First-Order Progression beyond Local-Effect and Normal Actions

Daxin Liu[1] , **Jens Claßen**[2]
[1]The University of Edinburgh
[2]Roskilde University
daxin.liu@ed.ac.uk, classen@ruc.dk

Abstract

One of the fundamental problems in reasoning about action is progression, which is to update a knowledge base according to the effects of an action into another knowledge base that retains all proper information. The problem is notoriously challenging, as in general, it requires second-order logic. Efforts have been made to find fragments where progression is first-order definable. Liu and Lakemeyer showed that for actions that have only local effects, progression is always first-order definable. They also generalized the result to so-called normal actions, that allow for non-local effects, as long as the affected fluent predicates only depend on local-effect ones, under certain restrictions on the knowledge base. In addition, they showed that for so-called proper+ knowledge bases, progression for normal actions can be efficient under reasonable assumptions. In this paper, we consider a larger class of theories, called the *acyclic* ones, that strictly subsumes normal actions. In such theories, dependencies between non-local effect fluent predicates are allowed, as long as they do not contain any cycles. We prove progression to be equally first-order definable for this class. Furthermore, under similar but stronger assumptions than those made by Liu and Lakemeyer, we show that progression is efficient as well.

1 Introduction

In reasoning about action and change, *projection* is perhaps the most fundamental problem. Given a sequence of actions and a target formula, the task is to determine whether the formula comes to hold after executing the action sequence. Typically, the domain in question is represented through an action theory that includes, among other things, a knowledge base (KB) describing the initial state of the world, as well as axioms encoding the pre- and post-conditions of actions. There are at least two different approaches to projection: On the one hand, there is *regression*, where one transforms the formula backward into an equivalent one about the initial situation, and then checks whether that formula holds according to the initial KB. Conversely, in *progression*, one updates the initial KB to reflect the changes brought about by the action sequence and then determines whether the goal formula is entailed. In the well-studied Situation Calculus [McCarthy and Hayes, 1969; Reiter, 2001] formalism, regression is very straightforward. However, in practice, it very quickly becomes unfeasible, as the resulting formula can grow exponentially in size (unless one introduces new fluents as in [van Ditmarsch *et al.*, 2007]), which is particularly problematic the longer the action sequence gets. One benefit of progression is that one only has to progress the KB once, and can then check multiple queries against it. However, the problem of computing a progression for a given action and KB is notoriously challenging: A strong negative result due to Lin and Reiter [1997] is that in general, the progression of a first-order theory may require second-order logic. Vassos and Levesque [2008] later showed that the second-order nature is inescapable in the sense that even when allowing for infinite theories, a restriction to first-order theories is strictly weaker.

Efforts have been made to identify restricted classes of theories, where first-order definability can be guaranteed. In their seminal work, Lin and Reiter presented such results for so-called *relatively complete* theories, where the initial KB has complete knowledge about the state of the world, and *context-free theories*, where action effects are limited to adding and deleting ground literals as in the classical STRIPS [Fikes and Nilsson, 1971] planning language. Liu and Lakemeyer [2009], extending earlier work due to Vassos et al. [2008], later showed the existence of a first-order progression for a less restricted class of theories, namely where actions only have *local effects*. Intuitively, this means that actions must mention all affected objects explicitly in their arguments. For example, an action $move(x, y, z)$ for moving a block x from y to z only affects the truth of $on(x, y)$ and $on(x, z)$. Classical examples of actions that are *not* local-effect include moving a briefcase, which moves all (unmentioned) contained items along with it, and exploding a bomb, which destroys all (unmentioned) objects in its vicinity. To capture such domains, Liu and Lakemeyer [2009] generalized the result on local-effect actions to so-called *normal* actions, which allow for non-local effects, as long as the affected fluent predicates only depend on ones that are themselves only subject to local effects, under certain conditions to the initial KB. They also showed that for certain types of KB, called $proper^+$ [Lakemeyer and Levesque, 2002], progression is

computationally efficient.

An example that goes beyond what is expressable by means of normal actions, and that is hence not supported by Liu and Lakemeyer's results, is when objects *contained* in a box that is *on* a shelf become *broken* from dropping the box. Here, $broken$ is a non-local-effect predicate that depends on another non-local-effect predicate, nameley on. In this paper, we show that for a class of actions that includes such scenarios, and that strictly subsumes that of normal actions, progression is also first-order definable. In such action theories, called *acyclic*, dependencies between non-local effect fluent predicates are allowed, as long as they do not contain any cycles. We also show that progression is efficient under reasonable assumptions, where dependencies among fluents do not introduce significant additional complexity.

The remainder of the paper is organized as follows. The following section presents formal preliminaries about action theories and progression and reviews the first-order progression result on local-effect actions and normal actions by Liu and Lakemeyer [2009]. Our results on progression of the more general class of acyclic theories are presented in Section 3. We then go on to provide computability results in Section 4, after which we discuss related work and conclude.

2 Preliminaries

In this section, we review progression in the Situation Calculus and the result on local-effect and normal actions by [Liu and Lakemeyer, 2009].

We start with a first-order (FO) language $\mathcal{L}$ with equality. For simplicity, we only consider predicates and ignore functions. The set of formulas of $\mathcal{L}$ is the least set that contains the atomic formulas, and if ϕ and ψ are in the set and x is a variable, then $\neg\phi$, $\phi \wedge \psi$ and $\forall x\phi$ are in the set. The connectives $\vee, \supset, \equiv$, and $\exists$ are understood as the usual abbreviations. For readability, we will use parentheses around quantifiers to indicate the scopes, and "dot" to indicate that the quantifier preceding the dot has maximum scope, e.g., $\forall x.\phi(x) \supset \psi(x)$ stands for $\forall x(\phi(x) \supset \psi(x))$. Leading universal quantifiers might be omitted in writing sentences, i.e., we assume that free variables are implicitly $\forall$-quantified from the outside. E.g., we identify $\phi(x)$ with $\forall x.\phi(x)$. We use $\psi \Leftrightarrow \psi$ to mean ϕ and ψ are logically equivalent. Let ψ be a formula, and let μ and μ' be two expressions (terms or formulas). We denote by $\phi(\mu/\mu')$ the result of simultaneously substituting every occurrence of μ in ϕ with μ'.

2.1 Basic Action Theories

The situation calculus [Reiter, 2001] $\mathcal{L}_{sc}$ is a many-sorted language with some second-order (SO) features suitable for describing dynamic worlds. There are three sorts: *action*, *situation*, and *object*. $\mathcal{L}_{sc}$ contains the following features: a distinct constant S_0 denoting the initial situation; a binary function $do(a, s)$ deriving the resulting situation from doing action a in situation s; a binary relation $Poss(a, s)$ expressing action a being executable in situation s; action functions, e.g. $move(x, y)$; a finite set of fluent predicates, i.e., predicates whose last argument is a situation term.

A (possibly second-order) formula ϕ is uniform in a situation term s if ϕ does not mention any other situation terms except s, does not quantify over situation variables, and does not mention $Poss$.

The dynamics of a domain is specified by a *basic action theory* (BAT) in $\mathcal{L}_{sc}$ of the form

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

1. Σ_{ind} is a set of domain-independent axioms that ensure situations are well-structured;
2. $\mathcal{D}_{ap}$ is a set of action precondition axioms;
3. $\mathcal{D}_{ss}$ is a set of successor state axioms (SSAs), one for each fluent predicate F, of the form

   $$F(\vec{x}, do(a, s)) \equiv \gamma_F^+(\vec{x}, a, s) \vee \neg\gamma_F^-(\vec{x}, a, s) \wedge F(\vec{x}, s),$$

   where γ_F^+ and γ_F^- are uniform in s. γ_F^+ and γ_F^- are the conditions under which the action causes F to become true and false, respectively;
4. $\mathcal{D}_{una}$ is the set of unique names axioms for actions: $A(\vec{x}) \neq A'(\vec{y})$, and $A(\vec{x}) = A(\vec{y}) \supset \vec{x} = \vec{y}$, where A, A' are distinct action symbols;
5. $\mathcal{D}_{S_0}$, the initial database (or initial KB), is a finite set of sentences uniform in S_0.

Successor state axioms constitute Reiter's [1991] solution to the frame problem. In particular, it is required that for all fluent predicates F, $\mathcal{D} \models \neg(\gamma_F^+ \wedge \gamma_F^-)$. Henceforth, given a ground action α, we use S_α to refer to the situation $do(\alpha, S_0)$.

2.2 Progression

A progression should retain all proper information, i.e., logical entailments in terms of the future of the initial KB. The below definition by [Vassos and Levesque, 2013] formalizes this intuition, and it is equivalent to the original model-theoretical one by [Lin and Reiter, 1997].

Definition 1 (Progression)**.** Let $\mathcal{D}$ be a basic action theory, α a ground action, and $\mathcal{D}_{S_\alpha}$ a set of (first-order or second-order) sentences uniform in S_α. We say that $\mathcal{D}_{S_\alpha}$ is a *progression* of $\mathcal{D}_{S_0}$ wrt $\alpha, \mathcal{D}$ iff for every sentence ϕ uniform in S_α,

$$\mathcal{D} \models \phi \text{ iff } (\mathcal{D} - \mathcal{D}_{S_0}) \cup \mathcal{D}_{S_\alpha} \models \phi.$$

Lin and Reiter [1997] proved that progression is always second-order definable. We write the instantiation of $\mathcal{D}_{ss}$ wrt α and S_0 as $\mathcal{D}_{ss}[\alpha, S_0]$, i.e. $\mathcal{D}_{ss}[\alpha, S_0]$ is the set of sentences $F(\vec{x}, do(\alpha, S_0)) \equiv \Phi_F(\vec{x}, \alpha, S_0)$, where Φ_F denotes the right-hand side of the SSA for F. Let $F_1, \ldots F_n$ be the set of all fluents. For each fluent F_i, we introduce a new predicate symbol P_i. We use $\phi \uparrow S_0$ to denote the result of replacing every $F_i(\vec{t}, S_0)$ in ϕ by $P_i(\vec{t})$ and call P_i the lifting predicate for F_i. For a finite set of formulas Σ, we also use Σ to denote the conjunctions of its elements. Using this notation, the following is a progression of $\mathcal{D}_{S_0}$ wrt α:

$$\exists \vec{R}.\{(\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{ss}[\alpha, S_0]) \uparrow S_0\}(\vec{P}/\vec{R}) \qquad (1)$$

where $\vec{R} = \{R_1, \ldots, R_n\}$ are SO predicate variables.

Efforts have been made to identify fragments of the Situation Calculus where progression is first-order definable, i.e. conditions under which Eq. (1) is equivalent to a first-order theory. For instance, [Lin and Reiter, 1997] showed that this

is the case if the initial KB is *relatively complete*, i.e., for every sentence ϕ uniform in S_0, KB entails either ϕ or its negation, or if the basic action theory is *context-free*, i.e. actions' effects are independent of situations. See also [Vassos and Patrizi, 2013] for a classification of FO definable KBs and action theories.

Local-Effect Actions

Liu and Lakemeyer [2009] (henceforth LL09) showed that if the basic action theory is *local-effect*, then progression is always FO definable. Intuitively, a ground action has local effects if it only affects the truth of ground fluent atoms that mention only the action's parameters. For example, an action $move(x, y, z)$ for moving a block x from y to z only affects the truth of $on(x, y)$ and $on(x, z)$.

Definition 2. An SSA is *local-effect* if both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^-(\vec{x}, a, s)$ are disjunctions of formulas of the form $\exists \vec{z}[a = A(\vec{u}) \wedge \phi(\vec{u}, s)]$, where A is an action function, $\vec{u}$ contains $\vec{x}$, $\vec{z}$ is the remaining variables of $\vec{u}$, and ϕ is called the context. An action theory is local-effect if each SSA is local-effect.

The result in LL09 to progress local-effect action theories is based on *forgetting* [Lin and Reiter, 1994]. Intuitively, forgetting a ground atom (or predicate) in a theory leads to a weaker theory that entails the same set of sentences that are "irrelevant" to the atom (or predicate). Forgetting in propositional logic is equivalent to propositional existential quantification: the result of forgetting atom p from formula ϕ is $\phi(p/\text{TRUE}) \vee \phi(p/\text{FALSE})$, where $\phi(p/\text{TRUE})$ (resp. $\phi(p/\text{FALSE})$) denotes the result of replacing each occurrence of p in ϕ by TRUE (resp. FALSE). Forgetting a ground atom in an FO theory is more complicated but can be computed in a similar spirit; we refer interested readers to [Lin and Reiter, 1994]. Lastly, forgetting a predicate P in a theory T, written as $forget(T, P)$, amounts to replacing the predicate with a SO variable and existentially quantifying it from the outside, i.e. $\exists R.T(P/R)$. Progression as in Eq. (1) thus amounts to adding the effects of the action ($\mathcal{D}_{ss}[\alpha, S_0]$) and forgetting all lifting predicates of fluents (quantified existentially from the outside). LL09 observed that for a local-effect action theory, every ground action will affect only finitely many fluent atoms, the so-called *characteristic set* Ω, determined by the action's parameters. Hence, forgetting the lifting predicates can be reduced to forgetting these finitely many instances, resulting in a first-order theory. For space reasons, we do not go into details about how to compute such an FO progression by forgetting, but simply assume that we can apply the procedure described by LL09 when we need to forget local-effect fluents.

Normal Actions

Actions with *global effects* are ubiquitous, for example, exploding a bomb will destroy every fragile object in its vicinity. The inability of local-effect action theories to capture such dynamic domains motivated LL09 to propose a more general version of action theories that admit so-called *normal actions*. The idea is to allow *non-local-effect* fluents in a limited fashion. By imposing additional constraints on the initial KB and SSAs, a technique for SO quantifier elimination based on Ackermann's Lemma [1935] can be applied to forget the lifting predicates for those fluents.

Definition 3. A finite theory T is *semi-definitional* wrt a predicate P if the only occurrence of P in T is of the form $P(\vec{x}) \supset \phi(\vec{x})$ or $\psi(\vec{x}) \supset P(\vec{x})$. ϕ (respectively ψ) is called a necessary (sufficient) condition of P.

Let WSC_P (weakest sufficient condition) be the disjunction of formulas $\psi(\vec{x})$ such that $\psi(\vec{x}) \supset P(\vec{x})$ is in T, and SNC_P (strongest necessary condition) be the conjunction of formulas $\phi(\vec{x})$ with $P(\vec{x}) \supset \phi(\vec{x})$ in T.

Theorem 1 ([Liu and Lakemeyer, 2009])**.** *Let T be finite and semi-definitional wrt P, and T' the set of sentences in T not mentioning P. Then $forget(T, P) \Leftrightarrow T' \wedge \forall \vec{x}.\text{WSC}_P(\vec{x}) \supset \text{SNC}_P(\vec{x})$.*

This provides a means to eliminate the SO quantifiers introduced by forgetting a predicate.

Proposition 1. *In any model of $\mathcal{D}$, the sentence $F(\vec{x}, S_\alpha) \equiv \gamma_F^+(\vec{x}, \alpha, S_0) \vee \neg\gamma_F^-(\vec{x}, \alpha, S_0) \wedge F(\vec{x}, S_0)$ is equivalent to the conjunction of following sentences: $\neg\gamma_F^+ \wedge F(\vec{x}, S_\alpha) \supset F(\vec{x}, S_0)$, $F(\vec{x}, S_0) \supset \gamma_F^- \vee F(\vec{x}, S_\alpha)$, $\gamma_F^+ \supset F(\vec{x}, S_\alpha)$, and $\gamma_F^- \supset \neg F(\vec{x}, S_\alpha)$.* [1]

The proposition suggests that after lifting, SSAs are semi-definitional wrt the respective lifting predicates.

Definition 4. A ground action α is said to have *local effects* on a fluent F, if by using $\mathcal{D}_{una}$, $\gamma_F^+(\vec{x}, \alpha, s)$ and $\gamma_F^-(\vec{x}, \alpha, s)$ can be simplified to a disjunction of formulas of the form $\vec{x} = \vec{t} \wedge \psi(s)$, where $\vec{t}$ is a vector of ground terms, and ψ is a formula whose only free variable is s.

Let $\text{LE}(\alpha)$ be the set of fluents on which α has local effects and $\text{NLE}(\alpha) = \{F \mid F \text{ is in } \mathcal{D}, \text{and } F \notin \text{LE}(\alpha)\}$, i.e., the set of remaining fluents.

Definition 5 (Normal Action)**.** A ground action α is *normal* if for each fluent F, all the fluents that appear in γ_F^+ and γ_F^- are in $\text{LE}(\alpha)$.

Clearly, local-effect actions are normal actions. We say that $\mathcal{D}_{S_0}$ is *normal* wrt α if $\mathcal{D}_{S_0}$ is semi-definitional wrt each fluent in $\text{NLE}(\alpha)$.

Theorem 2 ([Liu and Lakemeyer, 2009])**.** *Let $\mathcal{D}_{S_0}$ be normal wrt a normal action α, then progression of $\mathcal{D}_{S_0}$ wrt α is FO definable and computable.*

Intuitively, this is because for such a theory $\mathcal{D}$ and action α, one can use Theorem 1 to forget the lifting predicates of fluents in $\text{NLE}(\alpha)$, and thereafter use the techniques for local-effect theories to forget lifting predicates for fluents in $\text{LE}(\alpha)$.

Example 1. Consider the domain that is described by the two fluents $broken(x, s)$ and $shielded(x, s)$ that say, respectively, that object x is broken and shielded in situation s. The action $explode$ will destroy everything that is unshielded. The

[1] The proposition is almost identical to Proposition 4.3 of LL09, the only difference being that they use $\neg\gamma_F^+ \wedge \gamma_F^- \supset \neg F(\vec{x}, S_\alpha)$ instead of $\gamma_F^- \supset \neg F(\vec{x}, S_\alpha)$. Our simplification is justified by the assumption that $\mathcal{D} \models \neg(\gamma_F^+ \wedge \gamma_F^-)$.

following BAT expresses such a domain:

$$\begin{aligned} broken(x, do(a,s)) &\equiv a = explode \wedge \neg shielded(x,s) \vee \\ &\quad broken(x,s) \\ shielded(x, do(a,s)) &\equiv shielded(x,s) \end{aligned}$$

Clearly, the ground action $\alpha = explode$ has local effects on *shielded* (in fact it has no effects on *shielded* at all) and it is a normal action with $broken \in \mathrm{NLE}(\alpha)$.

Let $\mathcal{D}_{S_0}$ be $\{broken(x, S_0) \supset x = A\}$, then $\mathcal{D}_{S_0}$ is normal wrt *explode*. Grounding $\mathcal{D}_{ss}$ wrt *explode* and applying Prop. 1, we obtain

$$\left\{ \begin{array}{l} shielded(x,S_0) \wedge broken(x,S_\alpha) \supset broken(x,S_0), \\ broken(x,S_0) \supset broken(x,S_\alpha), \\ \neg shielded(x,S_0) \supset broken(x,S_\alpha) \end{array} \right\}$$

Applying Theorem 1 to forget the lifting predicate for $broken(x, S_0)$ and substituting S_0 by S_α, we obtain

$$\left\{ \begin{array}{l} \neg shielded(x,S_\alpha) \supset broken(x,S_\alpha), \\ shielded(x,S_\alpha) \wedge broken(x,S_\alpha) \supset x = A \end{array} \right\}.$$

3 Progressing Acyclic Actions

Although the class of normal actions is capable of capturing the phenomenon of global effects, the requirement that for each fluent F, fluents in γ_F^+ and γ_F^- have to be local-effect is a limitation. Among other things, it disallows a non-local-effect fluent to depend on other non-local-effect fluents. In this section, we lift this limitation and show that first-order definability preserves for a more general class of actions, i.e. *acyclic actions*, where non-local-effect fluents can interact in a complex manner as long as the dependency among fluents does not contain cycles. Intuitively, for such actions, one can use Theorem 1 to iteratively forget the lifting predicates of non-local-effect fluents, following the order of fluent dependencies. Additional syntactical conditions on the SSAs guarantee that after each such step, the resulting KB is in the right form to apply Theorem 1 again.

To begin with, we broaden the class of formulas that can be handled by considering formulas that can be *equivalently* transferred into a semi-definitional form. Note that it is in general undecidable to determine if a formula is equivalent to a one in semi-definitional form wrt a predicate.

Definition 6. A formula $\phi(\vec{x})$ is said to be in *good form* wrt a predicate P if it is of the form

$$[\psi(\vec{x}) \vee P(\vec{t})] \wedge [\psi'(\vec{x}) \vee \neg P(\vec{t})] \wedge \psi''(\vec{x})$$

where ψ, ψ', ψ'' contains no P, and terms in $\vec{t}$ are either ground terms or free variables among $\vec{x}$.

Proposition 2. *If $\phi(\vec{x})$ is in* good form *wrt a predicate P, then $\neg\phi(\vec{x})$ can be rewritten in good form wrt P.*

Lemma 1. *If $\phi(\vec{x})$ is in* good form *wrt a predicate P, then $\phi(\vec{x})$ can be rewritten to be semi-definitional wrt P.*

Proof. $\phi(\vec{x})$ is equivalent to the set

$$\Phi = \{\forall\vec{x}(\psi(\vec{x}) \vee P(\vec{t})), \forall\vec{x}(\psi'(\vec{x}) \vee \neg P(\vec{t})), \forall\vec{x}\psi''(\vec{x})\}. \quad (2)$$

The first sentence (likewise for the second) is equivalent to $\forall\vec{y}.(\exists\vec{x}.\ \vec{y} = \vec{t} \wedge \neg\psi(\vec{x})) \supset P(\vec{y})$. The third one is obvious as it does not mention P. □

Definition 7. Given predicates $\vec{P} = \{P_1, \ldots, P_n\}$, a finite theory T is said to be *separably semi-definitional* wrt $\vec{P}$, if each sentence $\phi_i \in T$ mentions at most one predicate $P_i \in \vec{P}$ and ϕ_i is semi-definitional wrt P_i.

Separably semi-definitional theories exclude formulas like $P_i(\vec{x}) \supset P_j(\vec{x})$ where both P_i, P_j are in $\vec{P}$.

Definition 8 (Dependency Graph)**.** Given successor state axioms $\mathcal{D}_{ss}$ and a ground action α, the dependency graph G of fluents in $\mathcal{D}_{ss}$ wrt α is a directed graph defined as follows: (1) the vertices of G are the set of fluents; (2) there is an edge $F \to F'$ for two fluents F, F' iff $F \in \mathrm{NLE}(\alpha)$ and F' appears in γ_F^+ or γ_F^-.

Definition 9 (Acyclic Action)**.** A ground action α is *acyclic* (or has acyclic effects, or the BAT is acyclic wrt α) if

- its dependency graph G is acyclic; and
- for each fluent F, either all fluents in γ_F^+ and γ_F^- are in $\mathrm{LE}(\alpha)$, or both γ_F^+ and γ_F^- are in good form wrt *at most* one fluent in $\mathrm{NLE}(\alpha)$.

Theorem 3. *Let α be an acyclic action, and $\mathcal{D}_{S_0}$ be separably semi-definitional wrt $\mathrm{NLE}(\alpha)$. Then the progression of $\mathcal{D}_{S_0}$ wrt α is first-order definable and computable.*

Proof. The non-trivial part lies in forgetting the lifting predicates for fluents in $\mathrm{NLE}(\alpha)$. We define the depth of a fluent F in G as the length of the longest path starting in F.

We forget the lifting predicates in decreasing depth order. For a fluent F with maximal depth, lifting its instantiated SSA, we obtain (by Prop. 1)

$$\neg\gamma_F^+ \wedge F(\vec{x}, S_\alpha) \supset P(\vec{x}) \quad (3a)$$

$$P(\vec{x}) \supset \gamma_F^- \vee F(\vec{x}, S_\alpha) \quad (3b)$$

$$\gamma_F^+ \supset F(\vec{x}, S_\alpha) \quad (3c)$$

$$\gamma_F^- \supset \neg F(\vec{x}, S_\alpha) \quad (3d)$$

Clearly, only Eq. (3a) and Eq. (3b) contain the lifting predicate P. Since $\mathcal{D}_{S_0}$ is separably semi-definitional wrt $\mathrm{NLE}(\alpha)$, forgetting P in $\{(\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \cup \mathcal{D}_{ss}[\alpha, S_0]) \uparrow S_0\}$ is equivalent to replacing the formulas mentioning P in the lifted KB with the conjunction of formulas

$$SC_P(\vec{x}) \supset NC_P(\vec{x}) \quad (4a)$$

$$\neg\gamma_F^+ \wedge F(\vec{x}, S_\alpha) \supset NC_P(\vec{x}) \quad (4b)$$

$$SC_P(\vec{x}) \supset \gamma_F^- \vee F(\vec{x}, S_\alpha) \quad (4c)$$

where $NC_P(\vec{x})$ is the conjunction of necessary conditions for P in $\mathcal{D}_{S_0} \uparrow S_0$, and $SC_P(\vec{x})$ is the disjunction of sufficient conditions for P in $\mathcal{D}_{S_0} \uparrow S_0$. Note that by assumption both $NC_P(\vec{x})$ and $SC_P(\vec{x})$ contain no fluents in $\mathrm{NLE}(\alpha)$.

By Prop. 2 and Lemma 1, the new formulas (4a)–(4c) together with (3c) and (3d) can all be transformed into a formula that is semi-definitional wrt *at most* one lifting predicate of a fluent in $\mathrm{NLE}(\alpha)$. We add the transformed result to $\mathcal{D}_{S_0}$. Note the assumption that F has maximal depth guarantees that F is not mentioned in the resulting formulas, and so forgetting the remaining lifting predicates can be iterated. □

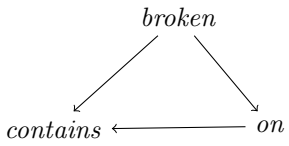

Figure 1: Dependency graph for BAT in Example 2.

Example 2. Consider a box domain with three fluents, adapted from [Claßen and Zarrieß, 2017]: $contains(x, y, s)$ says that x contains y, $on(x, y, s)$ says that x is on y, and $broken(x, s)$ says that x is broken in situation s. Action $drop(x, y)$ denotes dropping container x from shelf y, causing all things in x to become broken and no longer being positioned on y. The SSAs are given by

$$\gamma^+_{broken} := \exists y, z.a = drop(y, z) \wedge on(y, z, s) \wedge contains(y, x, s)$$

$$\gamma^-_{broken} := \text{FALSE} \qquad \gamma^+_{on} := \text{FALSE}$$

$$\gamma^-_{on} := \exists z.a = drop(z, y) \wedge (z = x \vee contains(z, x, s))$$

$$\gamma^+_{contains} \equiv \gamma^-_{contains} \equiv \text{FALSE}$$

where $drop$ has no effect on $contains$. Hence, for ground action $\alpha = drop(Box, Shelf)$, we have $\mathrm{LE}(\alpha) = \{contains\}$, and $\mathrm{NLE}(\alpha) = \{on, broken\}$. We can confirm that α is acyclic as follows. Obviously, the dependency graph, as shown in Fig. 1, does not contain any cycles. Furthermore, for fluent $broken$, grounding γ^+_{broken} by α yields $on(Box, Shelf, s) \wedge contains(Box, x, s)$, while γ^-_{broken} is simply FALSE, so both are in good form wrt on. Moreover, for fluent on, the SSA only mentions fluents from $\mathrm{LE}(\alpha)$. Now let $\mathcal{D}_{S_0}$ be

$$\left\{ \begin{array}{l} contains(Box, Vase, S_0), \\ x = Box \wedge y = Shelf \supset on(x, y), \\ broken(x, S_0) \supset \neg\exists y.contains(y, x, S_0) \end{array} \right\}$$

That is to say, Box contains $Vase$, Box is on $Shelf$, and every broken object is not contained in anything. Applying Prop. 1 to $(\mathcal{D}_{S_0} \cup \mathcal{D}_{ss}) \uparrow S_0$, we obtain (let P, P' be the corresponding lifting predicates for on and $broken$):

$$T_0 = \left\{ \begin{array}{l} contains(Box, Vase, S_0), \\ x = Box \wedge y = Shelf \supset P(x, y), \\ P'(x) \supset \neg\exists y.contains(y, x, S_0) \end{array} \right\} \cup$$

$$\left\{ \begin{array}{l} \neg[P(Box, Shelf) \wedge contains(Box, x, S_0)] \\ \qquad \wedge broken(x, S_\alpha) \supset P'(x), \\ P'(x) \supset broken(x, S_\alpha), \\ P(Box, Shelf) \wedge contains(Box, x, S_0) \\ \qquad \supset broken(x, S_\alpha), \end{array} \right\} \cup$$

$$\left\{ \begin{array}{l} on(x, y, S_\alpha) \supset P(x, y), \\ P(x, y) \supset y = Shelf \wedge [Box = x \\ \quad \vee contains(Box, x, S_0)] \vee on(x, y, S_\alpha), \\ y = Shelf \wedge (x = Box \vee \\ \qquad contains(Box, x, S_0)) \supset \neg on(x, y, S_\alpha) \end{array} \right\}$$

Forgetting P' in T_0 by Theorem 1 yields a sentence

$$T_1 = \{\neg[P(Box, Shelf) \wedge contains(Box, x, S_0)] \wedge (broken(x, S_\alpha) \supset \neg\exists y.contains(y, x))\}.$$

By Prop. 2 and Lemma 1, it can be rewritten to be semi-definitional wrt P. Let $Re(T_1)$ be the result of rewriting T_1. Now, deleting all sentences with P' in $T_0 \cup Re(T_1)$, forgetting P, and replacing S_0 by S_α, we have (with simplifications)

$$\left\{ \begin{array}{l} contains(Box, Vase, S_\alpha), \\ (x = Box \vee contains(Box, x, S_\alpha)) \\ \qquad \supset \neg on(x, Shelf, S_\alpha), \\ contains(Box, x, S_\alpha) \supset broken(x, S_\alpha), \\ broken(x, S_\alpha) \supset (contains(Box, x, S_\alpha) \vee \\ \qquad \neg\exists y.contains(y, x, S_\alpha)) \end{array} \right\}$$

That is, Box still contains $Vase$, yet Box and all things contained in it are no longer on $Shelf$. Furthermore, all things contained in Box are broken, and all broken objects are either contained in Box, or were among the previously broken objects not contained in anything.

We note that relaxing $\mathcal{D}_{S_0}$ from separable semi-definitional to semi-definitional would cause troubles. In the above example, suppose another binary fluent $F(x, y)$ is affected globally by the action $\alpha = drop(Box, Shelf)$. Let $\gamma^+_F(\alpha, x, y, s)$ be $on(Box, Shelf)$ and $\gamma^-_F(\alpha, x, y, s)$ be FALSE. Consider $\mathcal{D}_{S_0} = \{F(x, y, S_0) \supset on(x, y, S_0)\}$, which is semi-definitional wrt F and on. The formula of Eq. (4b) obtained by forgetting the lifting predicate of F in the grounded theory amounts to $\neg on(Box, Shelf, S_0) \wedge F(x, y, S_\alpha) \supset on(x, y, S_0)$, which is no longer in good form wrt on.

For a similar reason, our approach cannot handle *non*-acyclic actions such as the action inc of incrementing a counter: The SSA $C(x, do(a, s)) \equiv a = inc \wedge C(x - 1, s) \vee C(x, s)$ would yield a self-loop $C \to C$ in the dependency graph.

Finally, requiring γ^+_F (also γ^-_F) to be in good form wrt *at most* one fluent in $\mathrm{NLE}(\alpha)$ means the dependent fluents appear essentially as literals. This seems to be inevitable as forgetting the lifting predicate of F generates formulas involving both γ^+_F and its negation (cf. Eq. (3c) and Eq. (4b)). It is desirable but also challenging to extend the result beyond formulas in good form. We leave it for future exploration.

4 Computability Results

So far, we have discussed progression in terms of FO-definability, but did not pay attention to computability. Interestingly, LL09 showed that for local-effect actions and normal actions, the progression of so-called proper$^+$ KBs [Lakemeyer and Levesque, 2002] is not just FO-definable, but also efficiently computable (polynomial in the size of KB) under reasonable assumptions. Here, we show that the progression of such KBs against acyclic actions is efficient as well, under similar but stronger assumptions.

Let e range over *ewffs*, i.e., quantifier-free formulas whose only predicate is equality. We let $\forall\phi$ denote the universal closure of ϕ.

Definition 10. Let e be an ewff and d a clause. Then a formula of the form $\forall(e \supset d)$ is called a $\forall$-*clause*. A KB is called *proper*$^+$ if it is a finite non-empty set of $\forall$-clauses.

The class of proper$^+$ KBs is very expressive. Similar to how a standard relational database corresponds to a maximally consistent and finite set of ground literals, in the sense

that every atom is known to be true iff stored in the database (and known to be false iff derivable by negation as failure), a proper$^+$ KB corresponds to a consistent but infinite set of ground clauses, not necessarily maximal. Besides, it is shown that for certain types of queries, query evaluation against proper$^+$ KB is efficient and can be computed in polynomial time [Liu and Levesque, 2005].

Theorem 4 ([Liu and Lakemeyer, 2009]). *Let Σ be a proper$^+$ KB, and $P(\vec{c})$ a ground atom. Then the result of forgetting $P(\vec{c})$ in Σ is definable as a proper$^+$ KB and can be computed in $O(n + 4^w m^2)$ time, where n is the size of Σ, m is the size of sentences in Σ where P appears, and w is the maximum number of appearances of P in a sentence of Σ.*

This is because every sentence in Σ can be transformed to a certain normal form wrt $P(\vec{c})$, i.e., a formula of the form $\phi_1 = \forall(e_1 \supset d_1 \vee P(\vec{t}))$ or $\phi_2 = \forall(e_2 \supset d_2 \vee \neg P(\vec{t}))$ where either $\vec{t} = \vec{c}$ or $e_1 \wedge \vec{t} = \vec{c}$ (likewise for e_2) is unsatisfiable. Moreover, this procedure takes $O(n + 2^w m)$ time [Liu and Lakemeyer, 2009]. In addition, forgetting $P(\vec{c})$ in the transformed proper$^+$ KB amounts to computing all the $\forall$-resolvents for any two input clauses: for ϕ_1 and ϕ_2 as above one obtains $\forall(e_1 \wedge e_2 \supset d_1 \vee d_2)$, yielding a KB that is again in proper$^+$ form. Hence the overall complexity is $O(n+4^w m^2)$.

In the same spirit, they showed:

Theorem 5 ([Liu and Lakemeyer, 2009]). *Let Σ be a proper$^+$ KB which is semi-definitional wrt predicate P. Then the result of forgetting P in Σ is definable as a proper$^+$ KB and can be computed in $O(n + m^2)$ time, where n is the size of Σ, and m is the size of sentences in Σ where P appears.*

In the above result, assuming $w = O(1)$ and $m^2 = n$, both forms of forgetting can be computed in $O(n)$ time, i.e. efficiently. Hence, LL09 had the following as a theorem. An SSA is said to be *essentially quantifier-free* if for each ground action α, by using $\mathcal{D}_{una}$, each of $\gamma^+_F(\vec{x}, \alpha, s)$ and $\gamma^-_F(\vec{x}, \alpha, s)$ can be simplified to a quantifier-free formula.

Theorem 6 ([Liu and Lakemeyer, 2009]). *Suppose that $\mathcal{D}_{ss}$ is essentially quantifier-free, α is a normal action, and $\mathcal{D}_{S_0}$ is a proper$^+$ KB which is normal wrt α. Then progression of $\mathcal{D}_{S_0}$ wrt α is definable as a proper$^+$ KB and can be efficiently computed.*

This is by the fact that if $\mathcal{D}_{ss}$ is essentially quantifier-free, then $\mathcal{D}_{ss}[\alpha, S_0]$ is definable as a proper$^+$ KB. Thereafter, one can use Theorem 5 to efficiently forget non-local-effect fluents and Theorem 4 to efficiently forget local-effect fluents via forgetting the characteristic set.

An implicit assumption of the above result is that after forgetting a non-local-effect fluent, the number of sentences involving other non-local-effect fluents in the resulting KB does not increase so that the process can be iterated to forget other non-local-effect fluents. This is indeed the case for normal actions where non-local-effect fluents only depend on local-effect ones. The situation differs for acyclic actions. For instance, in Example 2 forgetting the lifting predicate P' of $broken$ will generate a formula that involves the lifting predicate P for on, i.e., T_1.

Lemma 2. *If $\phi(\vec{x})$ is in good form wrt a predicate P and quantifier-free, then $\phi(\vec{x})$ can be transformed into a set of $\forall$-clauses that is semi-definitional wrt P.*

Proof. $\forall\vec{x}(\psi(\vec{x}) \vee P(\vec{t}))$ can equivalently be rewritten as $\forall\vec{x}\forall\vec{y}.\ \vec{y} = \vec{t} \supset (\psi(\vec{x}) \vee P(\vec{y}))$, which has the right form if w.l.o.g. ψ is in clausal form (and quantifier-free by assumption). Similar for $\forall\vec{x}(\psi'(\vec{x}) \vee \neg P(\vec{t}))$ and $\forall\vec{x}\psi''(\vec{x})$. □

The lemma implies that although new formulas might be generated by forgetting a non-local-effect fluent for an acyclic action, the new formulas can be rewritten as a proper$^+$ KB that is semi-definitional wrt the non-local-effect fluent it might depend on. This is crucial for the progression to iteratively forget multiple fluents.

Besides, the number of generated formulas that involve a non-local-effect fluent is linear in the size of sentences that involve the forgotten fluent (the generated sentences in Eq. (4b) and Eq. (4c) need to consider all the sufficient and necessary conditions). In addition, these formulas (conditions) will accumulate with the increase of the depth of a non-local-effect fluent in the graph. Namely, forgetting a non-local-effect fluent P with maximum depth in the dependency graph takes time $O(l^2)$ where l is the size of all sentences in the initial KB that involve fluents that have a path to P, including P itself. As a special case, for normal actions, we have $l = m$, where m is the size of sentences in the KB that involve P. Therefore, assuming $l^2 = n$ where n is the size of KB, we have:

Theorem 7. *Suppose that $\mathcal{D}_{ss}$ is essentially quantifier-free, α is an acyclic action, and $\mathcal{D}_{S_0}$ is a proper$^+$ KB which is seperably semi-definitional wrt $\mathrm{NLE}(\alpha)$. Then the progression of $\mathcal{D}_{S_0}$ wrt α is definable as a proper$^+$ KB and can be efficiently computed.*

Example 3. In Example 2, $\mathcal{D}_{ss}$ is essentially quantifier-free. Moreover, consider the ground action $\alpha = drop(Box, Shelf)$ and $\mathcal{D}_{S_0}$ as follows:

$$\left\{ \begin{array}{l} contains(Box, Vase, S_0), \\ x = Box \wedge y = Shelf \supset on(x, y, S_0), \\ \neg broken(x, S_0) \vee contains(y, x, S_0) \end{array} \right\}$$

Clearly, $\mathcal{D}_{S_0}$ is a proper$^+$ KB and is semi-definitional wrt $\mathrm{NLE}(\alpha)$ (it is equivalent to the initial theory before).

Now, applying Prop. 1 to $\mathcal{D}_0 \cup \mathcal{D}_{ss}[\alpha, S_0] \uparrow S_0$, we obtain the following set of $\forall$-clauses (marked as T'_0):

$$\left\{ \begin{array}{l} contains(Box, Vase, S_0), \\ x = Box \wedge y = Shelf \supset P(x, y), \\ \neg P'(x) \vee contains(y, x, S_0) \end{array} \right\} \cup$$

$$\left\{ \begin{array}{l} P(Box, Shelf) \vee \neg broken(x, S_\alpha) \vee P'(x), \\ contains(Box, x, S_0) \vee \neg broken(x, S_\alpha) \vee P'(x), \\ \neg P'(x) \vee broken(x, S_\alpha), \neg P(Box, Shelf) \vee \\ \neg contains(Box, x, S_0) \vee broken(x, S_\alpha) \end{array} \right\} \cup$$

$$\left\{ \begin{array}{l} \neg on(x, y, S_\alpha) \vee P(x, y), \\ \neg P(x, y) \vee y = Shelf \wedge x = Box \vee on(x, y, S_\alpha), \\ \neg P(x, y) \vee contains(Box, x, S_0) \vee on(x, y, S_\alpha), \\ x = Box \wedge y = Shelf \supset \neg on(x, y, S_\alpha), \\ y \neq Shelf \vee \neg contains(Box, x, S_\alpha) \\ \qquad \vee \neg on(x, y, S_\alpha) \end{array} \right\}$$

Forgetting P' in T_0' by computing all $\forall$-resolvents yields the following set (marked as T_1')

$$\left\{ \begin{array}{l} contains(Box, x, S_\alpha) \vee \neg broken(x, S_\alpha) \\ \qquad\qquad \vee contains(y, x, S_0), \\ P(Box, Shelf) \vee \neg broken(x, S_\alpha) \\ \qquad\qquad \vee contains(y, x, S_0) \end{array} \right\}.$$

It is easy to check that T_1' is logically equivalent to T_1 from Example 2. By introducing new variables and renaming those in T_1', we rewrite the second $\forall$-clause as $x = Box \wedge y = Shelf \supset (P(x, y) \vee \neg broken(x', S_\alpha) \vee contains(y', x', S_0))$ so that T_1' is semi-definitional wrt P. Similarly, all sentences in T_0' mentioning $P(Box, Shelf)$ can be rewritten in this way.

Now, deleting all sentences in $T_0' \cup T_1'$ that mention P' and forgetting P, we obtain the following proper$^+$ KB, which is equivalent to the progression result from before.

$$\left\{ \begin{array}{l} contains(Box, Vase, S_\alpha), \neg on(Box, Shelf, S_\alpha) \\ \neg contains(Box, x, S_\alpha) \vee \neg on(x, Shelf, S_\alpha), \\ \neg contains(Box, x, S_\alpha) \vee broken(x, S_\alpha), \\ \neg broken(x, S_\alpha) \vee contains(Box, x, S_\alpha) \\ \qquad\qquad \vee contains(y, x, S_\alpha) \end{array} \right\}$$

5 Related Work

In their seminal work, Lin and Reiter [1997] provided a general account of progression, extending earlier concepts such as the update of a database by means of STRIPS operators, or Pednault's [1987; 1989] ADL formalism, where databases had to be logically complete theories. There, Lin and Reiter also showed that progression is always second-order definable and that theories exist whose progression is not representable by a finite first-order theory, and they conjecture that first-order logic is generally insufficient to capture progression, even when allowing for infinite theories. This conjecture was later proved to be true in [Vassos and Levesque, 2008], where it is shown that an infinite first-order theory is generally too weak to capture all entailments of a progressed theory. Since then, efforts have been made to identify fragments where progression is first-order definable.

Lin and Reiter [1997] showed that if the KB is relatively complete, i.e. for every sentence ϕ uniform in S_0, KB entails either ϕ or its negation, or, if the basic action theory is *context-free*, i.e. actions' effects are independent of situations, progression is FO definable in case the KB is separably semi-definitional wrt all fluents (although they did not use this terminology). Essentially, context-free actions are special cases of acyclic actions where the effect conditions γ_F^+ and γ_F^- do not contain fluents at all. Later, [Vassos *et al.*, 2008] proved that progression through local-effect actions is FO definable, yet no explicit form was provided until [Liu and Lakemeyer, 2009]. In [Liu and Lakemeyer, 2009], normal actions are examined, extending the concept of local-effect actions. However, as described in this paper, their result was limited as to how non-local-effect fluents might depend on one another, a limitation we overcome with the notion of acyclic theories. [Liu and Lakemeyer, 2009] also showed that if the KB is additionally in a certain form called *proper*$^+$, progression can be carried out efficiently. To capture domains that involve actions that may not be local-effect or normal, [Vassos *et al.*, 2009] studied range-restricted action theories, i.e., theories where action effects are "bounded" in a certain sense. One appealing aspect of such action theories compared to our acyclic ones is that they allow fluents to depend on themselves, yet additional constraints have to be imposed on the action theories and initial KB (different from our notion of separably semi-definitional) to ensure a first-order progression. The above results on local-effect, normal actions and range-restricted action theories were extended to cover functions in [Belle and Lakemeyer, 2011]. More recently, [De Giacomo *et al.*, 2016] showed that progression is FO definable for *bounded situation calculus action theories*, i.e., action theories where the number of fluent instances affected by actions is bounded by a given constant.

Since its inception by Lin and Reiter [1997], progression has seen many applications and implications. [Lakemeyer and Levesque, 2009] consider *only knowing* a KB and show that progressing it wrt an action and *sensing* result amounts to only knowing another KB. This result was extended to probabilistic knowledge bases for noisy actions and sensing in [Liu and Feng, 2023]. See also [Belle and Levesque, 2014; Belle and Levesque, 2020] for the progression of a probabilistic knowledge base in the classical Situation Calculus. Other works that involve progression include [Fang *et al.*, 2019] for multi-agent modal logic, [Schwering *et al.*, 2015; Claßen and Delgrande, 2022] for belief revision, and [Claßen, 2013; Liu *et al.*, 2023] for planning and verification in GOLOG. Notably, [Zarrieß and Claßen, 2016] introduced a concept of acyclic action theories that ours is inspired by, but that differs in its formal definition.[2] There, they were used to show that the verification of temporal properties for non-terminating GOLOG programs is decidable, under additional restrictions. Yet, their result is based on regression, and it was open whether (though plausible that) this implies a first-order result on progression. The result of acyclic actions provided in this paper contributes to completing the picture.

6 Conclusion

In this work, we extend the first-order progression of local-effect and normal actions by Liu and Lakemeyer [2009] and show that for acyclic actions that strictly subsume normal actions, progression is still first-order definable. Furthermore, we show that, for proper$^+$ KBs, under similar but more stringent assumptions than made by Liu and Lakemeyer, progression is efficient as well.

For future work, a possible direction is to explore whether weaker restrictions than being in good form can be applied. Besides, a well-known algorithm for second-order quantifier elimination [Gabbay *et al.*, 2008] is the DLS algorithm [Doherty *et al.*, 1997], which is proven to terminate if the dependency of SO variables is subject to certain acyclicity properties [Conradie, 2006]. It would be interesting to investigate how exactly the acyclicity there is connected to our notion of acyclicity of fluents for acyclic actions.

[2] [Zarrieß and Claßen, 2016] subdivides the formulas comprising γ_F^+ and γ_F^- into an *effect descriptor* and *context condition*. For the dependency graph, only effect descriptors are considered.

Acknowledgments

Daxin is funded by a Royal Society University Research Fellowship.

References

[Ackermann, 1935] Wilhelm Ackermann. Untersuchungen über das Eliminationsproblem der mathematischen Logik. *Mathematische Annalen*, 110(1):390–413, 1935.

[Belle and Lakemeyer, 2011] Vaishak Belle and Gerhard Lakemeyer. On progression and query evaluation in first-order knowledge bases with function symbols. In *IJCAI*, 2011.

[Belle and Levesque, 2014] Vaishak Belle and Hector Levesque. How to progress beliefs in continuous domains. In *KR*, 2014.

[Belle and Levesque, 2020] Vaishak Belle and Hector J. Levesque. Regression and progression in stochastic domains. *Artificial Intelligence*, 281:103247, 2020.

[Claßen and Delgrande, 2022] Jens Claßen and James P. Delgrande. Projection of belief in the presence of non-deterministic actions and fallible sensing. In *KR*, pages 400–404, 2022.

[Claßen and Zarrieß, 2017] Jens Claßen and Benjamin Zarrieß. Decidable verification of decision-theoretic Golog. In *Proceedings of the Eleventh International Symposium on Frontiers of Combining Systems (FroCoS 2017)*, volume 10483 of *LNCS*, pages 227–243. Springer, 2017.

[Claßen, 2013] Jens Claßen. *Planning and verification in the agent language Golog*. PhD thesis, RWTH Aachen University, 2013.

[Conradie, 2006] Willem Conradie. On the strength and scope of DLS. *Journal of Applied Non-Classical Logics*, 16(3-4):279–296, 2006.

[De Giacomo *et al.*, 2016] Giuseppe De Giacomo, Yves Lespérance, Fabio Patrizi, and Stavros Vassos. Progression and verification of situation calculus agents with bounded beliefs. *Studia Logica*, 104:705–739, 2016.

[Doherty *et al.*, 1997] Patrick Doherty, Witold Łukaszewicz, and Andrzej Szałas. Computing circumscription revisited: A reduction algorithm. *Journal of Automated Reasoning*, 18:297–336, 1997.

[Fang *et al.*, 2019] Liangda Fang, Yongmei Liu, and Hans Van Ditmarsch. Forgetting in multi-agent modal logics. *Artificial Intelligence*, 266:51–80, 2019.

[Fikes and Nilsson, 1971] Richard Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3/4):189–208, 1971.

[Gabbay *et al.*, 2008] Dov M. Gabbay, Renate A. Schmidt, and Andrzej Szalas. *Second-Order Quantifier Elimination. Foundations, Computational Aspects and Applications*. College Publications, 2008.

[Lakemeyer and Levesque, 2002] Gerhard Lakemeyer and Hector J. Levesque. Evaluation-based reasoning with disjunctive information in first-order knowledge bases. In *KR*, pages 73–81, 2002.

[Lakemeyer and Levesque, 2009] Gerhard Lakemeyer and Hector J. Levesque. A semantical account of progression in the presence of defaults. *Conceptual Modeling: Foundations and Applications: Essays in Honor of John Mylopoulos*, pages 82–98, 2009.

[Lin and Reiter, 1994] Fangzhen Lin and Ray Reiter. Forget it. In *Working Notes of AAAI Fall Symposium on Relevance*, pages 154–159, 1994.

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

[Liu and Feng, 2023] Daxin Liu and Qihui Feng. On the progression of belief. *Artificial Intelligence*, 322:103947, 2023.

[Liu and Lakemeyer, 2009] Yongmei Liu and Gerhard Lakemeyer. On first-order definability and computability of progression for local-effect actions and beyond. In *IJCAI*, pages 860–866, 2009.

[Liu and Levesque, 2005] Yongmei Liu and Hector J. Levesque. Tractable reasoning in first-order knowledge bases with disjunctive information. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 639. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[Liu *et al.*, 2023] Daxin Liu, Qinfei Huang, Vaishak Belle, and Gerhard Lakemeyer. Verifying belief-based programs via symbolic dynamic programming. In *ECAI 2023*, pages 1497–1504. IOS Press, 2023.

[McCarthy and Hayes, 1969] John McCarthy and Patrick Hayes. Some philosophical problems from the standpoint of artificial intelligence. In B. Meltzer and D. Michie, editors, *Machine Intelligence 4*, pages 463–502. American Elsevier, New York, 1969.

[Pednault, 1987] Edwin Peter Dawson Pednault. *Toward a mathematical theory of plan synthesis*. Stanford University, 1987.

[Pednault, 1989] Edwin PD Pednault. ADL: exploring the middle ground between STRIPS and the situation calculus. In *KR*, pages 324–332, 1989.

[Reiter, 1991] Raymond Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pages 359–380, 1991.

[Reiter, 2001] Raymond Reiter. *Knowledge in action: logical foundations for specifying and implementing dynamical systems*. MIT press, 2001.

[Schwering *et al.*, 2015] Christoph Schwering, Gerhard Lakemeyer, and Maurice Pagnucco. Belief revision and progression of knowledge bases in the epistemic situation calculus. In *IJCAI*, 2015.

[van Ditmarsch *et al.*, 2007] Hans P. van Ditmarsch, Andreas Herzig, and Tiago de Lima. Optimal regression for reasoning about knowledge and actions. In *AAAI*, pages 1070–1076. AAAI Press, 2007.

[Vassos and Levesque, 2008] Stavros Vassos and Hector J. Levesque. On the progression of situation calculus basic action theories: Resolving a 10-year-old conjecture. In *AAAI*, pages 1004–1009, 2008.

[Vassos and Levesque, 2013] Stavros Vassos and Hector J. Levesque. How to progress a database III. *Artificial Intelligence*, 195:203–221, 2013.

[Vassos and Patrizi, 2013] Stavros Vassos and Fabio Patrizi. A classification of first-order progressable action theories in situation calculus. In *IJCAI*, pages 1132–1138, 2013.

[Vassos *et al.*, 2008] Stavros Vassos, Gerhard Lakemeyer, and Hector J. Levesque. First-order strong progression for local-effect basic action theories. In *KR*, pages 662–672, 2008.

[Vassos *et al.*, 2009] Stavros Vassos, Stavros Sardina, and Hector Levesque. Progressing basic action theories with non-local effect actions. In *Ninth International Symposium on Logical Formalizations of Commonsense Reasoning*, pages 135–140, 2009.

[Zarrieß and Claßen, 2016] Benjamin Zarrieß and Jens Claßen. Decidable verification of Golog programs over non-local effect actions. In *AAAI*, 2016.