Tractable Reasoning with Incomplete First-Order Knowledge in Dynamic Systems with Context-Dependent Actions

Yongmei Liu and **Hector J. Levesque**
Department of Computer Science
University of Toronto
Toronto, ON, Canada M5S 3G4
{yliu, hector}@cs.toronto.edu

Abstract

A basic reasoning problem in dynamic systems is the projection problem: determine if a formula holds after a sequence of actions has been performed. In this paper, we propose a tractable[1] solution to the projection problem in the presence of incomplete first-order knowledge and context-dependent actions. Our solution is based on a type of progression, that is, we progress the initial knowledge base (KB) wrt the action sequence and answer the query against the resulting KB. The form of reasoning we propose is always logically sound and is also logically complete when the query is in a certain normal form and the agent has complete knowledge about the context of any context-dependent actions.

1 Introduction

In the area of Knowledge Representation and Reasoning, there is a well-known tradeoff between the expressiveness of the representation language and the computational tractability of the associated reasoning task. At one extreme, we have databases for which queries can be efficiently evaluated. But databases are too limited for many AI applications because they require complete knowledge about the domains. Levesque [1998] proposes a generalization of a database called a *proper KB*, which allows a limited form of incomplete knowledge, equivalent to a (possibly infinite) consistent set of ground literals. Since the deduction problem for proper KBs is undecidable, Levesque proposes an evaluation-based reasoning procedure called V that is logically sound and, when the query is in a certain *normal form* called $\mathcal{NF}$, also logically complete. Moreover, later Liu and Levesque [2003] show that despite the incomplete knowledge, database techniques can be used to implement V efficiently.

In this paper, we apply the procedure V to reasoning in *dynamic systems* where the state of the world changes as a result of the actions of agents. For such applications, a basic reasoning problem is the so-called *projection problem*: given an action theory that specifies the preconditions and effects of actions, and an initial KB, determine whether or not a formula holds after a sequence of actions is performed. Two settings where this problem arises naturally are for planning and for high-level program execution [Levesque *et al.*, 1997]. A prerequisite to planning is the ability to determine if a goal is satisfied after a sequence of actions. To execute a high level robotic program such as "while there is a block on the table, pick up a block and put it away", one needs to determine after various sequences of actions whether there is still a block on the table.

In practice, there are two ways to deal with projection: we can *progress* the initial KB wrt the action sequence and answer the query against the resulting KB; or we can *regress* the query wrt the action sequence and answer the resulting query against the initial KB. Progression has at least two advantages: First, it avoids a duplication of effort when multiple queries need to be answered wrt the same action sequence, and especially when that sequence is long. Second, in a robotics setting, a robot can use its "mental idle time" to compute a progression while it is busy performing physical actions. Projection via progression has three main computational requirements: the new KB must be efficiently computed, its size should be at most linear in the size of the initial KB (to allow for iterated progression), and it must be possible to answer the query efficiently from the new KB. Lin and Reiter [1997] give a formal study of progression. They show that progression is not always first-order definable, and identify a few important cases where progression is first-order definable and computationally tractable. However, the third requirement is not addressed in their paper.

In this paper, we propose a tractable, sound, and sometimes complete solution to the projection problem in the presence of incomplete first-order knowledge and context-dependent actions. We restrict our attention to actions with only "local" effects, and where incomplete knowledge is in the form of a proper KB. We define a version of progression where a proper KB remains proper afterward, and where applying V to the progressed KB and the query returns the same value as applying V to the initial KB and a regressed query. We prove that when the query is in $\mathcal{NF}$ and the initial KB has complete knowledge about the context of any context-dependent actions, our solution is logically complete. It is also logically complete when the query is in $\mathcal{NF}$ and there are sensing actions that provide dynamic information about the context of the relevant context-dependent actions.

[1] By "tractable" we mean "solvable in polynomial time".

2 Preliminaries

In this section, we review proper KBs, V, and $\mathcal{NF}$. Also, we briefly review the situation calculus, and formally define local-effect action theories and regression for them.

2.1 Proper KBs, V, and the normal form $\mathcal{NF}$

We use a standard first-order logical language $\mathcal{L}$ with equality, a countably infinite set of constants $\mathcal{C} = \{c_1, c_2, \ldots\}$, and no other function symbols. We restrict our attention to *standard interpretations*, where equality is identity, and there is a bijection between the set of constants and the domain of discourse. This restriction can be captured by a set of axioms $\mathcal{E}$, consisting of the axioms of equality and the set of formulas $\{c_i \neq c_j \mid i \neq j\}$. Since we treat equality separately, when we say "predicate", "atom" or "literal", we exclude equality.

We use ρ to range over atoms whose arguments are distinct variables. We use e to range over *ewffs*, that is, quantifier-free formulas with only equalities (*i.e.* no predicates). We use $\forall\phi$ to denote the universal closure of ϕ. We write ϕ^x_c to denote ϕ with all free occurrences of x replaced by constant c. We write $\Sigma \models_{\mathcal{E}} \phi$ to denote $\mathcal{E} \cup \Sigma \models \phi$.

Definition 1 A KB Σ is *proper* if $\mathcal{E} \cup \Sigma$ is consistent and Σ is a finite set of formulas of the form $\forall(e \supset \rho)$ or $\forall(e \supset \neg\rho)$.

It is not hard to see that the problem of determining whether a sentence is logically entailed by a proper KB is undecidable, since when the KB is empty, this reduces to classical validity. Levesque [1998] proposes an evaluation-based reasoning procedure called V instead. Given a proper KB and a query, V returns one of three values 0 (known false), 1 (known true), or $\frac{1}{2}$ (unknown) as follows:

1. $V[\Sigma, P(\vec{c})] = \begin{cases} 1 & \text{if there is a } \forall(e(\vec{x}) \supset P(\vec{x})) \\ & \quad \text{in } \Sigma \text{ such that } V[\Sigma, e(\vec{c})] = 1 \\ 0 & \text{if there is a } \forall(e(\vec{x}) \supset \neg P(\vec{x})) \\ & \quad \text{in } \Sigma \text{ such that } V[\Sigma, e(\vec{c})] = 1 \\ \frac{1}{2} & \text{otherwise} \end{cases}$
2. $V[\Sigma, c = c'] = 1$ if c is identical to c', and 0 otherwise;
3. $V[\Sigma, \neg\phi] = 1 - V[\Sigma, \phi]$;
4. $V[\Sigma, \phi \vee \psi] = \max\{V[\Sigma, \phi], V[\Sigma, \psi]\}$;
5. $V[\Sigma, \exists x\phi] = \max_{c \in H^+} V[\Sigma, \phi^x_c]$, where H^+ is the union of the constants in Σ or ϕ, and an extra constant.

This V procedure is logically sound and, when the query is in a certain normal form called $\mathcal{NF}$, also logically complete:

Theorem 1 ([Levesque, 1998]) *Let Σ be proper. Then*

1. *for every $\phi \in \mathcal{L}$, if $V[\Sigma, \phi] = 1$ then $\Sigma \models_{\mathcal{E}} \phi$; and if $V[\Sigma, \phi] = 0$ then $\Sigma \models_{\mathcal{E}} \neg\phi$.*
2. *for every $\phi \in \mathcal{NF}$, $V[\Sigma, \phi] = 1$ iff $\Sigma \models_{\mathcal{E}} \phi$; and $V[\Sigma, \phi] = 0$ iff $\Sigma \models_{\mathcal{E}} \neg\phi$.*

For the interested readers, the following is the definition of $\mathcal{NF}$ from [Levesque, 1998]:

Definition 2 A set Γ of sentences is logically separable iff for every consistent set of ground literals L, if $L \cup \Gamma$ has no standard model, then $L \cup \{\phi\}$ is inconsistent for some $\phi \in \Gamma$.

Definition 3 The normal form $\mathcal{NF}$ is the least set such that

1. if ϕ is a ground atom or ewff, then $\phi \in \mathcal{NF}$;
2. if $\phi \in \mathcal{NF}$, then $\neg\phi \in \mathcal{NF}$;
3. if $\phi_1, \ldots, \phi_n \in \mathcal{NF}$, and $\{\phi_1, \ldots, \phi_n\}$ is logically separable, then $\wedge\phi_i \in \mathcal{NF}$;
4. if $\Gamma \subseteq \mathcal{NF}$, Γ is logically separable, and for some ϕ, $\Gamma = \{\phi^x_c \mid c \in \mathcal{C}\}$, then $\forall x\phi \in \mathcal{NF}$.

The intuition behind $\mathcal{NF}$ is that different parts of a formula must be logically independent. A simple example of a formula not in $\mathcal{NF}$ is $(p \vee \neg p)$, where p is atomic. In the propositional case, a CNF formula is in $\mathcal{NF}$ if its clauses are non-tautologous and closed under resolution.

Liu and Levesque [2003] show that V can be implemented efficiently using database techniques (projections, joins, *etc*). Here we present a cleaner variant of this result.

Let $\mathcal{L}^k$ denote the set of formulas from $\mathcal{L}$ that use at most k different variables. Let $R = \{\vec{c}_1, \ldots, \vec{c}_m\}$ be a finite set of n-tuples. We use $\vec{x} \in R$ to denote $\vec{x} = \vec{c}_1 \vee \ldots \vee \vec{x} = \vec{c}_m$.

Definition 4 Let L be P or $\neg P$ for some predicate P. The *ewff defining L* in a proper Σ, denoted by ξ_L, is the disjunction of all e such that $\forall(e(\vec{x}) \supset L(\vec{x})) \in \Sigma$. We can write ξ_L in the form of $\vec{x} \in I_L \vee e_L \wedge \vec{x} \notin O_L$ so that I_L and O_L are finite relations with as many tuples as possible, and e_L is an ewff. The *e-size* of Σ is the maximum size of an e_L in Σ.

Then a corollary to Theorem 4.8 in [Liu and Levesque, 2003]:

Corollary 2 *Let Σ be proper, and let $\phi \in \mathcal{L}^k$. Then $V[\Sigma, \phi]$ can be computed in time $O(lmn^k)$, where l is the size of ϕ, m is the e-size of Σ, and n is the size of Σ.*

Although the time complexity scales exponentially with k, this is typical even of queries over ordinary databases, and so is perhaps as good as can be expected.

2.2 Situation calculus

Our account of action and change is formulated in the language of the situation calculus [McCarthy and Hayes, 1969; Reiter, 2001]. We will not go over the language here except to note the following components: there are three disjoint sorts for actions, situations, and objects; there is a special constant S_0 denoting the *initial situation*, namely the one in which no actions have yet occurred; there is a distinguished binary function $do(a, s)$ denoting the successor situation to s resulting from performing action a; relations whose truth values vary from situation to situation, are called (relational) *fluents*, and are denoted by predicate symbols taking a situation term as their last argument; and there is a special predicate $Poss(a, s)$ stating that action a is executable in situation s.

We relate the language of the situation calculus to $\mathcal{L}$ as follows: There is a set of constants of sort object which are constants of $\mathcal{L}$. The situation-independent predicates and relational fluents are predicates from $\mathcal{L}$. That is, if $P(\vec{x})$ is a situation-independent predicate, and $F(\vec{x}, s)$ is a relational fluent, then $P(\vec{x})$ and $F(\vec{x})$ are predicates from $\mathcal{L}$.

We extend the language $\mathcal{L}$ to $\mathcal{L}^+$ by allowing equalities involving action functions. Let $\phi \in \mathcal{L}^+$, and let τ be a situation term. We use $\phi[\tau]$ to denote the situation calculus formula obtained from ϕ by taking τ as the situation arguments of all fluents mentioned by ϕ. We use α to range over ground actions, and we use δ to range over sequences of ground actions.

Let $\delta = \langle \alpha_1, \ldots, \alpha_n \rangle$. We use $do(\delta, S_0)$ to denote the end situation of δ, that is, $do(\alpha_n, do(\alpha_{n-1}, \ldots do(\alpha_1, S_0) \ldots))$.

A particular domain of application will be specified by a basic action theory of the following form:[2]

$$\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_{S_0}, \text{ where}$$

1. $\mathcal{D}_{ap}$ is a set of action precondition axioms, one for each action function A, with form $Poss(A(\vec{x}), s) \equiv \Pi_A(\vec{x})[s]$,[3] where $\Pi_A(\vec{x}) \in \mathcal{L}$.
2. $\mathcal{D}_{ss}$ is a set of successor state axioms (SSAs), one for each fluent, of the form $F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a)[s]$, where $\Phi_F(\vec{x}, a) \in \mathcal{L}^+$. Usually, $\Phi_F(\vec{x}, a)$ has the form
$$\gamma_F^+(\vec{x}, a) \vee (F(\vec{x}) \wedge \neg\gamma_F^-(\vec{x}, a)).$$
SSAs take the place of the so-called effect axioms, and provide a solution to the frame problem.
3. $\mathcal{D}_{una}$ is the set of unique names axioms for actions:
$$A(\vec{x}) \neq A'(\vec{y}), \text{ and } A(\vec{x}) = A(\vec{y}) \supset \vec{x} = \vec{y},$$
where A and A' are distinct action functions.
4. $\mathcal{D}_{S_0}$ is of the form $\{\phi[S_0] \mid \phi \in \Sigma_0\}$, where $\Sigma_0 \subseteq \mathcal{L}$. Σ_0 is called the initial KB.

In this setting, the projection task can be defined as follows: determine if $\mathcal{D} \models_{\mathcal{E}} \phi[do(\delta, S_0)]$, where $\phi \in \mathcal{L}$, and δ is a sequence of ground actions.

As a running example, we will use a simple blocks world.[4] We use a single action, $move(x, y, z)$, moving a block x from block y to block z (treating the table as just another block). We use three fluents: $clear(x, s)$, block x has no blocks on top of it; $on(x, y, s)$, block x is on block y; $eh(x, s)$, the height of block x is even. We have the following action precondition axiom and successor state axioms:

$$Poss(move(x, y, z), s) \equiv clear(x) \wedge on(x, y) \wedge clear(z).$$
$$clear(x, do(a, s)) \equiv (\exists y, z)a = move(y, x, z) \vee clear(x, s) \wedge \neg(\exists y, z)a = move(y, z, x);$$
$$on(x, y, do(a, s)) \equiv (\exists z)a = move(x, z, y) \vee on(x, y, s) \wedge \neg(\exists z)a = move(x, y, z);$$
$$eh(x, do(a, s)) \equiv (\exists y, z)[a = move(x, y, z) \wedge \neg eh(z, s)] \vee eh(x, s) \wedge \neg(\exists y, z)[a = move(x, y, z) \wedge eh(z, s)].$$

2.3 Local effect action theories and regression

Actions in many dynamic domains have only *local effects* in the sense that if an action $A(\vec{c})$ changes the truth value of an atom $F(\vec{d})$, then $\vec{d}$ is contained in $\vec{c}$. This contrasts with actions having *universal effects* such as exploding a bomb, which kills all those near it. We can define this as follows:

Definition 5 A successor state axiom is *local-effect* if both $\gamma_F^+(\vec{x}, a)$ and $\gamma_F^-(\vec{x}, a)$ are disjunctions of formulas of the form $\exists \vec{z}[a = A(\vec{y}) \wedge \phi(\vec{y})]$, where A is an action function, $\vec{y}$ contains $\vec{x}$, $\vec{z}$ is the remaining variables of $\vec{y}$, and ϕ (called a *context formula*) is a quantifier-free formula from $\mathcal{L}$. An action theory is local-effect if each SSA is local-effect.

[2] We use slightly different notation from that in [Reiter, 2001].

[3] We omit the leading universal quantifiers.

[4] To justify the concerns for the tractability of reasoning, the reader should imagine there being a very large number of blocks.

Our blocks world example above is clearly local-effect.

The notion of a successor state axiom being local-effect is a generalization of that of being strictly context-free defined by Lin and Reiter [1997]. An SSA is *strictly context-free* if $\gamma_F^+(\vec{x}, a)$ and $\gamma_F^-(\vec{x}, a)$ are disjunctions of formulas of the form $\exists \vec{z}[a = A(\vec{y})]$, where A, $\vec{y}$, and $\vec{z}$ are as above. For instance, the SSA for fluent on is strictly context-free, while that for fluent eh is not.

By using the unique names axioms, the instantiation of a local-effect SSA on a ground action can be significantly simplified. Suppose the SSA for F is local-effect. Let $\alpha = A(\vec{c})$ be a ground action, and let $*$ be $+$ or $-$. Then $\gamma_F^*(\vec{x}, \alpha)$ is equivalent to a formula of the following form:

$$\vec{x} = \vec{d}_1 \wedge \psi_1 \vee \ldots \vee \vec{x} = \vec{d}_n \wedge \psi_n,$$

where $\vec{d}_i$ is a vector of constants contained in $\vec{c}$, and ψ_i is a sentence. We will use $\gamma_F^*(\alpha)(\vec{x})$ to denote the above formula, and we will write $(\vec{d}, \psi) \in \gamma_F^*(\alpha)$ to mean that $\vec{x} = \vec{d} \wedge \psi$ is one of the disjuncts. Also, we will use $\Phi_F(\alpha)(\vec{x})$ to denote $\gamma_F^+(\alpha)(\vec{x}) \vee (F(\vec{x}) \wedge \neg\gamma_F^-(\alpha)(\vec{x}))$. In the case of our blocks world, instances of the SSAs can be simplified as follows:

$$clear(x, do(move(c_1, c_2, c_3), s)) \equiv x = c_2 \vee clear(x, s) \wedge \neg(x = c_3).$$
$$on(x, y, do(move(c_1, c_2, c_3), s)) \equiv x = c_1 \wedge y = c_3 \vee on(x, y, s) \wedge \neg(x = c_1 \wedge y = c_2).$$
$$eh(x, do(move(c_1, c_2, c_3), s)) \equiv x = c_1 \wedge \neg eh(c_3, s) \vee eh(x, s) \wedge \neg(x = c_1 \wedge eh(c_3, s)).$$

An important computational mechanism for reasoning about actions is regression [Reiter, 2001]. Here we define a one-step regression operator for local-effect action theories.

Definition 6 Let $\phi \in \mathcal{L}$. We use $\mathcal{R}_\alpha(\phi)$ to denote the formula obtained from ϕ by replacing each fluent atom $F(\vec{t})$ with $\Phi_F(\alpha)(\vec{t})$. We call $\mathcal{R}_\alpha(\phi)$ the *regression* of ϕ wrt α.

Note that $\mathcal{R}_\alpha(\phi)$ remains in $\mathcal{L}$. Let $\delta = \langle \alpha_1, \ldots, \alpha_n \rangle$. We use $\mathcal{R}_\delta$ to denote $\mathcal{R}_{\alpha_1} \circ \cdots \circ \mathcal{R}_{\alpha_n}$. We now state a simple form of the regression theorem [Reiter, 2001]. Recall that Σ_0 is the initial KB of $\mathcal{D}$.

Theorem 3 (The Regression Theorem)
For every $\phi \in \mathcal{L}$, $\mathcal{D} \models_{\mathcal{E}} \phi[do(\delta, S_0)]$ *iff* $\Sigma_0 \models_{\mathcal{E}} \mathcal{R}_\delta(\phi)$.

This theorem shows that regression is a sound and complete solution to the projection problem. In this paper, we prove all our results about progression by using regression as a bridge.

3 Progression of Proper KBs

In this section, we define a variant of classical progression, and show how to compute it for local-effect action theories.

First consider *classical progression*. Suppose we have a KB Σ. Let M be a possible state of Σ, that is, a model of Σ. Let α be a ground action. Then the *successor state* of M wrt α is the model M' such that for any ground fluent atom $F(\vec{c})$, $M' \models F(\vec{c})$ iff $M \models \Phi_F(\alpha)(\vec{c})$. A KB Σ' is a progression of Σ wrt α if the models of Σ' are exactly the successor states of models of Σ wrt α. A basic property of progression is: Suppose that Σ' is a progression of Σ wrt α. Then for every $\phi \in \mathcal{L}$, $\Sigma' \models_{\mathcal{E}} \phi$ iff $\Sigma \models_{\mathcal{E}} \mathcal{R}_\alpha(\phi)$. It is in this sense that we say classical progression preserves classical entailment.

It would be nice if the classical progression of a proper KB were proper, so that we could use it and V to solve the projection problem. However, this is unfortunately not the case even for very simple action theories. Consider the following example from [Petrick and Levesque, 2002]:

$$F(do(a,s)) \equiv a = A \wedge G(s) \vee F(s); \;\; G(do(a,s)) \equiv G(s).$$

Then any progression of the empty KB (which is proper) wrt action A results in disjunctive information, $(F \vee \neg G)$, and hence is no longer proper. So what we will propose is a variant of classical progression where the progression of a proper KB does remain proper and the progression preserves V instead of preserving classical entailment.

Definition 7 Let Σ and Σ' be proper. We say that Σ' is a *(weak) progression* of Σ wrt a ground action α if for every $\phi \in \mathcal{L}$, $V[\Sigma', \phi] = V[\Sigma, \mathcal{R}_\alpha(\phi)]$.

We now show that for local-effect action theories, it is easy to compute a weak progression of a proper KB.

Definition 8 Let $\mathcal{D}$ be local-effect and Σ be proper. We define $\mathcal{P}_\alpha(\Sigma)$ as the set of the following sentences:

$$\forall [\vec{x} \in A_F \vee \xi_F(\vec{x}) \wedge \vec{x} \notin D_F \supset F(\vec{x})],$$
$$\forall [\vec{x} \in (A_{\neg F} - D_{\neg F}) \vee \xi_{\neg F}(\vec{x}) \wedge \vec{x} \notin D_{\neg F} \supset \neg F(\vec{x})],$$

where F ranges over fluents, ξ_F (resp. $\xi_{\neg F}$) is the ewff defining F (resp. $\neg F$) in Σ (c.f. Definition 4), and

1. $A_F = \{\vec{d} \mid (\vec{d}, \psi) \in \gamma^+_F(\alpha)$ and $V[\Sigma, \psi] = 1\}$,
   $D_{\neg F} = \{\vec{d} \mid (\vec{d}, \psi) \in \gamma^+_F(\alpha)$ and $V[\Sigma, \psi] \neq 0\}$;
2. $A_{\neg F} = \{\vec{d} \mid (\vec{d}, \psi) \in \gamma^-_F(\alpha)$ and $V[\Sigma, \psi] = 1\}$,
   $D_F = \{\vec{d} \mid (\vec{d}, \psi) \in \gamma^-_F(\alpha)$ and $V[\Sigma, \psi] \neq 0\}$.

Then we get the following:

Theorem 4 *Let $\mathcal{D}$ be local-effect and Σ be proper. Then $\mathcal{P}_\alpha(\Sigma)$ is a weak progression of Σ wrt α.*

Let $\delta = \langle \alpha_1, \ldots, \alpha_n \rangle$. We use $\mathcal{P}_\delta$ to denote $\mathcal{P}_{\alpha_n} \circ \cdots \circ \mathcal{P}_{\alpha_1}$. By a simple induction, we have that for every $\phi \in \mathcal{L}$, $V[\mathcal{P}_\delta(\Sigma), \phi] = V[\Sigma, \mathcal{R}_\delta(\phi)]$.

The intuition behind A_F and $D_{\neg F}$ is simple. For $\vec{d} \in A_F$, $F(\vec{d})$ will become true in every possible successor state, so we add $F(\vec{d})$ to Σ. For $\vec{d} \in D_{\neg F}$, $F(\vec{d})$ may become true in some possible successor state, so we delete $\neg F(\vec{d})$ from Σ. Now consider our blocks world example. Let $\Sigma = \{on(c_1, c_2), clear(c_1), clear(c_3), eh(c_1)\}$. After action $move(c_1, c_2, c_3)$ is performed, we add $clear(c_2)$, $\neg clear(c_3)$, $on(c_1, c_3)$, and $\neg on(c_1, c_2)$ to Σ, and delete $clear(c_3)$, $\neg clear(c_2)$, $on(c_1, c_2)$, $\neg on(c_1, c_3)$, $eh(c_1)$, and $\neg eh(c_1)$ from Σ. We delete $eh(c_1)$ because if $eh(c_3)$ holds in the current state, $eh(c_1)$ will become false in the successor state; similarly, we delete $\neg eh(c_1)$.

We now define a reasoning procedure PV to solve the projection task using weak progression and V as follows:

Definition 9 Let $\mathcal{D}$ be a local-effect action theory with a proper Σ_0. We define $PV[\delta, \phi]$ as $V[\mathcal{P}_\delta(\Sigma_0), \phi]$.

Now suppose that $PV[\delta, \phi] = 1$. Then $V[\Sigma_0, \mathcal{R}_\delta(\phi)] = V[\mathcal{P}_\delta(\Sigma_0), \phi] = 1$. By soundness of V, $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\delta(\phi)$. By the Regression Theorem, $\mathcal{D} \models_\mathcal{E} \phi[do(\delta, S_0)]$. Similarly, if $PV[\delta, \phi] = 0$, then $\mathcal{D} \models_\mathcal{E} \neg\phi[do(\delta, S_0)]$. Thus PV is logically sound for projection. It is easy to see that $\mathcal{P}_\alpha(\Sigma)$ can be computed in $O(n)$ time, where n is the size of Σ. By Corollary 2, we have the following tractability result:

Theorem 5 *Let $\mathcal{D}$ be a local-effect action theory with a proper Σ_0, and let $\phi \in \mathcal{L}^k$. Then $PV[\delta, \phi]$ can be computed in time $O(pn + lmn^k)$, where p is the length of δ, l the size of ϕ, m the e-size of Σ_0, and n the size of Σ_0.*

Thus PV provides an efficient and logically sound solution to the projection problem despite the incomplete knowledge. In the next two sections, we will explore under what conditions, PV is also logically complete.

4 A Completeness Result

Since PV uses V, it is not surprising that we need a query to be in normal form for logical completeness. In this section, we will show that the only other thing we need is for the initial KB to have complete knowledge of the context of any context-dependent actions.

More precisely, we say that a KB Σ is *complete* wrt a set G of ground atoms if for all $l \in G$, either $\Sigma \models_\mathcal{E} l$ or $\Sigma \models_\mathcal{E} \neg l$. A KB Σ is complete wrt a predicate P if it is complete wrt all ground atoms of P. Now let Σ be proper, and ϕ a quantifier-free sentence such that Σ is complete wrt all atoms of ϕ. Then it is easy to see that $V[\Sigma, \phi]$ is either 0 or 1.

Definition 10 A KB Σ is *context-complete* (wrt $\mathcal{D}$) if it is complete wrt every predicate appearing in every γ^+_F and γ^-_F.

So Σ is context-complete if it has complete knowledge about the predicates in the context of any context-dependent actions. For example, in our blocks world, a Σ is context-complete if it is complete wrt eh; it may be incomplete wrt $clear$ and on. So context-completeness still allows incomplete knowledge.

There are two useful special cases where we get context-completeness. An SSA is *equality-only* if no predicate appears in γ^+_F or γ^-_F. Obviously, any Σ is context-complete wrt equality-only SSAs. Indeed, many SSAs we come across are equality-only. An SSA is *context-free* if no fluent appears in γ^+_F or γ^-_F. It is reasonable to assume that an agent has complete knowledge about situation-independent predicates. Under such an assumption, any Σ is context-complete wrt context-free SSAs.

The logical completeness of PV is obtained by showing that progression preserves context-completeness and that under context-completeness, our progression coincides with classical progression.

Theorem 6 *Let Σ be context-complete. Then*
1. $\mathcal{P}_\alpha(\Sigma)$ is context-complete too;
2. $\mathcal{P}_\alpha(\Sigma)$ is a classical progression of Σ.

Proof: (2) We prove that for every model M', $M' \models \mathcal{P}_\alpha(\Sigma)$ iff there is a model M s.t. $M \models \Sigma$ and M' is the successor state of M wrt α. For the only-if direction, we construct M as follows: for every fluent atom $F(\vec{c})$, if $V[\Sigma, F(\vec{c})] = 1$, then $M \models F(\vec{c})$; if $V[\Sigma, F(\vec{c})] = 0$, then $M \models \neg F(\vec{c})$; otherwise, $M \models F(\vec{c})$ iff $M' \models F(\vec{c})$. The proof uses the fact that $V[\Sigma, \gamma^*_F(\alpha)(\vec{c})] \in \{0, 1\}$, where $*$ is $+$ or $-$. ■

So under context-completeness, our progression preserves classical entailment. Now let Σ_0 be context-complete. By a simple induction, we have: for every $\phi \in \mathcal{L}$, $\mathcal{P}_\delta(\Sigma_0) \models_\mathcal{E} \phi$ iff $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\delta(\phi)$. Now let $\phi \in \mathcal{NF}$. By completeness of V for $\mathcal{NF}$, $V[\mathcal{P}_\delta(\Sigma_0), \phi] = 1$ iff $\mathcal{P}_\delta(\Sigma_0) \models_\mathcal{E} \phi$ iff $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\delta(\phi)$ iff $\mathcal{D} \models_\mathcal{E} \phi[do(\delta, S_0)]$. Thus when the initial KB is context-complete and the query is in normal form, PV is logically complete for projection.

5 Incorporating Sensing

In many applications, it is asking too much to require complete knowledge in the initial KB about the context of the context-dependent actions. In this section, we follow de Giacomo and Levesque [1999] and relax this restriction in two ways: first, we only need context-completeness relative to the sequence of actions and the query in question; second, we can achieve this local context-completeness dynamically by resorting to sensing actions, that is, actions that get knowledge from outside the system. In other words, we show that when a history of actions and sensing results is "just-in-time" for a normal form query, PV is once again logically complete.

We first extend our account of action and change to incorporate sensing. Assume that in addition to ordinary actions that change the world, we also have binary sensing actions that do not change the world but tell the agent whether some condition ϕ holds in the current situation. We use the predicate $SF(a, s)$ to characterize what the sensing action tells the agent about the world. Now our basic action theory has an extra component $\mathcal{D}_{sf}$, which is a set of sensed fluent axioms (SFAs), one for each action, of the form $SF(A(\vec{x}), s) \equiv \phi_A(\vec{x})[s]$, where $\phi_A \in \mathcal{L}$. We say that $\mathcal{D}_{sf}$ is atomic if each ϕ_A is an atom.

For instance, we may add three sensing actions to the blocks world example: $sense_{clear}(x)$, $sense_{on}(x, y)$, and $sense_{eh}(x)$. The axiom $SF(sense_{eh}(x), s) \equiv eh(x, s)$ says that the action $sense_{eh}(x)$ tells the agent if $eh(x, s)$ holds.

To describe a sequence of actions and sensing results, we use the notion of a *history*, that is, a sequence of pairs (α, μ) where α is a ground action and $\mu \in \{0, 1\}$ is the sensing result: when α is an ordinary action, we simply let $\mu = 1$. We use $end(\sigma)$ to denote the end situation of history σ, and $Sensed(\sigma)$ to denote the situation calculus formula stating all sensing results of σ. Formally,

- $end(\varepsilon) = S_0$, where ε is the empty history;
  $end(\sigma \cdot (\alpha, \mu)) = do(\alpha, end(\sigma))$.
- $Sensed(\varepsilon) = True$;
  $Sensed(\sigma \cdot (\alpha, 1)) = Sensed(\sigma) \wedge SF(\alpha, end(\sigma))$;
  $Sensed(\sigma \cdot (\alpha, 0)) = Sensed(\sigma) \wedge \neg SF(\alpha, end(\sigma))$.

Naturally, we are only interested in consistent histories, that is, histories with reasonable sensing results. Formally,

Definition 11 A history σ is *consistent* if $\mathcal{E} \cup \mathcal{D} \cup \{Sensed(\sigma)\}$ is a consistent theory.

Now the projection problem including sensing is formulated as deciding if $\mathcal{D} \cup \{Sensed(\sigma)\} \models_\mathcal{E} \phi[end(\sigma)]$, where $\phi \in \mathcal{L}$, and σ is a consistent history.

In the rest of this section, we assume that $\mathcal{D}_{sf}$ is atomic. To prepare for the definition of just-in-time-history, we first extend our regression and progression operators to incorporate sensing. The ideas are quite simple. For example, if we regress the formula $clear(c_1) \wedge eh(c_1)$ wrt $(sense_{eh}(c_1), 1)$, we should obtain $clear(c_1)$. If we progress a proper KB Σ wrt $(sense_{eh}(c_1), 1)$, we should obtain $\Sigma \cup \{eh(c_1)\}$.

Let $\alpha = A(\vec{c})$ be a ground action, and let $\mu \in \{0, 1\}$. We define $\mathcal{R}_{(\alpha,\mu)}(\phi)$ as follows: If α is an ordinary action, then $\mathcal{R}_{(\alpha,\mu)}(\phi) = \mathcal{R}_\alpha(\phi)$. Otherwise, let the SFA be $SF(A(\vec{x}), s) \equiv F(\vec{x}, s)$. Then $\mathcal{R}_{(\alpha,\mu)}(\phi)$ is the formula obtained from ϕ by replacing each atom $F(\vec{t})$ with $\vec{t} = \vec{c} \vee F(\vec{t})$ when $\mu = 1$ and with $F(\vec{t}) \wedge \vec{t} \neq \vec{c}$ when $\mu = 0$.

Now we turn to progression with sensing. Let Σ be proper. Let $\alpha = A(\vec{c})$ be a ground action, and let $\mu \in \{0, 1\}$. We define $\mathcal{P}_{(\alpha,\mu)}(\Sigma)$ as follows: If α is an ordinary action, then $\mathcal{P}_{(\alpha,\mu)}(\Sigma) = \mathcal{P}_\alpha(\Sigma)$. Otherwise, let the SFA be $SF(A(\vec{x}), s) \equiv F(\vec{x}, s)$. Then $\mathcal{P}_{(\alpha,1)}(\Sigma) = \Sigma \cup \{F(\vec{c})\}$, and $\mathcal{P}_{(\alpha,0)}(\Sigma) = \Sigma \cup \{\neg F(\vec{c})\}$. We have the following extended progression theorem:

Theorem 7 *Let σ be a consistent history. Then*

1. *$\mathcal{E} \cup \mathcal{P}_\sigma(\Sigma_0)$ is consistent. Hence $\mathcal{P}_\sigma(\Sigma_0)$ is proper.*
2. *For every $\phi \in \mathcal{L}$, $V[\mathcal{P}_\sigma(\Sigma_0), \phi] = V[\Sigma_0, \mathcal{R}_\sigma(\phi)]$.*

Another concept we need is dependency set.

Definition 12 The *dependency set* of a formula ϕ wrt an ordinary action α, denoted by $DS_\alpha(\phi)$, is the set of ground atoms that appear in $\gamma^+_F(\alpha)$ or $\gamma^-_F(\alpha)$ for some fluent F in ϕ.

For example, let $\alpha = move(c_1, c_2, c_3)$. Then $DS_\alpha(clear(x))$ is the empty set, and $DS_\alpha(eh(x)) = \{eh(c_3)\}$.

Definition 13 Let σ be a consistent history, and $\phi \in \mathcal{L}$. We say that σ is a *just-in-time-history* (JIT-history) for ϕ if for every division $\sigma_1 \cdot (\alpha, \mu) \cdot \sigma_2$ of σ, if α is an ordinary action then $\mathcal{P}_{\sigma_1}(\Sigma_0)$ is complete wrt $DS_\alpha(\mathcal{R}_{\sigma_2}(\phi))$, and if α is a sensing action then the sensed fluent appears in $\mathcal{R}_{\sigma_2}(\phi)$.

Intuitively, σ is a JIT-history for ϕ if whenever performing an ordinary action α, the agent has complete knowledge about the context of α wrt fluents related to ϕ. This complete knowledge may come from the sensing actions preceding α. For example, let $\Sigma_0 = \{clear(c_1), on(c_1, c_2), clear(c_3)\}$. Then the history $(sense_{eh}(c_3), 1) \cdot (move(c_1, c_2, c_3), 1)$ is a JIT history for the formula $\exists x[clear(x) \wedge eh(x)]$. Note that here the agent has incomplete knowledge about both fluents $clear$ and eh. Thus a JIT history does not require complete knowledge about the component fluents of the query.

For JIT histories and initial KBs that are proper, we have the following extended regression theorem:

Theorem 8 *Let σ be a JIT-history for ϕ, and let Σ_0 be proper. Then $\mathcal{D} \cup \{Sensed(\sigma)\} \models_\mathcal{E} \phi[end(\sigma)]$ iff $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\sigma(\phi)$.*

By a proof essentially the same as the one in the previous section, we get the following result:

Theorem 9 *Let σ be a JIT-history for ϕ. Then $\mathcal{P}_\sigma(\Sigma_0) \models_\mathcal{E} \phi$ iff $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\sigma(\phi)$.*

Now let σ be a JIT-history for $\phi \in \mathcal{NF}$. Then we have $V[\mathcal{P}_\sigma(\Sigma_0), \phi] = 1$ iff $\mathcal{P}_\sigma(\Sigma_0) \models_\mathcal{E} \phi$ iff $\Sigma_0 \models_\mathcal{E} \mathcal{R}_\sigma(\phi)$ iff $\mathcal{D} \cup \{Sensed(\sigma)\} \models_\mathcal{E} \phi[end(\sigma)]$. Thus when a history is just-in-time for a normal form query, PV is again logically complete for projection.

6 Related Work

As mentioned in the introduction, Lin and Reiter [1997] give a systematic study of classical progression. As a part of their study, they view STRIPS as a mechanism for computing progression and thus provide a logical semantics for STRIPS. In this respect, they consider strictly context-free SSAs and initial KBs in the form of relational databases or sets of ground literals. These are special cases of local-effect SSAs and proper KBs, and our weak progression coincides with classical progression in these cases. Son and Baral [2001] propose the so-called 0-approximation semantics for an extension of action language $\mathcal{A}$. They define an a-state (approximate state) as a consistent set of fluent literals, and define a transition function which maps an a-state and an action into the next a-state. So 0-approximation is essentially a kind of approximate progression. However, their work is restricted to the propositional case, and our progression coincides with theirs therein. Amir and Russell [2003] present efficient algorithms for (approximate) logical filtering, where filtering means updating an agent's belief state in response to actions and observations. So logical filtering is essentially progression. But again, their work is restricted to the propositional case. De Giacomo and Mancini [2004] study how to exploit relational database technology to implement progression, but only when the initial KB has complete knowledge. We get to use database techniques in the incomplete case via the results in [Liu and Levesque, 2003].

The idea of progression is widely used in planning under incomplete knowledge. Most systems use propositional representations, for example, BDDs [Cimatti and Roveri, 2000], and clauses [Brafman and Hoffmann, 2004]. Being propositional makes it possible for them to consider arbitrary incomplete knowledge and perform classical progression. Although techniques are employed so that the systems can achieve reasonable performance in practice, there is no theoretical guarantee of the tractability of their solutions to projection. The PKS system of Petrick and Bacchus [2002] uses a first-order representation. The form of incomplete knowledge they consider is mainly a set of ground literals but with some other features. The general idea behind their progression is similar to ours, but without a semantical characterization of what it preserves. Moreover, they do not address the issue of the restrictions they need to get completeness.

The idea of JIT histories in this paper comes from [De Giacomo and Levesque, 1999]. They use JIT histories to obtain complete knowledge about the component fluents of the query. However, we use JIT histories only to obtain complete knowledge about the context of actions to be performed.

7 Conclusions

In this paper, we have proposed a tractable, sound, and sometimes complete solution to the projection problem in the presence of context-dependent actions and incomplete first-order knowledge in the form of a proper KB. Our solution is via a version of progression that preserves properness and V.

For simplicity of presentation, in this paper we require actions to have local effects only, and we make the extra requirement that context formulas (c.f. Definition 5) be quantifier-free. However, the soundness and completeness results in this paper will still hold if we relax these two requirements. The tractability result will also hold if context formulas use a bounded number of variables. As for local effects, the tractability result only needs them to ensure that the progressed KB is not much larger than the original KB. We believe that there are other ways of doing this that would include a substantial class of actions with universal effects. Finally, our definition of JIT-history rules out sensing a fluent that is irrelevant to the query. We believe that this can be handled in a more natural way by a restriction on basic action theories.

For the future, we would like to conduct experimental evaluation of our solution to projection, and apply it to first-order planning systems. Also, we would like to extend our work here to deal with functional fluents and disjunctive incomplete knowledge.

References

[Amir and Russell, 2003] E. Amir and S. Russell. Logical filtering. In *Proc. IJCAI-03*, pages 75–82, 2003.

[Brafman and Hoffmann, 2004] R. Brafman and J. Hoffmann. Conformant planning via heuristic forward search. In *Proc. ICAPS-04*, pages 355–364, 2004.

[Cimatti and Roveri, 2000] A. Cimatti and M. Roveri. Conformant planning via symbolic model checking. *Journal of Artificial Intelligence Research*, 13:305–338, 2000.

[De Giacomo and Levesque, 1999] G. De Giacomo and H. J. Levesque. Projection using regression and sensors. In *Proc. IJCAI-99*, pages 160–165, 1999.

[De Giacomo and Mancini, 2004] G. De Giacomo and T. Mancini. Scaling up reasoning about actions using relational database technology. In *Proc. AAAI-04*, pages 245–250, 2004.

[Levesque *et al.*, 1997] H. J. Levesque, R. Reiter, Y. Lespérance, F. Lin, and R. Scherl. Golog: A logic programming language for dynamic domains. *J. of Logic Programming*, 31:59–84, 1997.

[Levesque, 1998] H. J. Levesque. A completeness result for reasoning with incomplete first-order knowledge bases. In *Proc. KR-98*, pages 14–23, 1998.

[Lin and Reiter, 1997] F. Lin and R. Reiter. How to progress a database. *Artificial Intelligence*, 92(1–2):131–167, 1997.

[Liu and Levesque, 2003] Y. Liu and H. J. Levesque. A tractability result for reasoning with incomplete first-order knowledge bases. In *Proc. IJCAI-03*, pages 83–88, 2003.

[McCarthy and Hayes, 1969] J. McCarthy and P. Hayes. Some philosophical problems from the standpoint of artificial intelligence. In *Machine Intelligence*, volume 4, pages 463–502. 1969.

[Petrick and Bacchus, 2002] R. Petrick and F. Bacchus. A knowledge-based approach to planning with incomplete information and sensing. In *Proc. AIPS-02*, pages 212–222, 2002.

[Petrick and Levesque, 2002] R. Petrick and H. J. Levesque. Knowledge equivalence in combined action theories. In *Proc. KR-02*, pages 303–314, 2002.

[Reiter, 2001] R. Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.

[Son and Baral, 2001] T. C. Son and C. Baral. Formalizing sensing actions – A transition function based approach. *Artificial Intelligence*, 125(1–2):19–91, 2001.