Finitely-Verifiable Classes of Sentences

Fangzhen Lin
Department of Computer Science
Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong

Abstract

This paper proposes a notion of finitely-verifiable classes of sentences. Informally, a class of sentences is finitely-verifiable if whether a sentence in this class is a theorem of a given theory can be checked with respect to a finite set of models of the theory. The usefulness of this notion is illustrated using examples from arithmetics, first-order logic, game theory, and planning.

Introduction

Theorem discovery is the process of coming up with interesting and useful consequences of a theory, and is a highly creative human endeavor. Computer-aided theorem discovery attempts to make this discovery process as automatic as possible. One example is from combinatorial mathematics. *The American Mathematical Monthly* used to have problems about the computation of hypergeometric identities such as

$$f(n) = \sum_{0 \leq k \leq n/3} 2^k \frac{n}{n-k} \left(\begin{array}{c} n-k \\ 2k \end{array} \right)$$

in problem 10424. With the work of Sister Mary Celine Fasenmyer, Gosper, Petkovsek, Wilf, and Zeilberger (cf. [Petkovsek *et al.*, 1996]), the discoveries and the proofs of these hypergeometric identities have been largely automated. For instance, using the programs given in [Petkovsek *et al.*, 1996], it is a simple matter to compute that $f(n) = 2^{n-1} + cos(n\pi/2)$ for the above identity.

More recently in AI, Lin [2003; 2004] showed that for certain classes of causal theories, the discoveries and proofs of equivalent action theories in the forms of successor state axioms and STRIPS-like systems can be automated, and for certain types of actions theories, the discoveries and proofs of certain types of state invariants can be automated as well. Lin and Chen [2005] showed that the discoveries of certain classes of strongly equivalent logic programs under answer set semantics can also be automated to certain extent.

In this paper we shall propose a notion of finitely-verifiable classes of sentences that we believe underpins recent work on computer-aided theorem discovery. Briefly, a class of sentences is finitely-verifiable if there is a finite set of models such that whether a sentence in the class is a theorem can be proved by checking whether it is true in all models in this finite set. Thus if the theorems that one is looking for fall into such a class, then in principle one could discover them by going through the sentences in this class one by one to see if any of them is a theorem, assuming that whether a sentence is true in a model can be checked effectively. In practice, though, such brute-force search may not work and some effective search control strategies need to be used.

As we shall see, the formal definition of finitely-verifiable classes of sentences is actually very simple. The main part of this paper is to give examples to show that this notion is general enough to cover many interesting cases.

The rest of the paper is organized as follows. We define formally the notion of finitely-verifiable classes of sentences in a given theory in the next section. We then show how this notion can be applied in arithmetics for computing the polynomial sum of a series of polynomials, in first-order logic for checking the validity of some prenex first-order sentences, in game theory for discovering classes of two-person games with unique Nash equilibria, and in planning for discovering state invariants.

Basic Definitions

In this paper, a *theory* is defined to be a triple $(\mathcal{L}, \mathcal{M}, \rhd)$, where $\mathcal{L}$ is a set of sentences called the *language* of the theory, $\mathcal{M}$ a set called the class of *models* of the theory, and $\rhd$ a subset of $\mathcal{M} \times \mathcal{L}$ called the *satisfiability relation* of the theory.

Given a theory $\mathcal{T} = (\mathcal{L}, \mathcal{M}, \rhd)$, a sentence φ in $\mathcal{L}$ is said to be a *theorem* of $\mathcal{T}$ iff for all $M \in \mathcal{M}$, $M \rhd \varphi$.

A set S of sentences in $\mathcal{T}$ is said to be *finitely-verifiable* if there is a finite set Δ of models such that for any sentence φ in S, φ is a theorem of $\mathcal{T}$ iff for all $M \in \Delta$, $M \rhd \varphi$. In this case, we also say that S is finitely-verifiable on Δ. The hardness of proving and discovering theorems in S can be measured by the size of the smallest Δ on which S is finitely-verifiable.

Clearly, if φ is a theorem, then $\{\varphi\}$ is finitely-verifiable. This is like saying that if a given set of clauses in propositional logic is satisfiable, one can determine in constant time whether this set of clauses is satisfiable. It is also clear that if S is finite, then it is finitely-verifiable. However, even in

this case, computing a finite set Δ of models on which S is finitely-verifiable may not be trivial. For instance, let $\mathcal{L}$ be the set of sentences in a first-order language, $\mathcal{M}$ the set of first-order structures for this language, and $\rhd$ the usual satisfiability relation. If S is the set of sentences that are no more than, say 100 characters long, then S is a finite set thus finitely-verifiable on the set

$$\{M_\varphi \mid \varphi \in S, \text{ and } \neg\varphi \text{ is satisfiable}\},$$

where M_φ is any model that satisfies $\neg\varphi$. However, it is not clear how this set can be actually computed.

We also remark that if the set $\mathcal{M}$ of models in a theory is finite, then any set of sentences is trivially finitely-verifiable. Thus the notion of finitely-verifiable classes of sentences becomes trivial when, say, $\mathcal{M}$ is a singleton consisting of the standard model of number theory.

Computing the Sum of a Series of Polynomials

Our first example is from arithmetics, for computing the sum of a series of polynomials. It is a simple case of the problem mentioned above for computing hypergeometric identities.

The sentences in this theory are strings of form

$$\sum_{i=0}^{i=x} p_1(i) = p_2(x), \tag{1}$$

where p_1 is a polynomial in i, and p_2 a polynomial in x, and their coefficients are real numbers. We call $max\{k_1, k_2\}$ the *degree* of the sentence, where k_i is the degree of the polynomial $p_i, i = 1, 2$. A model of the theory is an assignment that maps x to a non-negative integer. A model $x = n$ satisfies the sentence

$$\sum_{i=0}^{i=x} p_1(i) = p_2(x)$$

if

$$\sum_{i=0}^{i=n} p_1(i) = p_2(n)$$

is true.

In this language, for any finite number k, the set S of sentences whose degrees are no greater than k is finitely-verifiable: for any sentence in S, it is a theorem iff it is satisfied by all models in $\{x = 0, ..., x = k, x = k + 1\}$. "Only if" is obvious. To prove the "if" part, suppose

$$\sum_{i=0}^{i=x} p_1(i) = p_2(x)$$

is in S, and that it is true for $x = 0, ..., k, k + 1$. We show by induction that it is true for all x. The base case for $x = 0$ is true by assumption. Inductively, suppose it is true for $x = n - 1$, we need to show that it is true for $x = n$:

$$\sum_{i=0}^{i=n} p_1(i) = p_2(n).$$

By the inductive assumption, this is equivalent to

$$p_2(n) - p_2(n - 1) - p_1(n) = 0.$$

By our assumption, this equation holds for $n = 1, ..., k, k + 1$, and that $p_2(n) - p_2(n - 1) - p_1(n)$ is a polynomial of degree at most k. Thus this equation must be true for all n, for otherwise the polynomial $p_2(n) - p_2(n-1) - p_1(n)$ will have $k+1$ distinct solution, a contradiction to the fundamental theorem of algebra.

Thus to discover a polynomial f_2 such that (1) holds, one could assume a general form for $f_2(x) = c_0 + c_1x + \cdots c_kx^k$, let $k = 0$ to begin with and increase it by one on each iteration until one finds a k and $k + 1$ numbers $c_0, ..., c_k$ such that (1) holds for $x = 0, ..., k + 1$.

$\forall\exists$-Prenex Formulas in First-Order Logic

Consider the theory $(\mathcal{L}, \mathcal{M}, \rhd)$, where $\mathcal{L}$ is the set of sentences in a finite first-order language, $\mathcal{M}$ the class of first-order structures, and $\rhd$ the usual satisfiability relation $\models$.

In the following, for each natural number n, we denote by $\mathcal{M}^n$ the set of structures whose domain is $\{1, 2, ..., n\}$:

$$\mathcal{M}^n = \{M \mid M \in \mathcal{M} \wedge domain(M) = \{1, 2, ..., n\}\}.$$

One can consider $\mathcal{M}^n$ to be the set of *canonical* first-order structure with a domain of size n.

We call a formula of the form $\forall\vec{x}Q$ a $\forall$-*prenex* formula, where Q contains no quantifiers or proper function symbols. We call the sum of the number of variables in $\vec{x}$ and the number of constants in Q the *degree* of the $\forall$-prenex formula. Similarly, a $\forall\exists$-prenex formula is one of the form $\forall\vec{x}\exists\vec{y}Q$, where Q contains no quantifiers or proper function symbols, and its degree is the sum of the number of variables in $\vec{x}$ and the number of constants in Q. An $\exists\forall$-prenex formula is one of the form $\exists\vec{x}\forall\vec{y}Q$, where Q contains no quantifiers or proper function symbols, and its degree is the sum of the number of variables in $\vec{x}$ and the number of constants in Q.

Proposition 1 *For any finite number k, the set of $\forall\exists$-prenex sentences whose degrees are not greater than k is finitely-verifiable on $\bigcup_{i=1}^{i=n} \mathcal{M}^i$, where $n = 1$ if $k = 0$, and $n = k$ otherwise.*

This proposition follows from the following simple lemma in first-order logic (c.f. Exercise 19(a), page 96, of [Enderton, 1972]):

Lemma 1 *If an $\exists\forall$-prenex sentence is satisfiable, then it is true in an interpretation with a domain of at most $max\{1, k\}$ objects, where k is the degree of the prenex sentence.*

Proposition 1 (Lemma 1) can be extended to entailment under a universal theory.

Let $\mathcal{L}$ be again the set of sentences in a finite first-order language. Let Σ be a set of $\forall$-prenex sentences. Let $\mathcal{M}_\Sigma$ be the set of models of Σ, and $\rhd$ the usual satisfiability relation $\models$. Let $\mathcal{M}^n_\Sigma = \mathcal{M}_\Sigma \cap \mathcal{M}^n$.

Proposition 2 *If S is a set of $\forall\exists$-prenex sentences such that the degree of each sentence in S is less than k for some fixed finite k, then S is finitely-verifiable on $\bigcup_{i=1}^{i=n} \mathcal{M}^i_\Sigma$ under the theory $(\mathcal{L}, \mathcal{M}_\Sigma, \models)$, where n is the maximal element in the following set:*

$\{1\} \cup$

$\{degree(\varphi) +$ *(number of constants in* Σ *but not in* φ*)* $\mid \varphi \in S\}$

Proof: First of all, notice that since our first-order language is finite, there is only a finite number of constants in the language, thus only finite number of them in Σ. Therefore the number n in the proposition is well-defined. To prove this proposition, we only need to prove that for any formula φ in S, if there is a $M \in \mathcal{M}_\Sigma$ such that $M \models \neg\varphi$, then there is a model $M' \in \mathcal{M}^i_\Sigma$, $1 \leq i \leq n$, such that $M' \models \neg\varphi$. Suppose that $\forall\vec{x}\exists\vec{y}Q$ is in S, $M \in \mathcal{M}_\Sigma$, and $M \models \neg\forall\vec{x}\exists\vec{y}Q$. Thus there is a variable assignment σ such that

$$M, \sigma \models \forall\vec{y}\neg Q.$$

Now let D be the following set:

$$\{\sigma(x) \mid x \in \vec{x}\} \cup$$
$$\{c^M \mid a \text{ is a constant occurring in } \Sigma \cup \{Q\}\}.$$

Now let M_1 be a structure with domain D (if D is empty then let its domain be any singleton set) and

- The interpretation of each predicate in M_1 is the restriction of its interpretation in M on D.
- For each constant c mentioned in $\Sigma \cup \{Q\}$, $c^{M_1} = c^M$.

It is clear that for any such M_1,

$$M_1, \sigma \models \forall\vec{y}\neg Q,$$

thus $M_1 \models \forall\vec{x}\exists\vec{y}Q$. It is also clear that $M_1 \in \mathcal{M}_\Sigma$. By the definition of D, its size is not greater than the number n in the proposition. Thus $M_1 \in \mathcal{M}^i$ for some $1 \leq i \leq n$. ■

These results can be extended to first-order logic with many sorts. To do this, we first need to extend the notion of degrees of a prenex formula.

A *rank* τ of a many-sorted first-order language is a set of the form

$$\tau = \{(g, n) \mid g \text{ a primitive sort, and } n \text{ an ordinal}\}.$$

We say that τ is a finite rank if for all $(g, n) \in \tau$, n is finite. Given two ranks τ and τ', we say that τ is smaller or equal to τ', written $\tau \leq \tau'$ if for each sort g, $(g, n) \in \tau$, and $(g, n') \in \tau'$, $n \leq n'$.

Now the degree of a $\forall\exists$-prenex formula $\forall\vec{x}\exists\vec{y}Q$ is the rank τ defined as follows: for each primitive sort g, if n is the sum of the number of variables that may be of sort g in $\vec{x}$ and the number of constants in Q that may be of sort g, then $(g, n) \in \tau$, where a variable (constant) may be of sort g if it is either of sort g or of sort g' that contains g.

Similarly the degree of an $\exists\forall$-prenex formula $\exists\vec{x}\forall\vec{y}Q$ is the rank τ defined as follows: for each primitive sort g, if n is the sum of the number of variables that may be of sort g in $\vec{x}$ and the number of constants in Q that may be of sort g, then $(g, n) \in \tau$.

A first-order structure is said to be a *τ-structure* if for each $(g, n) \in \tau$, the domain of sort g in the structure has at most n elements. Similarly, a τ-model of a sentence (theory) is a τ-structure that satisfies the sentence (theory). Notice that the domain of a non-primitive sort is computed from domains of primitive sorts.

The following lemma generalizes Lemma 1 ([Lin, 2004]).

Lemma 2 *If an $\exists\forall$-prenex formula in a many-sorted first-order language is satisfiable, then it is satisfiable in a τ-structure, where the rank τ is defined as follows: for any primitive sort g, if (g, n) is in the degree of the prenex formula, then $(g, max\{n, 1\}) \in \tau$.*

To generalize Proposition 2 to many-sorted case, we need to first generalize $\mathcal{M}^n_\Sigma$, the set of canonical models of size n. In the following, let L be the set of sentences in a finite many-sorted first-order language, Σ a set of $\forall$-prenex sentences, and $\mathcal{M}_\Sigma$ the set of models of Σ. For each finite rank τ, let $\mathcal{M}^\tau_\Sigma$ be the set of models M in $\mathcal{M}_\Sigma$ such that for each primitive sort g and $(g, n) \in \tau$, the domain of M for sort g is $\{g.1, g.2, ..., g.n\}$.

Proposition 3 *Let υ be a finite rank, and S a set of $\forall\exists$-prenex sentences such that the rank of each sentence in S is less than or equal to υ. Then S is finitely-verifiable on $\bigcup_{\xi \leq \tau} \mathcal{M}^\xi_\Sigma$ under the theory $(L, \mathcal{M}_\Sigma, \models)$, where τ is the rank such that for each primitive sort g, $(g, n) \in \tau$ if n is the maximal element in the following set:*

$$\{1\} \cup \{k +$$ *(number of constants may be of sort g in Σ but not in φ)* $\mid var \in S, (g, k) \in degree(\varphi)\}$

In the following we show how these theorems can be used for discovering theorems in two-person game theory and planning.

Two-Person Games

The class of two-person games has been studied extensively. Some important concepts and classical results in games theory were initially done for such games.

A two-person game is a tuple $(A, B, \leq_1, \leq_2)$, where A and B are sets of (pure) strategies of players 1 and 2, respectively, and $\leq_1$ and $\leq_2$ are total orders on $A \times B$ called *preference relations* for players 1 and 2, respectively.

For each $b \in B$, the set of best responses by player 1 to the action b by player 2 is defined as follows:

$$B_1(b) = \{a \mid a \in A, \text{ and for all } a' \in A, (a', b) \leq_1 (a, b)\}.$$

Similarly, for each $a \in A$, the set of best responses by player 2 is:

$$B_2(a) = \{b \mid b \in B, \text{ and for all } b' \in B, (a, b') \leq_2 (a, b)\}.$$

A profile $(a, b) \in A \times B$ is a (pure-strategy) Nash equilibrium if both $a \in B_1(b)$ and $b \in B_2(a)$. A game can have one, more than one, or no Nash equilibria. There has been extensive study of properties of pure Nash equilibria. For instance, if a game is *strictly competitive* [Friedman, 1983; Moulin, 1976], then it has unique Nash equilibria in the sense that if both s and s' are Nash equilibria of the game, then the payoffs for them are the same for each player, i.e. $s \leq_i s'$ and $s' \leq_i s$ for $i = 1, 2$. This is also true for *weakly unilaterally competitive* games [Kats and Thisse, 1992]. It is also known that *ordinal potential* games [Topkis, 1998] and *super-modular* games [Monderer and Shapley, 1996] always have Nash equilibria.

To discover similar results, we first formulate two-person games in first-order logic.

Formulating two-person games in first-order logic

We consider a first-order language with two sorts α and β, equality, and two predicates $\leq_1$ and $\leq_2$. Sort α is for player 1's actions, and β for player 2's actions. In the following, we use variables $x, x_1, x_2, ..$ to range over α, and $y, y_1, y_2, ...$ to range over β. The two predicates represent the two players' preference relations. In the following, as we have already done above, we write $\leq_i (x_1, y_1, x_2, y_2)$ in infix notation as $(x_1, y_1) \leq_i (x_2, y_2), i = 1, 2$. We write $(x_1, y_1) <_i (x_2, y_2)$ as a shorthand for

$$(x_1, y_1) \leq_i (x_2, y_2) \wedge \neg(x_2, y_2) \leq_i (x_1, y_1),$$

and $(x_1, y_1) \simeq_i (x_2, y_2)$ as a shorthand for

$$(x_1, y_1) \leq_i (x_2, y_2) \wedge (x_2, y_2) \leq_i (x_1, y_1),$$

where $i = 1, 2$.

The two relations need to be total orders (in the rest of the paper, unless otherwise stated, all free variables in a displayed formula are assumed to be universally quantified from outside):

$$(x, y) \leq_i (x, y), \tag{2}$$

$$(x_1, y_1) \leq_i (x_2, y_2) \vee (x_2, y_2) \leq_i (x_1, y_1), \tag{3}$$

$$(x_1, y_1) \leq_i (x_2, y_2) \wedge (x_2, y_2) \leq_i (x_3, y_3) \supset (x_1, y_1) \leq_i (x_3, y_3), \tag{4}$$

where $i = 1, 2$. In the following, we denote by Σ the set of the above sentences. Thus two-person games correspond to first-order models of Σ, and two-person finite games correspond to first-order finite models of Σ.

We now show how some other notions in game theory can be formulated in first-order logic. The condition for a profile (ξ, ζ) to be a Nash equilibrium is captured by the following formula:

$$\forall x.(x, \zeta) \leq_1 (\xi, \zeta) \wedge \forall y.(\xi, y) \leq_2 (\xi, \zeta) \tag{5}$$

In the following, we shall denote the above formula by $NE(\xi, \zeta)$.

The following sentence expresses the uniqueness of Nash equilibria:

$$NE(x_1, y_1) \wedge NE(x_2, y_2) \supset (x_1, y_1) \simeq_1 (x_2, y_2) \wedge (x_1, y_1) \simeq_2 (x_2, y_2) \tag{6}$$

A game is strictly competitive if it satisfies the following property:

$$(x_1, y_1) \leq_1 (x_2, y_2) \equiv (x_2, y_2) \leq_2 (x_1, y_1). \tag{7}$$

Thus it should follow that

$$\Sigma \models (7) \supset (6). \tag{8}$$

Notice that we have assumed that all free variables in a displayed formula are universally quantified from outside. Thus (7) is a sentence of the form $\forall x_1, x_2, y_1, y_2 \varphi$. Similarly for (6).

Theorems like (8) can actually be generated automatically using the following theorem, which follows from the results in first-order logic in the last section.

Theorem 1 *Suppose Q is a formula without quantifiers, $\vec{x}_1$ and $\vec{x}_2$ tuples of variables of sort α, and $\vec{y}_1$ and $\vec{y}_2$ tuples of variables of sort β. We have that*

1. $\Sigma \models \exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q \supset (6)$ *iff for all model G of Σ such that $|A| \leq |\vec{x}_1| + 2$ and $|B| \leq |\vec{y}_1| + 2$, we have that $G \models \exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q \supset (6)$, where A is the domain of G for sort α, and B the domain of G for sort β.*
2. $\Sigma \models \exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q \supset \neg\exists x, y.NE(x, y)$ *iff for all model G of Σ such that $|A| \leq |\vec{x}_1| + 1$ and $|B| \leq |\vec{y}_1| + 1$ we have that $G \models \exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q \supset \neg\exists x, y.NE(x, y)$, where A is the domain of G for sort α, and B the domain of G for sort β.*

Proof: (1) By Proposition 3, noting that

$$\exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q \supset (6)$$

is equivalent to a $\forall\exists$-prenex sentence of the form

$$\exists\vec{x}_1\exists x_2, x_3\exists\vec{y}_1\exists y_2, y_3\forall\vec{x}\forall\vec{y}Q',$$

where x_2 and x_3 (y_2 and y_3) are new variables not in $\vec{x}_1$ ($\vec{y}_1$).

The proof of (2) is similar. ■

In other words, to prove that a sentence of the form $\exists\vec{x}_1\exists\vec{y}_1\forall\vec{x}_2\forall\vec{y}_2 Q$ is a sufficient condition for the uniqueness of Nash equilibria, it suffices to verify that this is the case for all games of sizes up to $(|\vec{x}_1| + 2) \times (|\vec{y}_1| + 2)$, and to prove that it is a sufficient condition for the non-existence of Nash equilibria, it suffices to verify this for games of sizes up to $(|\vec{x}_1| + 1) \times (|\vec{y}_1| + 1)$.

Theorem 1 holds for many specialized games as well. For instance, it holds for *strict games* as well. A game is strict if for both players, different profiles have different payoffs, that is, $(a, b) = (a', b')$ whenever $(a, b) \leq_i (a', b')$ and $(a', b') \leq_i (a, b)$, where $i = 1, 2$.

Theorem 2 *Theorem 1 holds when the following axioms are added to Σ:*

$$(x_1, y_1) \simeq_1 (x_2, y_2) \supset (x_1 = x_2 \wedge y_1 = y_2),$$
$$(x_1, y_1) \simeq_2 (x_2, y_2) \supset (x_1 = x_2 \wedge y_1 = y_2).$$

In fact, as can be seen from its proof, Theorem 1 holds when Σ is replaced by any set of $\forall$-prenex sentences.

Based on these theorems, Lin and Tang [2007] conducted some computer experimentations. Among others, their program re-discovered Kats and Thisse's class of weakly unilaterally competitive games which correspond to the following condition:

$$(x_1, y) \leq_1 (x_2, y) \supset (x_2, y) \leq_2 (x_1, y) \wedge (x, y_1) \leq_2 (x, y_2) \supset (x, y_2) \leq_1 (x, y_1).$$

Discovering State Invariants in Planning Domains

We now turn to the problem of discovering state constraints and state invariants in planning. In planning, state constraints are conditions that are true in all legal situations, i.e.

all legal initial situations as well as their future situations. State constraints are useful in planning for pruning search spaces, and several systems have been designed specifically for learning state constraints (c.f. [Huang *et al.*, 2000; Gerevini and Schubert, 1998]).

On the other hand, state invariants [Lin, 2004] are those that if true in a situation will be true in all successor situations. It is clear that a state constraint may not be a state invariant, and vice versa. However, many state constraints are invariants that are true in all legal initial situations. Thus given a planning domain with some example initial situations, the system in [Lin, 2004] tries to discover state invariants that are true in all the given initial situations. This system is based on a theorem similar to Proposition 3 that identifies a class of sentences that are finitely-verifiable.

A planning domain has a set of predicates representing fluents and other relations. It can also have some non-action functions such as $color(x)$ (the color of block x). We call this set of predicates and functions *domain language*. To formalize the effects of actions, we extend the domain language with a special sort $action$ for representing actions. Thus actions in the planning domain are represented by functions whose values are of $action$ sort. For each predicate in the domain language we also introduce a new predicate with the same name but with an extra argument of sort $action$. We call these new predicates "successor state predicates". For instance, if $clear$ is a fluent, then we write $clear(x)$ to mean that block x is clear in the given initial situation, and write $clear(x, unstack(x, y))$ to mean that x is clear in the successor situation of doing $unstack(x, y)$ in the initial state. We also assume a special unary predicate $Poss$ to denote the action precondition. Thus $Poss(unstack(x, y))$ will stand for the precondition of doing $unstack(x, y)$.

Definition 1 *An action theory is a family of first-order theories $\{T_A \mid A$ is an action type$\}$, where for each action type A, T_A consists of the following axioms:*

- *An action precondition axiom of the form*
$$\forall \vec{x}.Poss(A(\vec{x})) \equiv \Psi, \tag{9}$$
where Ψ is a formula in the domain language whose free variables are in $\vec{x}$. (Thus Ψ cannot mention $Poss$ and any successor state predicates.)
- *For each domain predicate F an axiom of the following form:*
$$(\forall \vec{x}, \vec{y}).F(\vec{x}, A(\vec{y})) \equiv \Phi_F(\vec{x}, \vec{y}), \tag{10}$$
where $\vec{x}$ and $\vec{y}$ do not share common variables, and Φ_F is a formula in the domain language whose free variables are from $\vec{x}$ and $\vec{y}$.

Example 1 In the blocks world, for the action type $stack$, we have the following axioms (all free variables below are universally quantified from outside):

$$Poss(stack(x, y)) \equiv holding(x) \wedge clear(y),$$
$$on(x, y, stack(u, v)) \equiv (x = u \wedge y = v) \vee on(x, y),$$
$$ontable(x, stack(u, v)) \equiv ontable(x),$$
$$handempty(stack(u, v)) \equiv true,$$
$$holding(x, stack(u, v)) \equiv false,$$
$$clear(x, stack(u, v)) \equiv clear(x) \wedge x \neq v.$$

The following definition captures the intuition that a state invariant is a formula that if true initially, will continue to be true after the successful completion of every possible action.

Definition 2 *Given an action theory $\{T_A \mid A$ is an action type$\}$, a formula W in the domain language is a state invariant if for each action type A,*

$$T_A \models \forall \vec{y}.W \wedge Poss(A(\vec{y})) \supset W(A(\vec{y})), \tag{11}$$

where $W(A(\vec{y}))$ is the result of replacing each atom $F(\vec{t})$ in W by $F(\vec{t}, A(\vec{y}))$, and $\models$ is the logical entailment in first-order logic.

The following theorem reduces the problem of checking whether a formula is a state invariant to that of the logical validity checking of a formula in the domain language.

Theorem 3 *Let W' be a state invariant. For any formula W in the domain language, $W \wedge W'$ is a state invariant iff for each action type A, the sentence*

$$\forall \vec{y}.W \wedge W' \wedge \Psi(\vec{y}) \supset \mathcal{R}(W, A(\vec{y})) \tag{12}$$

is valid, where Ψ is the action precondition of A as in the right side of (9), and $\mathcal{R}(W, A(\vec{y}))$, the regression of W over $A(\vec{y})$, is the result of replacing each atom $F(\vec{t})$ in W by $\Phi_F(\vec{t}, \vec{y})$ in the right of the axiom (10).

Lin [2004] showed that for so-called *simple* action theories, by Theorem 3, checking whether a conjunction of $\forall$-prenex formulas $\forall \vec{x} B$ is a state invariant can be reduced to checking the validity of a $\forall\exists$-prenex sentence, thus Proposition 3 will apply.

Definition 3 *An action theory is said to be* simple *if for each action type A:*

- *The formula Ψ in its action precondition axiom (9) has no quantifiers.*
- *The formula Φ_F in the axiom (10) for each F has no quantifiers, and (10) entails the following formula:*
$$(\forall \vec{x}, \vec{y}).\neg subset(\vec{x}, \vec{y}) \supset F(\vec{x}, A(\vec{y})) \equiv F(\vec{x}),$$
where $subset(\vec{x}, \vec{y})$, meaning $\vec{x}$ is a subset of $\vec{y}$, is the following formula:
$$\bigwedge_{x \in \vec{x}} \bigvee_{y \in \vec{y}} x = y.$$

Notice that for context-free action domains such as the blocks world and the logistics domain, successor state axioms (10) have the form:

$$F(\vec{x}, A(\vec{y})) \equiv E_1 \vee \cdots \vee E_n \vee (F(\vec{x}) \wedge \neg E_{n+1} \wedge \cdots \wedge \neg E_m),$$

where E_i's are conjunctions of equality atoms between variables in $\vec{x}$ and $\vec{y}$. For instance, in the blocks world, we have:

$$clear(x, unstack(y, z)) \equiv$$
$$\quad (x = z) \vee (clear(x) \wedge y \neq x),$$
$$on(x_1, x_2, unstack(y, z)) \equiv$$
$$\quad on(x_1, x_2) \wedge \neg(x_1 = y \wedge x_2 = z).$$

Thus context-free action domains are simple according to our definition as long as action preconditions do not mention any quantifiers.

$\forall$-prenex sentences $\forall \vec{x} B$ include many typical state constraints found in planning, such as the functional dependency constraints like

$$on(x, y) \wedge on(x, z) \supset y = z,$$

exclusiveness conditions like

$$(\exists y) on(x, y) \supset \neg ontable(x),$$

and information about types like

$$at(x, y) \wedge airplane(x) \supset airport(y).$$

Lin [2004] described a system based on these results, and reported that for the blocks world the system discovered the following invariants (more precisely the conjunction of the following sentences):

$$\neg handempty \vee \neg(\exists x) holding(x),$$
$$\neg clear(x) \vee \neg(\exists z) on(z, x),$$
$$\neg holding(x) \vee \neg clear(x),$$
$$\neg holding(x) \vee \neg(\exists y) on(x, y),$$
$$\neg holding(x) \vee \neg(\exists z) on(z, x),$$
$$\neg ontable(x) \vee \neg holding(x),$$
$$\neg ontable(x) \vee \neg(\exists y) on(x, y),$$
$$holding(x_1) \wedge holding(x_2) \supset x_1 = x_2,$$
$$on(x_1, x_2) \wedge on(x_1, x_3) \supset x_2 = x_3,$$
$$on(x_2, x_1) \wedge on(x_3, x_1) \supset x_2 = x_3$$
$$holding(x) \vee clear(x) \vee (\exists z) on(z, x),$$
$$ontable(x) \vee holding(x) \vee (\exists y) on(x, y),$$
$$handempty \vee (\exists x) holding(x)$$

As one can see, these include many familiar state constraints in the blocks world. However they are not complete in the sense that there are some illegal states that satisfy all of them. For instance, if B is the only block in the domain, then the state $\{on(B, B), handempty\}$ is apparently illegal but satisfies all the sentences in the above two sets. However, if we add the following sentences:

$$above(x, y) \equiv (on(x, y) \vee (\exists z)(on(x, z) \wedge above(z, y))),$$
$$above(x, y) \supset \neg on(y, x),$$

then we get a complete set of state invariants. It is not clear how sentences like these can be discovered as they involve new predicates, and are essentially second-order.

For the logistics domain, the system returns the following state invariants:

$$inCity(x1, x2) \wedge inCity(x1, x3) \supset x2 = x3,$$
$$at(x2, x1) \wedge airplane(x2) \supset airport(x1),$$
$$\neg(\exists y) in(x, y) \vee \neg(\exists y) at(x, y),$$
$$in(x_1, x_2) \wedge in(x_1, x_3) \supset x_2 = x_3,$$
$$package(x) \supset [(\exists y) in(x, y) \vee (\exists y) at(x, y)],$$
$$vehicle(x) \supset (\exists y) at(x, y),$$
$$(\exists y) inCity(x, y)$$

These constraints turn out to be complete for the logistics domain in the sense that a state is "legal" if it satisfies all the constraints.

Conclusion Remarks

We have proposed a notion of finitely-verifiable classes of sentences. The motivation is that if a class of sentences is finitely-verifiable, then checking whether a sentence in this class is a theorem can be done by model-checking in a finite set of models, thus paving the way for using computers to discover theorems in this class. Identifying finitely-verifiable classes of sentences is a domain-dependent task, and the main theoretical challenge for computer-aided theorem discovery.

References

H. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 1972.

J. Friedman. On characterizing equilibrium points in two-person strictly competitive games. *International Journal of Game Theory*, 12:245 – 247, 1983.

A. Gerevini and L. Schubert. Inferring state constraints for domain-independent planning. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI–98), AAAI Press, Menlo Park, CA.*, pages 905–912, 1998.

Y. Huang, B. Selman, and H. A. Kautz. Learning declarative control rules for constraint-based planning. In *ICML'2000*, pages 415–422, 2000.

A. Kats and J. Thisse. Unilaterally competitive games. *International Journal of Game Theory*, 21:291 – 299, 1992.

F. Lin and Y. Chen. Discovering classes of strongly equivalent logic programs. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI–05)*, pages 516–521, 2005.

F. Lin and P. Tang. Discovering theorems in game theory: Two-person games with unique Nash equilibria. http://www.cs.ust.hk/faculty/flin/papers/zerosum.pdf, 2007.

F. Lin. Compiling causal theories to successor state axioms and STRIPS-like systems. *Journal of Artificial Intelligence Research*, 19:279–314, 2003.

F. Lin. Discovering state invariants. In *Proceedings of the Nineth International Conference on Principles of Knowledge Representation and Reasoning (KR2004)*, pages 536–544, 2004.

D. Monderer and L. S. Shapley. Potential games. *Games and Economic Behavior*, 14:124 – 143, 1996.

H. Moulin. Cooperation in mixed equilibrium. *Mathematics of Operations Research*, 1:273 – 286, 1976.

M. Petkovsek, H. S. Wilf, and D. Zeilberger. *A = B*. Wellesley, Mass. : A K Peters, 1996.

D. Topkis. *Supermodularity and Complementarity*. Princeton University Press, New Jersey, 1998.