, where ψ is the conjuction of two FO formulas in Φ from Example 2 that defines the class of properly 3-coloured graphs. Then ϕ is an $\exists$SO formula which defines the class of 3-colourable graphs.*

Definition 7. A set S of strings is in the complexity class NP iff there exists a binary relation R, on strings, and a polynomial p such that

1. Deciding if $\langle x, y \rangle \in R$ can be done in time $p(|x| + |y|)$;
2. $x \in S \Leftrightarrow \exists y$ s.t. $|y| \leq p(|x|)$ and $R(x, y)$.

If $x \in S$, we call a y s.t. $\langle x, y \rangle$ satisfies the the second condition of Definition 7, a witness or certificate for x.

To make the connection between computational complexity and classes of structures, we choose any reasonable encoding of finite structures as strings.

Theorem 1 (Fagin)**.** *A class $\mathcal{K}$ of finite $\mathcal{L}$-structures is $\exists$SO-definable off it is in NP.*

Proof. (Idea) For $\Rightarrow$), let ϕ be $\exists \bar{R} A$. Observe that, for a $\mathcal{L}$-structure $\mathcal{M}$, if $\mathcal{M}'$ is an expansion of $\mathcal{M}$ that satisfies A, then the string encoding $\mathcal{M}'$ is of size at most n^k, where n is the size of the string encoding $\mathcal{M}$, and k is the sum of the arities of the symbols in $\bar{R}$. Along with the fact that FO model checking is in P, this establishes membership in NP, because the encoding of $\bar{R}^{\mathcal{M}'}$ is a polysize certificate for $\mathcal{M}$, and FO model checking is a polytime verifier. For $\Leftarrow$), we consider a polytime bounded turing machine T, and write a formula describing the accepting computations of T. $\square$

4 Specification of NP Search Problems

A search problem is a binary relation R, where $\langle x, y\rangle \in R$ iff y is a solution to problem instance x. When the set $S = \{x : \exists y R(x, y)\}$ is in NP, we say that the decision problem associated with R is in NP, and that R is an "NP search problem".

By the results of the previous section, we can consider an $\exists$SO formula as a specification of an NP search problem. If $\exists\bar{R}\psi$ is an $\exists$SO formula, the corresponding relation $R(x, y)$ holds between $\mathcal{L}$-structures x which are models of $\exists\bar{R}\psi$, and expansions y of those structures which are models of ψ. Equivalently, we can consider the pair $\langle\psi, \mathcal{L}\rangle$, where ψ is a FO sentence and $\mathcal{L}$ a subset of the vocabulary of ψ, to be a specification of the same search problem as $\exists\bar{R}\psi$.

Example 10. *Let ψ be the FO formula defining the class of properly 3-coloured graphs, as in Example 2. Then $\langle\psi, E\rangle$ can be regarded as a specification for the problem Graph 3-Colouring.*

Viewed in this logical way, an instance of a search problems is a finite structure, and a solution is an expansion of that structure to include new relations or functions that constitute a solution for the instance. The corresponding formal computational problem involves strings encoding these logical elements. The logical view is the way we normally think about our problems. For example, in graph colouring our interest is in colouring the graph, not the string that encodes the graph. We use a string to encode the graph as a pragmatic step to facilitate computation.

5 Defining a Relation in a Structure

In defining interesting classes of structures, it is often useful to think of particular subformulas as defining certain sets. This notion can be formalized.

Consider our vocabulary $\mathcal{L}_A$, and the standard $\mathcal{L}_A$ structure $\mathcal{N}$. Every $\mathcal{L}_A$ sentence is either true for false in $\mathcal{N}$. As we observed earlier, if a formula has free variables, its truth in a structure depends on what objects those free variables denote. For example, $\mathcal{N} \models x = sy\ [\sigma]$ iff σ is such that $\sigma(x) = \sigma(y) + 1$. A bit more interesting is the formula $\exists y(x = y + y)$, which is true in $\mathcal{N}$ with σ iff $\sigma(x)$ is an even number. We say that $\exists y(x = y + y)$ defines the set of even numbers in $\mathcal{N}$.

Notation: We write $\phi(x_1, \ldots x_n)$ to indicate that the free variables of ϕ are among $x_1, \ldots x_n$. So, if we let *Even* be the formula $\exists y(x = y + y)$, we may write that $Even(x)$ defines the even numbers in $\mathcal{N}$. We sometimes also use tuple notation, for example $\phi(\bar{x})$, to indicate that $\bar{x}$ is a tuple of variables which includes all the free variables of ϕ.

Definition 8. The relation defined by formula $\phi(x_1, \ldots x_k)$ in structure $\mathcal{A}$ is

$$\{\langle a_1, \ldots a_k \rangle \in A^k \mid \mathcal{A} \models \phi(x_1, \ldots x_k)\ [\sigma(x_1 \mapsto a_1) \ldots (x_k \mapsto a_k)]\}.$$

We sometimes write $\phi(\bar{x})^{\mathcal{M}}$ for the relation defined by ϕ in $\mathcal{M}$.

In Definition 8, σ does not matter, because we have explicitly enumerated the assignments to the variables that matter. Henceforth, when this is the case, we may leave σ out altogether, writing $[(x \mapsto a)...]$ instead of $[\sigma(x \mapsto a)...]$

Example 11.

1. *$x + y = sssss0$ defines, in $\mathcal{N}$, the set of pairs of numbers which sum to 5.*
2. *If $\mathcal{G} = \langle V, E \rangle$ is a graph, then $Euv \wedge Evw \wedge Ewu \wedge u \neq v \wedge v \neq w \wedge w \neq u$ is a formula with free variables u, v and w that defines the set of triangles in $\mathcal{G}$.*

Exercise 2.

1. *Write a formula $Odd(x)$ that defines the odd natural numbers in $\mathcal{N}$.*
2. *Write a formula $Source(x)$ that defines the set of vertices with in-degree zero, in any directed graph $\mathcal{G} = \langle V, E \rangle$.*

Exercise 3. *Suppose $Prime(x)$ is a formula that defines the prime numbers in $\mathcal{N}$, $Even(x)$ is as described previously, and $x < y$ — which is an abbreviation for $<(x, y)$ — defines the standard ordering on the naturals. Then Goldbach's conjecture, that every even integer greater than 2 is the sum of two primes, is expressed by*

$$\forall x((Even(x) \wedge x > 2) \rightarrow \exists y \exists z(Prime(y) \wedge Prime(z) \wedge x = y + z)).$$

Write formulas $Prime(x)$ and $(x > 2)$, in the vocabulary $\mathcal{L}_{\mathcal{A}}$, that define the primes and the numbers greater than 2 (respectively) in $\mathcal{N}$.