1

A Database-type Approach for Progressing Action Theories with Bounded Effects

STAVROS VASSOS AND SEBASTIAN SARDINA

ABSTRACT. In this paper we study the progression of situation calculus action theories that are able to handle a class of actions that, while extremely simple conceptually and common in many settings, cannot be handled by previous approaches. Specifically, based on the notion of *safe-range queries* from database theory and *just-in-time* action histories, we present a new type of action theories that ensures that actions have bounded effects over a restricted range of objects. Such theories may represent incomplete information and can be progressed by directly updating the knowledge base in an algorithmic manner.

We dedicate this paper to Hector Levesque, who as much as anyone, promoted the logic-based approach to problems and the fundamental insight that expressivity can be traded for efficiency. Both aspects are strongly reflected in this work. This paper is an extended version of a recent result that appeared in [Vassos, Sardina, and Levesque 2009]. As such, the work we present here has been greatly influenced by Hector, both in terms of the specific research results reported as well as in the fact that it involves the contribution of two generations of students under his supervision. Indeed, we feel privileged to have had Hector as our Ph.D. supervisor. His remarkable qualities, both intellectual and personal, have had major impact on our careers in a way that is impossible to put it in text. Just one thing we can say: Thanks Hector!

1 Introduction

One of the requirements for building agents with a pro-active behavior is the ability to reason about action and change. The ability to *predict* how the world will be after performing a sequence of actions is the basis for offline automated planning, scheduling, web-service composition, etc. In the situation calculus [McCarthy and Hayes 1969; Reiter 2001] such reasoning problems are examined in the context of the so-called basic action theories (BATs). These are logical theories that specify the preconditions and effects of actions, and an initial knowledge base (KB) that represents the initial state of the world before any action has occurred.

A basic action theory can be used to solve offline problems as well as to equip a situated agent with the ability to *keep track* of the current state of the world. As a theory is a static entity, in the sense that the axioms do not change over time, reasoning about the current state is typically carried over using techniques based on *regression* that transform queries about the future into queries about the initial state [Reiter 2001]. This is an effective choice for some applications, but a poor one for many settings where an agent may act autonomously for long periods of time. In those cases, it is mandatory that the theory be (periodically) updated so that the initial KB be replaced by a new one reflecting the changes due to the actions that have already occurred. This is identified as the problem of *progression* for basic action theories [Lin and Reiter 1997].

In general, a KB in a basic action theory is an unrestricted first-order logical theory that offers great flexibility and expressiveness. The price to pay though is high, as it is hard to find practical solutions for the related reasoning problems. As far as progression is concerned, it was shown by Lin and Reiter [1997] that the updated KB requires second-order logic in the general case. For this reason, several restrictions on the theories have been proposed so that the updated KB is first-order representable. In particular, two recent results show that progression is practical provided that actions are *local-effect* [Vassos, Lakemeyer, and Levesque 2008] or *normal* [Liu and Lakemeyer 2009].

The local-effect assumption essentially means that all the objects that may be affected by the action are directly specified by the arguments of the action. For example, an action that results in the robot moving from location l_1 to location l_2 needs to explicitly mention l_1 and l_2 in its arguments in order to be local-effect, e.g., $\mathit{move}(l_1, l_2)$. On the other hand, a so-called normal action is more general in that it may also affect objects that are not included as arguments, as long as these are specified by information that exists in the KB in a particular way. In this manner, the above moving action may be allowed also to affect the location of objects being held by the robot.

Nonetheless, it turns out that there are many simple actions that do not qualify as local-effect or normal. For example, action *moveFwd* (with no arguments) that causes a robot to move forward by a fixed length is neither local-effect nor normal. These actions arise naturally in many robotic domains and grid-style games, for instance, and cannot be handled by previous techniques for progression unless further restrictions are assumed (e.g., a finite domain). Furthermore, these actions cannot, in general, be reformulated into a well-behaved local-effect version that includes all the necessary ground arguments as the required information, for example, the robot's current location, may be unknown.

In order to handle such actions, we present what we call *range-restricted* basic action theories (RR-BATs). These theories are able to represent actions that may not be local-effect or normal but whose (range of) effects can be "bounded." For such theories, we describe a method for progression such that the new KB is first-order and finite, and we prove that the method is logically correct and effectively computed via database-like evaluation for a special case of interest. To the best of our knowledge, it is the first result that accounts for a possibly infinite domain, incomplete information, sensing, and a particular class of simple actions that goes beyond the local-effect assumption.

The rest of the paper is organized as follows. In Section 2, we review the situation calculus and the progression task for unrestricted basic action theories. Then, in Section 3, we specify the structure for the theories of action and queries to be considered, and define the notion of *possible answers* for queries. In Section 4, we present a general method for progression for these theories and examine the complexity of the approach in a special case. We conclude by discussing related and future work, and drawing final conclusions.

2 Formal preliminaries

The language $\mathcal{L}$ of the situation calculus as presented by Reiter [2001] is a three-sorted first-order logic language with equality and some limited second-order features. The three sorts are the following: *action*, *situation*, and a catch-all sort *object* for everything else depending on the domain of application.

Similar to a normal one-sorted first-order language, $\mathcal{L}$ includes function and predicate symbols. In this case since there are three sorts, each of the symbols has a type that specifies the sorts for the arguments it takes. The situation calculus includes symbols only of certain types each of which has a special role in the representation of the world and its dynamics.

An action term or simply an *action* represents an atomic action that may be performed in the world. For example consider the action $\mathit{move}(l_1, l_2)$ that may be used to represent that a robot moves from location l_1 to location l_2. A situation term or simply a *situation* represents a world history as a sequence of actions. The constant S_0 is used to denote the *initial situation* where no actions have occurred. Sequences of actions are built using the function symbol *do*, such that $do(\alpha, \sigma)$ represents the successor situation resulting from performing action α in situation σ.

A *relational fluent* is a predicate whose last argument is a situation, and thus whose truth value can change from situation to situation. For example, $\mathit{RobotAt}(l, \sigma)$ may be used to represent that the robot lies at location l in situation σ. In order to simplify the analysis we have restricted the language $\mathcal{L}$ so that there are no functional fluent symbols in $\mathcal{L}$, that is, functions whose last argument is a situation. This is not a restriction on the expressiveness of $\mathcal{L}$ as functional fluents can be represented by relational fluents with a few extra axioms.

Actions need not be executable in all situations, and the predicate atom $\mathit{Poss}(\alpha, \sigma)$ states that action α is executable in situation σ. For example, $\mathit{Poss}(\mathit{move}(l_1, l_2), \sigma)$ is intended to represent that the action $\mathit{move}(l_1, l_2)$ can be executed in situation σ. Actions may also have a sensing result: the special function $\mathit{SR}(\alpha, \sigma)$ denotes the sensing outcome of action α when executed in situation σ [Scherl and Levesque 2003].

In this paper, we shall restrict our attention to a language $\mathcal{L}$ with a finite number of relational fluent symbols that only take arguments of sort object (apart their last situation argument), an infinite number of constant symbols of sort object, and a finite number of function symbols of sort action that take arguments of sort object. We adopt the following notation with subscripts and superscripts: α and a for terms and variables of sort action; σ and s for terms and variables of sort situation; t and x, y, z, w for terms and variables of sort object. Also, we use A for action function symbols, F, G for fluent symbols, and b, c, d, e, o for constants of sort object. Finally, we will typically write $\phi(\vec{x})$ to state that the free variables of the formula are among $\vec{x}$.

The well-formed first-order formulas of $\mathcal{L}$ are defined inductively similarly to a normal one-sorted language but also respecting that each parameter has a unique sort. As far as the second-order formulas of $\mathcal{L}$ are concerned, only quantification over relations is allowed and the well-formed formulas are defined inductively similarly to a normal second-order language.

Often we will focus on sentences that refer to a particular situation. For this purpose, for any situation term σ, we define the set of *uniform formulas in* σ to be all those (first-order or second-order) formulas in $\mathcal{L}$ that do not mention any other situation terms except for σ, do not mention *Poss*, and where σ is not used by any quantifier [Lin and Reiter 1997].

2.1 Basic action theories

Within the language $\mathcal{L}$, one can formulate action theories that describe how the world changes as the result of the available actions. We focus on a variant of the *basic action theories (BATs)* [Reiter 2001] of the following form:[1]

[1]For legibility, we typically omit leading universal quantifiers. The difference with Reiter's BATs is the incorporation of $\mathcal{D}_{sr}$ for sensing and the set $\mathcal{E}$.

$$\mathcal{D} = \mathcal{D}_{ap} \cup \mathcal{D}_{ss} \cup \mathcal{D}_{una} \cup \mathcal{D}_{sr} \cup \mathcal{D}_0 \cup \Sigma \cup \mathcal{E},$$

where:

1. $\mathcal{D}_{ap}$ is the set of action precondition axioms (PAs), one per action symbol A, of the form $\mathit{Poss}(A(\vec{y}), s) \equiv \Pi_A(\vec{y}, s)$, where $\Pi_A(\vec{y}, s)$ is first-order and uniform in s. PAs characterize the conditions under which actions are physically possible.
2. $\mathcal{D}_{ss}$ is the set of successor state axioms (SSAs), one per fluent symbol F, of the form $F(\vec{x}, do(a, s)) \equiv \Phi_F(\vec{x}, a, s)$, where $\Phi_F(\vec{x}, a, s)$ is first-order and uniform in s. SSAs describe how fluents change between situations as the result of actions.
3. $\mathcal{D}_{sr}$ is the set of sensing-result axioms (SRAs), one for each action symbol A, of the form $\mathit{SR}(A(\vec{y}), s) = r \equiv \Theta_A(\vec{y}, r, s)$, where $\Theta_A(\vec{y}, r, s)$ is first-order and uniform in s. SRAs relate sensing outcomes with fluents, or more generally, with complex properties of the domain.
4. $\mathcal{D}_{una}$ is the set of unique-names axioms for actions.
5. $\mathcal{D}_0$, the *initial knowledge base (KB)*, is a set of first-order sentences uniform in S_0 describing the initial situation S_0.
6. Σ is the set of domain independent axioms of the situation calculus, formally defining the legal situations. A second-order induction axiom is included in Σ.
7. $\mathcal{E}$ is an infinite set of unique-names axioms for object constants.

Probably the most interesting component of a basic action theory is the set successor state axioms, which together encode the dynamics of the domain being represented. Technically, SSAs are meant to capture the effects and non-effects of actions. To achieve that in a parsimonious way, one typically follows Reiter [1991]'s well-known solution to the frame problem and writes successor state axioms of the following form:

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

where both $\gamma_F^+(\vec{x}, a, s)$ and $\gamma_F^+(\vec{x}, a, s)$ are first-order formulas uniform in s encoding the positive and negative effects, respectively, of action a on fluent F at situation s.

As a running example we consider the simple scenario of a robot that is capable of navigating and moving objects around in a grid-like world while some objects such as boxes may also contain other objects. The robot is equipped with a basic action theory $\mathcal{D}$ that captures the initial state and the dynamics of the example domain. $\mathcal{D}$ uses fluents $\mathit{RobotAt}(x, s)$ and $\mathit{RobotDir}(x, s)$ to represent information about the location of the robot and the direction it is facing, and $\mathit{Connected}(x_1, x_2, x_3, s)$ to represent that location x_3 is adjacent to location x_1 wrt direction x_2 in s.

$\mathcal{D}$ includes the following SSA to capture the way fluent $\mathit{RobotAt}(x, s)$ is affected by the actions of the robot:

$$\mathit{RobotAt}(x, do(a, s)) \equiv \gamma_{\mathit{RobotAt}}^+(x, a, s) \vee \mathit{RobotAt}(x, s) \wedge \neg\gamma_{\mathit{RobotAt}}^-(x, a, s), \quad (1)$$

where $\gamma_{\mathit{RobotAt}}^+(x, a, s)$ is the formula

$$\exists z_1, z_2.\ a = \mathit{moveFwd} \wedge \mathit{RobotAt}(z_1, s) \wedge \mathit{RobotDir}(z_2, s) \wedge \mathit{Connected}(z_1, z_2, x, s),$$

and $\gamma^-_{RobotAt}(x,a,s)$ is the formula $a = moveFwd$.

In words, the robot is at location x after executing action a iff a is the action of moving forward, the robot is currently at location z_1 and facing towards direction z_2, and x is the next adjacent location to z_1 towards direction z_2, or the robot was already in location x (i.e., $RobotAt(x,s)$ holds) and it has not performed a move action.

The current location of objects is modeled using fluent $At(x_1,x_2,s)$: object x_1 is at location x_2 at situation s, and fluent $In(x_1,x_2,s)$ represents that object x_2 is inside x_1. When the robot moves, all objects being carried by the robot move as well. This is accounted in the following SSA for fluent $At(x_1,x_2,s)$ that is included in $\mathcal{D}$:

$$At(x_1,x_2,do(a,s)) \equiv \gamma^+_{At}(x_1,x_2,a,s) \lor At(x_1,x_2,s) \land \neg\gamma^-_{At}(x_1,x_2,a,s), \qquad (2)$$

where $\gamma^+_{At}(x_1,x_2,a,s)$ is the formula

$$\begin{aligned}&\exists z_1,z_2\big(a = moveFwd \land RobotAt(z_1,s) \land RobotDir(z_2,s) \land\\ &\qquad Connected(z_1,z_2,x_2,s) \land Holding(x_1,s)\big) \lor\\ &\exists z_1,z_2,z_3\big(a = moveFwd \land RobotAt(z_1,s) \land RobotDir(z_2,s) \land\\ &\qquad Connected(z_1,z_2,x_2,s) \land Holding(z_3,s) \land In(z_3,x_1,s)\big),\end{aligned}$$

and $\gamma^-_{At}(x_1,x_2,a,s)$ is the formula

$$\begin{aligned}&a = moveFwd \land Holding(x_1,s) \lor\\ &\exists z.\ a = moveFwd \land Holding(z,s) \land In(z,x_1,s).\end{aligned}$$

Formula $\gamma^+_{At}(x_1,x_2,a,s)$ that describes the positive effects, states that object x_1 is at location x_2 if the robot has successfully moved to location x_2 and either the robot is holding x_1 or x_1 is inside an object that the robot is holding. Similarly, the negative effect formula $\gamma^-_{At}(x_1,x_2,a,s)$ in the SSA states that object x_1 is not (anymore) at location x_2 if the robot has moved while carrying x_1 or some other object that has x inside it.

Although we do not list them here, fluents $RobotDir(x,w)$, $Holding(x,s)$, $In(x_1,x_2,s)$, and $Connected(x_1,x_2,x_3,s)$ are also meant to have their corresponding SSAs in basic action theory $\mathcal{D}$ of our running example. For instance, the SSA for $RobotDir(x,s)$ states that the current direction the robot is facing is affected by the turning action that the robot can perform. Similarly, appropriate action precondition axioms are assumed.

Although the agent can pick up containers that have other objects inside, it may have at certain point incomplete information on what is inside the containers, that is, incomplete information about fluent $In(x_1,x_2,s)$. Nonetheless, we assume the agent can sense, using a special device, whether there is gold inside a container object. Technically, the agent can execute sensing action $senseGold(y)$ that determines whether there is gold inside container object y:

$$\begin{aligned}&Poss(senseGold(y),s) \equiv Holding(y,s),\\ &SR(senseGold(y),s) = r \equiv\ [(r=1) \equiv In(y,item_2,s)].\end{aligned}$$

In words, the agent can sense for gold in an object whenever the agent is holding the object, and the outcome of such action is 1 iff the container object in question contains gold pieces inside.

Finally, the initial KB $\mathcal{D}_0$, represents the initial knowledge the agent has about the world using sentences uniform in S_0, that is, sentences that only refer to the initial situation such as $RobotAt(loc_1,S_0)$ and $RobotDir(north,S_0)$. We postpone the specification of $\mathcal{D}_0$ for our example until Section 3.1 where we introduce a specific form of initial KBs.

2.2 Progression

The progression of a basic action theory is the problem of *updating* the initial KB so that it reflects the current state of the world after some actions have been performed instead of the initial state of the world. In other words, in order to do a one-step progression of a basic action theory $\mathcal{D}$ with respect to a ground action α, we are to replace $\mathcal{D}_0$ in $\mathcal{D}$ by a suitable set $\mathcal{D}_\alpha$ of sentences (i.e., a new KB), so that the original theory $\mathcal{D}$ and the theory $(\mathcal{D} - \mathcal{D}_0) \cup \mathcal{D}_\alpha$ are equivalent with respect to how they describe the situation $do(\alpha, S_0)$ as well as all situations in the future of $do(\alpha, S_0)$ [Lin and Reiter 1997].

In this paper, instead of the model-theoretic definition of progression of Lin and Reiter [1997], we follow the definition of the so-called *strong progression* of Vassos et al. [2008] which we extend slightly in order to account for sensing actions.

Let $\mathcal{D}$ be a basic action theory over relational fluents $F_1, \ldots, F_n$, and let $Q_1, \ldots, Q_n$ be second-order predicate variables. For any formula ϕ in $\mathcal{L}$, let $\phi\langle \vec{F} : \vec{Q} \rangle$ be the formula that results from replacing any fluent atom $F_i(t_1, \ldots, t_n, \sigma)$ in ϕ, where σ is a situation term, with atom $Q_i(t_1, \ldots, t_n)$.

DEFINITION 1. Let $\mathcal{D}$ be a basic action theory over fluents $\vec{F}$ such that $\mathcal{D}_0$ is finite, α a ground action term of the form $A(\vec{c})$, and d a sensing outcome result. Let $Prog(\mathcal{D}, \alpha, d)$ be the following second-order sentence uniform in $do(\alpha, S_0)$:

$$\exists \vec{Q}.\ \mathcal{D}_0\langle \vec{F} : \vec{Q} \rangle \wedge \Theta_A(\vec{c}, d, do(\alpha, S_0)) \ \wedge \bigwedge_{i=1}^{n} \forall \vec{x}.\ F_i(\vec{x}, do(\alpha, S_0)) \ \equiv \\ \big(\Phi_{F_i}(\vec{x}, \alpha, S_0)\langle \vec{F} : \vec{Q} \rangle\big).$$

Then, we say that a set of formulas $\mathcal{D}_\alpha$ uniform in situation $do(\alpha, S_0)$ is a *strong progression* of $\mathcal{D}$ wrt pair of action-outcome (α, d) iff $\mathcal{D}_\alpha \cup \mathcal{D}_{una} \cup \mathcal{E}$ is logically equivalent[2] to $\{Prog(\mathcal{D}, \alpha, d)\} \cup \mathcal{D}_{una} \cup \mathcal{E}$. $\square$

Informally, set $\mathcal{D}_\alpha$ represents the updated initial KB after action α has been executed with sensing outcome d. Although $Prog(\mathcal{D}, \alpha, d)$ is defined in second-order logic we are interested in cases where we can find a $\mathcal{D}_\alpha$ that is *first-order representable*. In the sequel we shall present some restrictions that are sufficient conditions for doing this as well as a method for computing a finite $\mathcal{D}_\alpha$ for a special case of interest.

3 Range-restricted basic action theories

In this section, we present a new type of basic action theories in which the initial KB is a *database of possible closures* and the axioms in $\mathcal{D}_{ap}$, $\mathcal{D}_{ss}$, and $\mathcal{D}_{sr}$ are *range-restricted*. Essentially, we will be specifying the effects of actions in a way that resembles to logic-programming. In order to make progression work in first-order we also require a semantic assumption that may not hold in all situations but may be enforced with an appropriate account of sensing. Under these three assumptions, we will show later that a finite first-order updated (initial) KB can be computed.

3.1 A database of possible closures

Intuitively, we treat each fluent as a *multi-valued function*, where the last argument of sort object is considered as the "*output*" and the rest of the arguments of sort object as the

[2]Whenever we say that two formulas are logically equivalent we assume that the logical symbol $=$ is always interpreted as the true identity.

"*input*" of the function.[3] This distinction is important as we require that $\mathcal{D}_0$ expresses incomplete information only about the output of fluents.

DEFINITION 2. Let V be a set of constants and τ a fluent atom of the form $F(\vec{c}, w, S_0)$, where $\vec{c}$ is a vector of constants and w a variable. We say that τ has the *ground input* $\vec{c}$ and the *output* w. The *atomic closure* χ of τ on V is the sentence:

$$\forall w. F(\vec{c}, w, S_0) \equiv \bigvee_{e \in V} (w = e).$$

The *closure* of the fluent vector $\vec{\tau} = \langle \tau_1, \ldots, \tau_n \rangle$ of distinct atoms on a ground input vector $\vec{V} = \langle V_1, \ldots, V_n \rangle$ of sets of constants is the conjunction $(\chi_1 \wedge \ldots \wedge \chi_n)$, where each χ_i is the atomic closure of τ_i on V_i. □

A closure of $\vec{\tau}$ expresses complete information about the output of all input-grounded fluents in $\vec{\tau}$. For example, consider fluents *RobotAt*(x, s) and *RobotDir*(x, s) that represent information about the location of the robot and the direction it is facing. Let χ_1 be the the atomic closure of *RobotAt*(w, S_0) on $\{loc_1\}$ and χ_2 the atomic closure of *RobotDir*(w, S_0) on $\{north\}$, that is:

$$\forall w. \textit{RobotAt}(w, S_0) \equiv (w = \textit{loc}_1); \qquad (\chi_1)$$

$$\forall w. \textit{RobotDir}(w, S_0) \equiv (w = \textit{north}). \qquad (\chi_2)$$

Then χ_1 and χ_2 express complete information about the location of the robot and the direction it is facing.

Consider now fluent *In*(x, y, s) which represents that object y is inside the object x at situation s, and the input-grounded fluent atom *In*(box_1, w, S_0). Let χ_3 be the following sentence:

$$\forall w. \textit{In}(\textit{box}_1, w, S_0) \equiv (w = \textit{item}_1 \vee w = \textit{item}_2). \qquad (\chi_3)$$

Then, χ_3 is the atomic closure of *In*(box_1, w, S_0) on $\{item_1, item_2\}$ which states that there are *exactly* two objects inside *box*$_1$, namely *item*$_1$ and *item*$_2$.

Finally, let χ_4 be the the atomic closure of *In*(box_2, w, S_0) on $\{gold\}$:

$$\forall w. \textit{In}(\textit{box}_2, w, S_0) \equiv (w = \textit{gold}). \qquad (\chi_4)$$

Then, $(\chi_3 \wedge \chi_4)$ is a (non-atomic) closure of the vector of input-grounded fluent atoms

$$\big\langle \textit{In}(\textit{box}_1, w, S_0), \textit{In}(\textit{box}_2, w, S_0) \big\rangle$$

that expresses complete information about the contents of *box*$_1$ and *box*$_2$.

We now show how we can combine these closure statements in order to express incomplete information.

DEFINITION 3. A *possible closures axiom (PCA)* for a vector of input-grounded fluents $\vec{\tau}$ is a disjunction of the form $(\chi_1 \vee \cdots \vee \chi_n)$, where each χ_i is a closure of $\vec{\tau}$ on a distinct vector $\vec{V_n}$ of constants. □

A PCA for $\vec{\tau}$ expresses *disjunctive information* about the output of all fluents in $\vec{\tau}$, by stating how such outputs can be combined together (in n possible ways). For example

[3]This is similar to *modes* in logic programming [Apt and Pellegrini 1994] and it can be easily generalized to multiple outputs.

consider again the input-grounded fluent atom $\mathit{In}(\mathit{box}_1, w, S_0)$, and let χ_5 be the closure of $\mathit{In}(\mathit{box}_1, w, S_0)$ on $\{\mathit{item}_1, \mathit{gold}\}$:

$$\forall w. \mathit{In}(\mathit{box}_1, w, S_0) \equiv (w = \mathit{item}_1 \vee w = \mathit{gold}). \qquad (\chi_5)$$

Then, $(\chi_3 \vee \chi_5)$ is a PCA for $\mathit{In}(\mathit{box}_1, w, S_0)$ stating that there are exactly two objects inside box_1, one being item_1 and the other being *either* item_2 *or gold.* Similarly, let χ_6 be the closure of $\mathit{In}(\mathit{box}_2, w, S_0)$ on $\{\mathit{item}_2\}$:

$$\forall w. \mathit{In}(\mathit{box}_2, w, S_0) \equiv (w = \mathit{item}_2). \qquad (\chi_6)$$

Then $(\chi_3 \wedge \chi_4) \vee (\chi_5 \wedge \chi_6)$ is a PCA for

$$\big\langle \mathit{In}(\mathit{box}_1, w, S_0,), \mathit{In}(\mathit{box}_2, w, S_0)\big\rangle,$$

which states that box_1 and box_2 contain the three items $\mathit{item}_1, \mathit{item}_2, \mathit{gold}$, but only two possible combinations are allowed: either *gold* is in box_2 and the other two items are in box_1, or item_2 is in box_2 and the other two items are in box_1.

Using a set of PCAs, each one referring to different input-grounded fluent atoms, we are now able to define the form of our initial knowledge bases.

DEFINITION 4. A *database of possible closures (DBPC)* is a set $\mathcal{D}_0 = \{\Xi^{\vec{\tau}_1}, \ldots, \Xi^{\vec{\tau}_\ell}\}$, where each $\Xi^{\vec{\tau}_i}$ is a PCA for $\vec{\tau}_i$ such that $\vec{\tau}_i \cap \vec{\tau}_j = \emptyset$, for all distinct $i, j \in \{1, \ldots, \ell\}$. For every i, each disjunct of $\Xi^{\vec{\tau}_i}$ is called a *possible closure* wrt $\mathcal{D}_0$. $\square$

So, for every fluent atom τ with a ground input (e.g., $\mathit{In}(\mathit{box}_1, w, S_0)$), either the output of τ is completely unknown in S_0 (i.e., τ is not mentioned in $\mathcal{D}_0$) or there is just one PCA $\Xi_{\vec{\tau}_i}$ (with $\tau \in \vec{\tau}_i$) that specifies its output value in several possible "worlds" (one per disjunct in the PCA).

For our running example, the initial knowledge base is as follows:

$$\mathcal{D}_0 = \{\chi_1, \chi_2, (\chi_3 \wedge \chi_4) \vee (\chi_5 \wedge \chi_6)\} \cup \{\chi_7, \ldots, \chi_{13}\},$$

where $\chi_1, \ldots, \chi_6$ are defined above, χ_7 is the closure of $\mathit{Connected}(\mathit{loc}_1, \mathit{north}, w, S_0)$ on $\{\mathit{loc}_2\}$, χ_8 is the closure of $\mathit{Holding}(w, S_0)$ on the set $\{\mathit{box}_2\}$, and $\chi_9, \ldots, \chi_{13}$ are the atomic closures of $\mathit{At}(\mathit{box}_1, w, S_0), \mathit{At}(\mathit{item}_1, w, S_0), \mathit{At}(\mathit{box}_2, w, S_0), \mathit{At}(\mathit{item}_2, w, S_0)$, and $\mathit{At}(\mathit{gold}, w, S_0)$ on set $\{\mathit{loc}_1\}$, respectively. Note that each of the ten sentences in $\mathcal{D}_0$ is a PCA, and each of χ_1, χ_2, $\chi_3 \wedge \chi_4$, $\chi_5 \wedge \chi_6$, $\chi_7, \ldots, \chi_{13}$ is a possible closure wrt $\mathcal{D}_0$.

We now turn our attention to the so-called *possible answers* to a formula wrt a KB.

DEFINITION 5. Let $\mathcal{D}_0$ be a DBPC and $\gamma(\vec{x})$ a first-order formula uniform in S_0. The *possible answers* to γ wrt $\mathcal{D}_0$, denoted $\mathsf{pans}(\gamma, \mathcal{D}_0)$, is the smallest set of pairs $(\vec{c}, \chi)$ such that:

- χ is a closure of some fluent vector $\vec{\tau}$ such that $\mathcal{E} \cup \{\chi\} \models \gamma(\vec{c})$; and
- χ is consistent with $\mathcal{D}_0$ and minimal in the sense that every atomic closure in χ is necessary. $\square$

Intuitively, $\mathsf{pans}(\gamma, \mathcal{D}_0)$ characterizes all the cases where $\gamma(\vec{x})$ is satisfied in a model of $\mathcal{D}_0 \cup \mathcal{E}$ for some instantiation of $\vec{x}$. In our example, $\mathsf{pans}(\mathit{In}(\mathit{box}_1, x, S_0), \mathcal{D}_0)$ is the set

$$\{(\mathit{item}_1, \chi_3), (\mathit{item}_2, \chi_3), (\mathit{item}_1, \chi_5), (\mathit{gold}, \chi_5)\}.$$

3.2 Formulas with finite possible answers

Observe that the possible answers to a formula may be *infinite*. For instance, let $\gamma_1(x)$ be $\mathit{In}(\mathit{box}_3, x, S_0)$ and $\mathcal{D}_0$ as before. Since nothing is said about $\mathit{In}(\mathit{box}_3, x, S_0)$ in $\mathcal{D}_0$, for every constant c in $\mathcal{L}$, $(c, \chi_c) \in \mathsf{pans}(\gamma_1, \mathcal{D}_0)$, where χ_c is the closure of $\mathit{In}(\mathit{box}_3, w, S_0)$ on $\{c\}$. Similarly, let $\gamma_2(x)$ be $\neg\,\mathit{In}(\mathit{box}_1, x, S_0)$. Then, $\mathsf{pans}(\gamma_2, \mathcal{D}_0)$ includes the infinite set $\{(c, \chi_3) \mid c \neq \mathit{item}_1, c \neq \mathit{item}_2\}$, since everything but item_1 or item_2 is not in box_1 when $\mathcal{E} \cup \{\chi_3\}$ is assumed.

Following this observation we distinguish two potential sources of an infinite set of possible answers to γ wrt to $\mathcal{D}_0$: first, when γ includes a fluent atom of the form $F(\vec{c}, w, S_0)$ that is not mentioned in $\mathcal{D}_0$, and second, when γ includes negative literals. Our objective is to identify a class of formulas γ for which $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is finite. To that end we appeal to the notions of *just-in-time* and *range-restricted*.

DEFINITION 6. Let $\mathcal{D}_0$ be a DBPC and $\gamma(\vec{x})$ a first-order formula uniform in S_0. Then $\gamma(\vec{x})$ is *just-in-time (JIT)* wrt $\mathcal{D}_0$ iff for every pair $(\vec{c}, \chi)$ in $\mathsf{pans}(\gamma, \mathcal{D}_0)$, there exists a consistent set T of possible closures wrt $\mathcal{D}_0$ such that $T \cup \mathcal{E} \models \chi$. $\square$

Definition 6 implies that each of the possible answers to γ wrt $\mathcal{D}_0$ consists of information that is listed *explicitly* in $\mathcal{D}_0$. For example, consider again $\mathsf{pans}(\mathit{In}(\mathit{box}_1, x, S_0), \mathcal{D}_0)$ and note that χ_3 and χ_5 are implied by $\chi_3 \wedge \chi_4$ and $\chi_5 \wedge \chi_6$, respectively, which are both possible closures wrt $\mathcal{D}_0$. This provides a way to avoid the cases that are similar to γ_1 (note that γ_1 is not JIT wrt $\mathcal{D}_0$). Nonetheless, this is not enough to avoid an infinite set of possible answers (note that γ_2 is JIT wrt $\mathcal{D}_0$). We shall also require formulas to be *range-restricted* in the following sense.

DEFINITION 7. The first-order formula γ is *safe-range* wrt a set of variables X according to the following rules:

1. let $\vec{t}$ be a vector of variables and constants, c a constant, and x a variable of sort object, then:
   - $x = c$ is safe-range wrt $\{x\}$;
   - $F(\vec{t}, c, S_0)$ is safe-rage wrt $\{\}$;
   - $F(\vec{t}, x, S_0)$ is safe-range wrt $\{x\}$ if x is not included in $\vec{t}$, and safe-range wrt $\{\}$ otherwise;
2. if ϕ is safe-range wrt X_ϕ, ψ is safe-range wrt X_ψ, then:
   - $\phi \vee \psi$ is safe-range wrt $X_\phi \cap X_\psi$;
   - $\phi \wedge \psi$ is safe-range wrt $X_\phi \cup X_\psi$;
   - $\neg\phi$ is safe-range wrt $\{\}$;
   - $\exists x \phi$ is safe-range wrt $X/\{x\}$ provided that $x \in X$;
3. no other formula is safe-range.

A formula is said to be *range-restricted* iff it is safe-range wrt the set of its free variables. $\square$

For example, $\mathit{In}(x, y, S_0)$ is safe-range wrt $\{y\}$ but not range-restricted and not JIT wrt $\mathcal{D}_0$ of our example. On the other hand, the formulas $\mathit{In}(x, y, S_0) \wedge x = \mathit{box}_1$ and $\mathit{In}(\mathit{box}_1, y, S_0)$ are both range-restricted and JIT wrt $\mathcal{D}_0$. Note also that γ_2 is not range-restricted.

We now state the main result of this section.

THEOREM 8. *Let $\mathcal{D}_0$ be a DBPC and $\gamma(\vec{x},\vec{y})$ a first-order formula uniform in S_0 that is JIT wrt $\mathcal{D}_0$ and safe-range wrt the variables in $\vec{x}$. Then for every constant vector $\vec{d}$ that has the same size as $\vec{y}$, $\mathsf{pans}(\gamma(\vec{x},\vec{d}),\mathcal{D}_0)$ is a finite set $\{(\vec{e}_1,\chi_1),\ldots,(\vec{e}_n,\chi_n)\}$ such that the following holds:*

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall\vec{x}.\gamma(\vec{x},\vec{d}) \equiv \bigvee_{i=1}^{n}(\vec{x}=\vec{e}_i \wedge \chi_i).$$

The proof is done by induction on the construction of γ. Since γ is safe-range wrt the variables in $\vec{x}$ we only need to consider the cases of Definition 7. Even though the ideas are straightforward, the actual proof is tedious and can be found in the appendix.

In other words, the range-restricted and JIT assumptions on a formula guarantee *finitely* many possible answers, and imply a syntactic normal form for the formula as shown next.

DEFINITION 9. Let γ be a first-order range-restricted formula uniform in S_0 that is JIT wrt $\mathcal{D}_0$. Then, the *possible answers normal form (PANS-NF)* of γ wrt $\mathcal{D}_0$ is a disjunction of formulas of the form $\vec{x}=\vec{e}\wedge\chi$ as implied by Theorem 8. $\Box$

For example, consider again $\mathsf{pans}(\mathit{In}(\mathit{box}_1,x,S_0),\mathcal{D}_0)$. The possible answers normal form of $\mathit{In}(\mathit{box}_1,x,S_0)$ wrt $\mathcal{D}_0$ is the formula

$$(x=\mathit{item}_1\wedge\chi_3)\vee(x=\mathit{item}_2\wedge\chi_3)\vee(x=\mathit{item}_1\wedge\chi_5)\vee(x=\mathit{gold}\wedge\chi_5).$$

3.3 Range-restricted action theories

We now have all the necessary machinery to define our type of basic action theories.

DEFINITION 10. A successor state axiom for F is *range-restricted* iff $\gamma_F^+(\vec{x},a,s)$ and $\gamma_F^-(\vec{x},a,s)$ are disjunctions of formulas of the form:

$$\exists\vec{z}(a=A(\vec{y})\wedge\phi(\vec{x},\vec{z},s)),$$

where $\vec{y}$ may contain some variables from $\vec{x}$, $\vec{z}$ corresponds to the remaining variables of $\vec{y}$, and ϕ (called the *context formula*) is uniform in s and is such that $\phi(\vec{x},\vec{z},S_0)$ is safe-range wrt a set that includes the variables in $\vec{x}$ that are not in $\vec{y}$.

An SRA for A is *range-restricted* iff $\Theta_A(\vec{y},r,S_0)$ is safe-range wrt any set of variables. A *range-restricted basic action theory (RR-BAT)* is one such that all axioms in $\mathcal{D}_{ss}$, $\mathcal{D}_{sr}$ are range-restricted and $\mathcal{D}_0$ is a DBPC. $\Box$

For example, observe that the SSAs for *RobotAt* and *At*, i.e., (1) and (2) follow the structure implied by Definition 10 as they are built on appropriate disjunctions. Observe also that the context formulas of the SSAs have been carefully crafted so that they also satisfy the required condition about being range-restricted. In particular note that the instantiated context formula of $\gamma^+_{\mathit{RobotAt}}$, that is,

$$\mathit{RobotAt}(z_1,S_0)\wedge\mathit{RobotDir}(z_2,S_0)\wedge\mathit{Connected}(z_1,z_2,x,S_0),$$

is safe-range wrt $\{x\}$, as required by Definition 10.

The important property of RR-BATs is that when a γ_F^*[4] formula in the successor state axioms is instantiated with a ground action $A(\vec{c})$, $\mathcal{D}_{una}$ can be used to eliminate the action term $a=A(\vec{c})$ so that γ_F^* becomes range-restricted (similarly for Θ_A in SRAs). So,

[4]We use γ^* to denote either γ^+ or γ^-.

whenever these formulas are also JIT wrt $\mathcal{D}_0$ then we can simplify them into PANS-NF by means of Theorem 8.

In the next section we show how to progress an RR-BAT whenever the JIT assumption holds and give an algorithm for a special case that the theory is built on conjunctive formulas.

4 Just-in-time progression

The RR-BATs are defined so that when a JIT assumption holds, there is a finite set of ground fluent atoms that may be affected by a ground action. The intuition is that in this case we can progress $\mathcal{D}_0$ by appealing to the progression technique in [Vassos, Lakemeyer, and Levesque 2008] that works when the set of fluents that may be affected is fixed by the arguments of the action.

4.1 The progression method for the general case

The next definition captures the condition under which our method for progression is logically correct.

DEFINITION 11. An RR-BAT $\mathcal{D}$ is *just-in-time (JIT)* wrt (α, d), where α is a ground action and d is a sensing result, iff for all fluent symbols F, $\gamma_F^+(\vec{x}, \alpha, S_0)$ and $\gamma_F^-(\vec{x}, \alpha, S_0)$ are JIT wrt $\mathcal{D}_0$, and $\Theta_A(\vec{c}, d, S_0)$ is JIT wrt $\mathcal{D}_0$, where α is $A(\vec{c})$. □

Essentially, this condition requires that for all input-grounded fluent atoms that are needed for the specification of the effects of α, there should be disjunctive or complete information about their output in the form of some PCA in $\mathcal{D}_0$.

We introduce the following notation.

DEFINITION 12. Let $\mathcal{D}$ be an RR-BAT that is JIT wrt (α, d). Without loss of generality we assume that $\Theta_A(\vec{e}, d, S_0)$ and all $\gamma_F^+(\vec{x}, \alpha, S_0)$, $\gamma_F^-(\vec{x}, \alpha, S_0)$ are in PANS-NF wrt $\mathcal{D}_0$. Then, the *context set* of (α, d) wrt $\mathcal{D}$, denoted as $\mathcal{J}$, is the set of all $F(\vec{c}, w, S_0)$ such that one of the following is true:

1. $\vec{x} = \langle \vec{c}, e \rangle \land \chi$ is a disjunct of some $\gamma_F^*(\vec{x}, \alpha, S_0)$;
2. $F(\vec{c}, w, S_0)$ appears in some $\gamma_F^*(\vec{x}, \alpha, S_0)$;
3. $F(\vec{c}, w, S_0)$ appears in $\Theta_A(\vec{c}, d, S_0)$, where $\alpha = A(\vec{c})$. □

Intuitively, the context set $\mathcal{J}$ specifies all those atomic closures that need to be updated after the action is performed (case 1) as well as those on which the change is conditioned on (case 2), and the atomic closures for which some condition is sensed to be true (case 3). The important property of $\mathcal{J}$, which follows from Theorem 8, is that it is a *finite* set.

In the simple case of our running example and the ground action *moveFwd*,[5] the context set is the following:

$$\big\{\, \mathit{RobotAt}(w, S_0), \mathit{RobotDir}(w, S_0), \mathit{Connected}(\mathit{loc}_1, \mathit{north}, w, S_0), \\ \mathit{Holding}(w, S_0), \mathit{At}(\mathit{box}_2, w, S_0), \mathit{At}(\mathit{item}_2, w, S_0), \mathit{At}(\mathit{gold}, w, S_0) \big\}.$$

Note that $\mathit{At}(\mathit{box}_1, w, S_0)$ and $\mathit{At}(\mathit{item}_1, w, S_0)$ are not needed in the progression as this information is neither affected when *moveFwd* is applied in $\mathcal{D}_0$, nor needed to specify some other effect of the action.

[5]For actions with no sensing result axioms, such as *moveFwd*, we typically do not refer to the sensing result *d*.

We now define the $\mathcal{J}$-models which will provide a way of separating $\mathcal{D}_0$ into two parts: one that remains unaffected after the action is performed and one that needs to be updated. The part that remains unaffected corresponds to all those PCAs that do not mention any atom in $\mathcal{J}$. For the part that needs updating, we will construct a large PCA that lists all the possibilities for the closures of the atoms in $\mathcal{J}$ as follows.

DEFINITION 13. Let $\mathcal{J} = \{\tau_1, \ldots, \tau_n\}$ be the context set of (α, d) wrt an RR-BAT $\mathcal{D}$. A *$\mathcal{J}$-model* θ is a sentence of the form $(\chi_1 \wedge \ldots \wedge \chi_m)$ such that every χ_i is a possible closure wrt $\mathcal{D}_0$ that mentions at least one of the atoms in $\mathcal{J}$, all atoms in $\mathcal{J}$ are mentioned in θ, θ is consistent, and atoms in θ appear in lexicographic order. □

Note that since $\mathcal{D}_0$ is finite there are finitely many $\mathcal{J}$-models. In the simple case of our running example there are two $\mathcal{J}$-models, namely:

$$\theta_1 : \chi_1 \wedge \chi_2 \wedge \chi_3 \wedge \chi_4 \wedge \chi_7 \wedge \chi_8 \wedge \chi_{11} \wedge \chi_{12} \wedge \chi_{13};$$
$$\theta_2 : \chi_1 \wedge \chi_2 \wedge \chi_5 \wedge \chi_6 \wedge \chi_7 \wedge \chi_8 \wedge \chi_{11} \wedge \chi_{12} \wedge \chi_{13}.$$

Note that χ_9 and χ_{10} that represent the location of *box*$_1$ and *item*$_1$ are not included in any of the $\mathcal{J}$-models as the corresponding atoms, that is, $At(box_1, w, S_0)$ and $At(item_1, w, S_0)$, are not included in the context set $\mathcal{J}$.

The disjunction ϕ of all the $\mathcal{J}$-models is a PCA that corresponds to the "cross-product" of the PCAs in $\mathcal{D}_0$ that hold information about $\vec{\tau}$. Essentially, ϕ corresponds to the part of $\mathcal{D}_0$ that needs updating, and the intuition is that we can progress $\mathcal{D}_0$ by progressing each of the $\mathcal{J}$-models separately.

DEFINITION 14. Let $\mathcal{J}$ be the context set of (α, d) wrt an RR-BAT $\mathcal{D}$, and θ a $\mathcal{J}$-model that is the closure of $\{F_1(\vec{c}_1, w, S_0), \ldots, F_n(\vec{c}_n, w, S_0)\}$ on $\langle V_1, \ldots, V_n \rangle$. We define Γ_i^+ as the smallest set of constants e such that:

1. $\vec{x} = \langle \vec{c}_i, e \rangle \wedge \chi$ is a disjunct of $\gamma_{F_i}^+(\vec{x}, \alpha, S_0)$;
2. $\{\chi \wedge \theta\} \cup \mathcal{E}$ is consistent.

The set Γ_i^- is defined similarly based on $\gamma_{F_i}^-(\vec{x}, \alpha, S_0)$. The *progression* of θ wrt (α, d) is the closure on the updated vector $\langle V_1', \ldots, V_n' \rangle$, where V_i' is the set $(V_i - \Gamma_i^-) \cup \Gamma_i^+$. The $\mathcal{J}$-model θ is *filtered* iff $\{\Theta_A(\vec{c}, d, S_0) \wedge \theta\} \cup \mathcal{E}$ is inconsistent, where α is $A(\vec{c})$. □

Each $\mathcal{J}$-model θ is updated based on the possible answers of the instantiated formulas $\gamma_{F_i}^*$. For every disjunct of $\gamma_{F_i}^*$ expressed in PANS-NF, the atom $w = e$ is either removed or added to the closure of $F_i(\vec{c}_i, w, S_0)$ provided that the condition χ for the change is consistent with θ. Similarly, a $\mathcal{J}$-model may be filtered out if it is not consistent with the conditions that are implied by the sensing result d.

For example, consider θ_1 and the atom $At(item_2, w, S_0)$ from $\mathcal{J}$. The corresponding Γ^+ and Γ^- for this atom are both the empty set $\{\}$ as *item*$_2$ is inside *box*$_1$ in model θ_1, and therefore its location remains intact. On the other hand, if one considers model θ_2, the corresponding Γ^+ and Γ^- for $At(item_2, w, S_0)$ are $\{loc_2\}$ and $\{loc_1\}$, respectively. Putting it all together, the updated $\mathcal{J}$-models θ_1' and θ_2' are as follows:

$$\theta_1' : \chi_1' \wedge \chi_2 \wedge \chi_3 \wedge \chi_4 \wedge \chi_7 \wedge \chi_8 \wedge \chi_{11}' \wedge \chi_{12} \wedge \chi_{13}',$$
$$\theta_2' : \chi_1' \wedge \chi_2 \wedge \chi_5 \wedge \chi_6 \wedge \chi_7 \wedge \chi_8 \wedge \chi_{11}' \wedge \chi_{12}' \wedge \chi_{13},$$

where χ_1', χ_{11}', χ_{12}', and χ_{13}' are the atomic closures of $RobotAt(w, S_0)$, $At(box_2, w, S_0)$, $At(item_2, w, S_0)$, and $At(gold, w, S_0)$ on $\{loc_2\}$, respectively.

We now state the main result of this section that illustrates how the new knowledge base is constructed from $\mathcal{D}_0$.

THEOREM 15. *Let $\mathcal{D}$ be an RR-BAT that is consistent and JIT wrt the ground action α and sensing result d, $\mathcal{J}$ the context set of (α, d) wrt $\mathcal{D}$, $\{\theta_1, \ldots, \theta_n\}$ the set of all the $\mathcal{J}$-models that are not filtered, and $\{\phi_1, \ldots, \phi_m\}$ the set of all PCAs in $\mathcal{D}_0$ that do not have any atoms in common with any $\mathcal{J}$-model. Let $\mathcal{D}_\alpha$ be the following set:*

$$\{\bigvee_{i=1}^{n} \theta'_i, \phi_1, \ldots, \phi_m\},$$

where θ'_i is the progression of θ_i wrt (α, d). Then, the set $\mathcal{D}_\alpha(S_0/\, do(\alpha, S_0))$ is a strong progression of $\mathcal{D}$ wrt (α, d), where $\mathcal{D}_\alpha(\sigma/\sigma')$ denotes the result of replacing every occurrence of σ in every sentence in $\mathcal{D}_\alpha$ by σ'.

In the absence of sensing, the proof idea is to show that assuming $\mathcal{E}$, $\mathcal{D}_0$ is equivalent to $\{\bigvee_1^n \theta_i, \phi_1, \ldots, \phi_m\}$, and progressing $\mathcal{D}_0$ reduces to updating each of the sub-models θ_i appropriately. For the case of sensing we only need to remove every θ_i that is not consistent with the instantiated SRA. As far our running example is concerned, observe that $\{\theta'_1 \vee \theta'_2, \chi_9, \chi_{10}\}$ is a strong progression of $\mathcal{D}$ wrt *moveFwd*.

We close by noting two important points. First, observe that the progression of $\mathcal{D}_0$ is again a DBPC. On the other hand, the fact that $\mathcal{D}$ is JIT wrt some action α may not be preserved in general after α or some other action is performed. This is because the JIT assumption essentially requires that there is disjunctive or complete information in the KB about the output w of all the fluent atoms of the form $F(\vec{c}, w, S_0)$ that are mentioned in the instantiated $\gamma^*(\vec{x}, \alpha, S_0)$ formulas, and these atoms may be different each time. Nonetheless, checking whether the JIT assumption holds may be performed by the same method that computes the possible answers of formulas as we will see in the next section. Moreover, the JIT assumption may be enforced by an appropriate account of sensing so that complete or disjunctive knowledge is acquired for these atoms. Please refer to Section 6 for a short discussion on this matter.

4.2 The case of conjunctive queries

Our method for progression is based on the ability to compute the possible answers to the γ_F^* and Θ_A formulas wrt $\mathcal{D}_0$, as we have assumed that they are in PANS-NF. In order to give some insight on the practicality of our method, we examine the case that these formulas are similar to the *conjunctive queries* of database theory [Abiteboul, Hull, and Vianu 1994].

DEFINITION 16. A *conjunctive query* γ is a formula uniform in S_0 of the form

$$\exists \vec{x}(\phi_1 \wedge \cdots \wedge \phi_n),$$

where ϕ_i is a fluent atom with free variables not necessarily in $\vec{x}$. □

Conjunctive queries offer an intuitive way of building safe-range formulas using the output of an atom with a ground input as an input to another atom. For instance, note that the context formula of $\gamma^+_{RobotAt}$ in (1) is a conjunctive query, and observe that the output of the fluent atoms *RobotAt*(z_1, s) and *RobotDir*(z_2, s) is used to specify the input of *Connected*(z_1, z_2, x, s). Similarly, observe that the γ^+_{At} and γ^-_{At} formulas in (2) are built on conjunctive queries of this sort.

Given a conjunctive query γ as input and a DBPC $\mathcal{D}_0$, $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ that is described in Algorithm 1 checks whether γ is range-restricted and JIT wrt $\mathcal{D}_0$, and if so, computes

Algorithm 1: $\mathsf{PANS}(\gamma, \mathcal{D}_0)$

```
if $\gamma$ is the empty conjunction then
    return $\{\}$;                                        // query successfully processed
$\Gamma := \{F(\vec{c}, t, S_0) \in \gamma \mid F(\vec{c}, w, S_0)$ is mentioned in $\mathcal{D}_0\}$;
if $\Gamma = \emptyset$ then
    return failure;                                       // no atom to continue
$X := \emptyset$;                                         // initialize the set of possible answers
$\tau$ := the first atom $F(\vec{c}, t, S_0)$ in $\Gamma$;
$\gamma' := \gamma$ without the atom $\tau$;
for possible closures $\chi$ of $\tau$ on $V$ wrt $\mathcal{D}_0$, do
    if $t$ is a constant then
        if $t \in V$ then
            $Y := \mathsf{PANS}(\gamma', \mathcal{D}_0)$;               // recursive call, $t$ constant
            if $Y =$ failure then
                return failure
            $Y' := \{(\omega, \chi \wedge \chi') \mid (\omega, \chi') \in Y, \chi \wedge \chi' \not\models \bot\}$;
            $X := X \cup Y'$;                             // merge the possible answers
    else
        for constants $e \in V$ do
            $Y := \mathsf{PANS}(\gamma'|^t_e, \mathcal{D}_0)$;          // recursive call, $t$ variable
            if $Y =$ failure then
                return failure
            $Y' := \{(t = e \wedge \omega, \chi \wedge \chi') \mid (\omega, \chi') \in Y, \chi \wedge \chi' \not\models \bot,\ t = e \wedge \omega \not\models \bot\}$;
            $X := X \cup Y'$;                             // merge the possible answers
return (the projection of X on the free variables of $\gamma$)
```

the set $\mathsf{pans}(\gamma, \mathcal{D}_0)$. The algorithm works by i) selecting an atom for which a finite set of possible answers is guaranteed (lines 3–8), ii) specifying the possible answers to this atom and recursively finding those to the query without the selected atom (lines 9–14, 17–21), and iii) merging the answers (lines 15–16, 22–23), until all atoms in γ have been selected.

The next theorem states the correctness of PANS.

THEOREM 17. *Let $\mathcal{D}_0$ be a finite DBPC and γ a first-order conjunctive query uniform in S_0. Then, PANS$(\gamma, \mathcal{D}_0)$ always terminates, and moreover if γ is range-restricted and JIT wrt $\mathcal{D}_0$, it returns the set pans$(\gamma, \mathcal{D}_0)$.*

Algorithm $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ can easily be extended to handle equalities as well as negated atoms. The first case can be addressed via standard unification procedures. For negated atoms the idea is that a *ground* literal of the form $\neg F(\vec{c}, d, S_0)$ such that $F(\vec{c}, w, S_0)$ is mentioned in $\mathcal{D}_0$, may also be selected by the algorithm. Then, the algorithm works in the same way as for the ground positive literal except that it iterates over the possible closures of $F(\vec{c}, w, S_0)$ for which $F(\vec{c}, d, S_0)$ is *not* true. (Observe the similarities with *negation as failure* of logic programming [Apt and Pellegrini 1994].)

We now turn our attention to the complexity of $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ and progression. First note that when γ is a range-restricted conjunctive query that is also JIT wrt $\mathcal{D}_0$, the size of $\mathsf{pans}(\gamma, \mathcal{D}_0)$ is $O(N^n)$, where n is the size of γ and N is the size of $\mathcal{D}_0$. This is because the assumptions for γ and Theorem 8 ensure that in the worst case for each atom τ in γ, the set $\mathsf{pans}(\tau, \mathcal{D}_0)$ corresponds to the information in the entire $\mathcal{D}_0$.

THEOREM 18. *Let γ be a range-restricted conjunctive query and $\mathcal{D}_0$ a finite DBPC. Let n be the number of atoms in γ, k the number of distinct atoms with a ground input in $\mathcal{D}_0$, m the maximum number of disjunct in a PCA in $\mathcal{D}_0$, and l the maximum number of constants in an atomic closure in $\mathcal{D}_0$. Then $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ runs in time $O((k \cdot l + n^2) \cdot (m \cdot l)^n)$.*

Even though the complexity is exponential on the size n of γ, our progression method evaluates the formulas γ_F^* and Θ_A from $\mathcal{D}_{ss}$ and $\mathcal{D}_{sr}$, whose sizes do not grow. In fact, it is safe to assume that the sizes of these formulas are bounded by a small integer N, in which case the complexity of $\mathsf{PANS}(\gamma, \mathcal{D}_0)$ is a polynomial with degree N.

Assuming then that all the relevant formulas for progressing $\mathcal{D}_0$ wrt α are in PANS-NF by the use of $\mathsf{PANS}(\gamma, \mathcal{D}_0)$, the complexity of our method is dominated by the process of specifying all the $\mathcal{J}$-models, where $\mathcal{J}$ is the context set of α. In the worst case, the size of $\mathcal{J}$ is k, which essentially means that with the execution of α all the atoms in $\mathcal{D}_0$ become *mutually constrained*, and thus need to appear in the same PCA.

Nonetheless, we expect the size of $\mathcal{J}$ to be manageable in practice, as the robotic or agent domains we have in mind are quite different from logical puzzles of this sort. In particular, we expect that the amount of disjunctive knowledge that is represented in $\mathcal{D}_0$ always comes in (many) small "packages." Under these plausible assumptions, it follows that m is always bounded by some small constant M, in which case, the running time of our progression method and the size of $\mathcal{D}_\alpha$ are both polynomial to the size of $\mathcal{D}_0$ with a degree that depends on the actual values of N, M.

These assumptions are relatively strong and certainly do not apply to all applications we have in mind, but we believe that they are safe for many "organic" scenarios where the theory $\mathcal{D}$ is used to represent the effects of a domain that is similar to the physical world. Furthermore, as far as the assumptions on the incomplete information are concerned, we believe that they can be enforced with an appropriate account of sensing.

5 Related work

The notion of progression for basic action theories was first introduced by Lin and Reiter [1997]. The version we use here is due to Vassos et al. [2008], which we extended slightly to account for sensing. As far as a first-order definable progression is concerned, Lin and Reiter [1997] were first to investigate restrictions on the successor state axioms as they introduced a *context-free* assumption for actions.

Liu and Levesque [2005] introduced the *local-effect* assumption for actions when they proposed a weaker version of progression that is logically incomplete, but remains practical. Vassos et al. [2008] showed that, under this assumption, a logically correct first-order progression can actually be computed by updating a finite $\mathcal{D}_0$. Our restriction of Definition 10 is similar, but goes beyond local-effect. The key difference is that we do not require the arguments of a fluent F be included in those of the action. This allows us to handle actions like *turn* and *moveFwd* (with no arguments) affecting fluents like *RobotDir*(x, s) and *RobotAt*(x, s). To stay practical, though, we restricted the structure of $\mathcal{D}_0$. It is important to point out that it is not always possible to revert to a *ground* action of the form *move*(x, y, z) as, for instance, the starting x or destination y may be unknown to the agent.

In [Liu and Lakemeyer 2009], the authors first showed how the result of Vassos et al. [2008] on local-effect progression relates to the notion of forgetting, and examined the more expressive so-called *normal* actions. Both our work on actions with bounded effects and that of Liu and Lakemeyer allow the representation of actions that go beyond local-effect, but each approach proceeds in a different direction.

The intuition behind normal actions is that a ground action α is allowed to have non-local effects on a fluent F as long as α has local effects on all fluents appearing in the effect formulas γ_F^*. For instance, consider the ground action $\mathit{move}(\mathit{box}_1, \mathit{loc}_1, \mathit{loc}_2)$ for moving object box_1 together with every object inside it from location loc_1 to location loc_2. Such action does have non-local effects on fluent $\mathit{At}(x, y, s)$ because it affects the location of objects that are inside box_1 and these are not named as arguments in the action. Nonetheless, assuming that the only mentioned fluent in γ_{At}^* is $\mathit{In}(x, y, s)$, the action is considered normal, since it trivially (i.e., has no effect whatsoever) has local-effects on $\mathit{In}(x, y, s)$.

While similar in its effects, the action *moveFwd* of our example, on the other hand, is not normal. This is because the action has non-local effects on fluent $\mathit{RobotAt}(x, y, s)$, and $\mathit{RobotAt}(x, y, s)$ is in fact mentioned in $\gamma_{\mathit{RobotAt}}^+$. To see this, one has to look carefully at the technical details of Definition 4.4 in [Liu and Lakemeyer 2009]: a ground action has local-effects on fluent F if using $\mathcal{D}_{\mathit{una}}$ and the action it is possible to simplify γ_F^+ and γ_F^- into a disjunction that explicitly states all potentially affected ground atoms of F. Since the agent may not know a-priori the boundaries of the world it is situated in, the number of potential locations is not bounded, hence such a simplification is not possible by only using $\mathcal{D}_{\mathit{una}}$. Our approach is different in that we also use $\mathcal{D}_0$ in order to achieve a similar simplification. As a result, we also require additional constraints on the structure of $\mathcal{D}_0, \mathcal{D}_{\mathit{ss}}, \mathcal{D}_{\mathit{sr}}$ as well as the JIT assumption.

Another approach to first-order progression involves imposing restrictions on the *queries* that can be performed over a basic action theory. This was first investigated by Lin and Reiter [1997] who showed that for answering projection queries that are uniform in some situation σ, a strong progression is always equivalent to a first-order set of sentences. Shirazi and Amir [2005] proved a similar result in the context of *logical filtering*. Vassos and Levesque [2008] extended this result showing that un-nested quantification over situations may also be allowed.

Outside of the situation calculus, Thielscher [1999] defined a dual representation for basic action theories based on state update axioms that explicitly define the direct effects of each action, and investigated progression in this setting. Unlike our work, where the sentences in $\mathcal{D}_0$ are replaced by an updated version, there, the update relies on expressing the changes using constraints which may need to be conjoined to the original database.

Finally, we observe that the notion of possible closures is a generalization of the *possible values* of Vassos and Levesque [2007]. The notions of the safe-range and range-restricted queries come from the database theory where this form of "safe" queries has been extensively studied [Abiteboul, Hull, and Vianu 1994]. The notion of just-in-time formulas was introduced for a different setting in [De Giacomo, Levesque, and Sardina 2001] and, in our case, is also related to the *active domain* of a database [Abiteboul, Hull, and Vianu 1994].

6 Discussion

We have proposed a type of basic action theories, where the KB is a *database of possible closures* and actions have *bounded effects*. For such theories, we showed a logically correct method for computing a finite first-order progression by directly updating the KB. Moreover, we developed an algorithm for the main progression task, namely computing possible answers, for the special case of conjunctive queries. Our progression account is, as far as we are aware, the first one to go beyond local-effect, while being able to handle a particular class of simple (though common) actions, sensing, and incomplete information with possibly infinite domains.

In particular, our approach is strictly more general than local-effect actions, but as far as the work of [Liu and Lakemeyer 2009] on normal actions is concerned, neither approach is strictly more general than the other. First, as we already illustrated, our RR-BATs are able to handle some very simple cases, like the action *moveFwd*, that are not normal. In addition, only our approach accounts for sensing actions. On the other hand, our approach relies on having disjunctive knowledge about the relevant part of the KB that needs to be updated, which is not always needed for a normal action. For example, the action $\mathit{move}(x_1, x_2)$ that results in the agent moving from x_1 to x_2 and taking along the objects she is holding is normal, and the KB can be progressed even if there is no information in the KB about the objects that the agent is holding. With our approach, however, we can only progress provided there is disjunctive (or complete) knowledge about the objects she is holding. There is also a mismatch between the structure of the KBs that are used. In the general case of Liu and Lakemeyer [2009], our RR-BATs are clearly more restricted, but in the case where the authors considered an implementation based on the so-called proper$^+$ KBs, both approaches have benefits depending on context.

Also, the two approaches are conceptually quite different. The work on normal actions is based on the notion of forgetting, while ours is rooted on database theory. Whereas the KBs used for normal actions are more general than our DBPCs, the latter are closer to *incomplete* databases, e.g., the so-called c-tables [Imielinski and Lipski 1984]. In fact, our work is motivated by the fact that almost all practical real agent systems or automated planners make use of database-type approaches to model the agent's beliefs. As incomplete/inconsistent databases is a very active area of research, we expect to be able to leverage on such technology (e.g., [Fuxman, Fazli, and Miller 2005]).

As far as future work is concerned, we envision work on practical systems that do not rely on one method for progression but on a collection of techniques that apply under different conditions. For example, one might build a system that uses a DBPC as a KB which is progressed wrt both normal and range-restricted actions depending on the conditions that are satisfied. In the cases that progression cannot work, the system may attempt to impose the JIT condition in order to perform a range-restricted progression or alternatively fall back to using regression (that accounts for much wider classes) for any reasoning about the future until the necessary conditions for progression hold. As far as RR-BATs are concerned, our future work focuses on a richer notion of sensing. At the moment, sensing can be used effectively to reduce the uncertainty as long as some "bounds" on such uncertainty is already available. In our example from Section 3.1, the $\mathit{senseGold}(y)$ action (see end of Sections 2.1) can be used to determine whether there is gold inside box_1, as the agent already has some information of what could be inside such box. In that way, sensing actions provide a "filtering" mechanism for what is already stated in the KB. We believe it is not difficult to generalize such sensing account to allow sensing of atomic closures (Definition 2) or even possible closures (Definition 3). However, one has to be careful on how to merge such general sensing outcomes with the knowledge already encoded in the KB.

References

Abiteboul, S., R. Hull, and V. Vianu [1994]. *Foundations of Databases: Logical Level.* Addison Wesley.

Apt, K. and A. Pellegrini [1994]. On the occur-check free Prolog program. *ACM Transactions on Programming Languages and Systems (TOPLAS) 16*(3), 687–726.

De Giacomo, G., H. J. Levesque, and S. Sardina [2001]. Incremental execution of guarded theories. *Computational Logic 2*(4), 495–525.

Fuxman, A., E. Fazli, and R. J. Miller [2005]. ConQuer: Efficient management of inconsistent databases. In *Proceedings of the ACM International Conference on Management of Data (SIGMOD)*, pp. 155–166.

Imielinski, T. and W. Lipski [1984]. Incomplete information in relational databases. *Journal of the ACM 31*(4), 761–791.

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

Liu, Y. and G. Lakemeyer [2009]. On first-order definability and computability of progression for local-effect actions and beyond. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 860–866.

Liu, Y. and H. J. Levesque [2005]. Tractable reasoning in first-order knowledge bases with disjunctive information. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 639–644.

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

Reiter, R. [1991]. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In V. Lifschitz (Ed.), *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*, pp. 359–380. San Diego, CA: Academic Press.

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

Scherl, R. and H. J. Levesque [2003]. Knowledge, action, and the frame problem. *Artificial Intelligence 144*(1–2), 1–39.

Shirazi, A. and E. Amir [2005]. First-order logical filtering. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 589–595.

Thielscher, M. [1999]. From situation calculus to fluent calculus: State update axioms as a solution to the inferential frame problem. *Artificial Intelligence 111*(1-2), 277–299.

Vassos, S., G. Lakemeyer, and H. J. Levesque [2008]. First-order strong progression for local-effect basic action theories. In *Proceedings of Principles of Knowledge Representation and Reasoning (KR)*, pp. 662–272.

Vassos, S. and H. Levesque [2007]. Progression of situation calculus action theories with incomplete information. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2029–2034.

Vassos, S. and H. J. Levesque [2008]. On the progression of situation calculus basic action theories: Resolving a 10-year-old conjecture. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pp. 1004–1009.

Vassos, S., S. Sardina, and H. Levesque [2009]. Progressing basic action theories with non-local effect actions. In *Proceedings of the Ninth International Symposium on Logical Formalizations of Commonsense Reasoning (CS'09)*, pp. 135–140.

A Proof of Theorem 8

Proof. We prove the theorem by induction on the construction of the formulas γ. Since γ is safe-range wrt the variables in $\vec{x}$ we only need to consider the cases of Definition 7.

Base case. We only show the case that $\gamma(x, y)$ is $F(\vec{c}_1, y, \vec{c}_2, x)$, where x, y are distinct variables and $\vec{c}_1, \vec{c}_2$ are vectors of constants of sort object, and the other cases are similar. Let d be an arbitrary constant of the language. Then $\gamma(x, d)$ is the formula $F(\vec{c}_1, d, \vec{c}_2, x)$. We first show that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$. Let e be a constant of $\mathcal{L}$. We take two cases as follows.

- Case i): $F(\vec{c}_1, d, \vec{c}_2, e)$ is consistent with $\mathcal{D}_0 \cup \mathcal{E}$. Then, by the fact that $\gamma(x, y)$ is just-in-time wrt $\mathcal{D}_0$ it follows that there is a closure χ such that $\{\chi\} \cup \mathcal{E} \models \gamma(e, d)$, where χ is a conjunction of closures each of which is a possible closure wrt $\mathcal{D}_0$. It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$.

- Case ii): $F(\vec{c}_1, d, \vec{c}_2, e)$ is not consistent with $\mathcal{D}_0 \cup \mathcal{E}$. It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$, such that $F(\vec{c}_1, d, \vec{c}_2, e)$ is not true in any of the possible closures wrt ϕ.

It follows that there is a possible closures axiom ϕ in $\mathcal{D}_0$ that mentions $F(\vec{c}_1, d, \vec{c}_2, w)$. Without loss of generality we assume that ϕ is a possible closures axiom for $F(\vec{c}_1, d, \vec{c}_2, w)$. We will show how to rewrite ϕ in the form that the theorem requires. The axiom ϕ has the form

$$\bigvee_{i=1}^{n} \chi_i,$$

where each χ_i is an atomic closure of $F(\vec{c}_1, d, \vec{c}_2, w)$ on some set of constants $\{e_1, \ldots, e_m\}$, i.e, a sentence of the form

$$\forall w. F(\vec{c}_1, d, \vec{c}_2, w) \equiv w = e_1 \vee \ldots \vee w = e_m.$$

For each χ_i of this form let χ'_i be the formula

$$\bigvee_{j=1}^{m} (x = e_j \wedge \chi_i),$$

and let ϕ' be

$$\forall x. F(\vec{c}_1, d, \vec{c}_2, x) \equiv \bigvee_{i=1}^{n} \chi'_i.$$

Note that ϕ' has the form that the theorem requires. So, in order to prove the theorem it suffices to show that $\mathcal{D}_0 \cup \mathcal{E} \models \phi'$. Let M be an arbitrary model of $\mathcal{D}_0 \cup \mathcal{E}$. Since ϕ is a sentence in $\mathcal{D}_0$ it follows that $M \models \phi$. Observe that this there is exactly one k, $1 \leq k \leq n$, such that $M \models \chi_k$. Observe that if we simplify χ_k to *true* and all the other χ_i to *false* in ϕ' we obtain the sentence χ_k. Therefore, $M \models \phi'$ and since M was an arbitrary model of $\mathcal{D}_0 \cup \mathcal{E}$, it follows that $\mathcal{D}_0 \cup \mathcal{E} \models \phi'$. Also, by the definition of a possible answer and the structure of ϕ' it follows that the set $\mathsf{pans}(\gamma(x, d), \mathcal{D}_0)$ is the set that the theorem requires. Finally, since d was arbitrary it follows that this holds for every d, thus the theorem holds for the case of $F(\vec{c}_1, d, \vec{c}_2, w)$.

Inductive step. We only show the case when γ is a disjunction of formulas and the other cases are similar. We assume the theorem holds for $\phi_1(\vec{x}_1, \vec{z}, \vec{y}_1)$ that is safe-range wrt the variables in $\vec{x}_1$ and $\vec{z}$, and does not mention any free variable other than $\vec{x}_1, \vec{z}, \vec{y}_1$. Similarly we assume that the theorem holds for $\phi_2(\vec{x}_2, \vec{z}, \vec{y}_2)$ that is safe-range wrt the variables in $\vec{x}_2$ and $\vec{z}$, and does not mention any free variable other than $\vec{x}_2, \vec{z}, \vec{y}_2$, where $\vec{x}_1$ and $\vec{x}_2$ share no variables. Let $\gamma(\vec{x}_1, \vec{y}_1, \vec{x}_2, \vec{y}_2, \vec{z})$ be the formula

$$\phi_1(\vec{x}_1, \vec{z}, \vec{y}_1) \vee \phi_2(\vec{x}_2, \vec{z}, \vec{y}_2).$$

We will show that the theorem holds for γ. By the definition of the safe-range formulas it follows that γ is safe-range for the variables in $\vec{z}$. Let $\vec{d}_1$ be a vector of constants of the same size as $\vec{y}_1$, and $\vec{d}_2$ be a vector of constants of the same size as $\vec{y}_2$. By the induction hypothesis it follows that $\mathsf{pans}(\phi_1(\vec{x}_1, \vec{z}, \vec{d}_1), \mathcal{D}_0)$ is a finite set of the form

$$\{(\langle \vec{c}_{11}, \vec{e}_{11} \rangle, \chi_{11}), \ldots, (\langle \vec{c}_{1n}, \vec{e}_{1n} \rangle, \chi_{1n})\}$$

and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}_1. \forall \vec{z}. \phi_1(\vec{x}_1, \vec{z}, \vec{d}_1) \equiv \bigvee_{i=1}^{n} (\vec{x}_1 = \vec{c}_{1i} \wedge \vec{z} = \vec{e}_{1i} \wedge \chi_{1i}).$$

Similarly, $\mathsf{pans}(\phi_2(\vec{x}_2, \vec{z}, \vec{d}_2), \mathcal{D}_0)$ is a finite set of the form

$$\{(\langle \vec{c}_{21}, \vec{e}_{21} \rangle, \chi_{21}), \ldots, (\langle \vec{c}_{2m}, \vec{e}_{2m} \rangle, \chi_{2m})\},$$

and the following holds:

$$\mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{x}_2. \forall \vec{z}. \phi_2(\vec{x}_2, \vec{z}, \vec{d}_2) \equiv \bigvee_{i=1}^{m} (\vec{x}_2 = \vec{c}_{2i} \wedge \vec{z} = \vec{e}_{2i} \wedge \chi_{2i}).$$

Let $\vec{b}_1$ be a vector of constants of the same size as $\vec{x}_1$ and $\vec{b}_2$ a vector of constants of the same size as $\vec{x}_2$. It follows that

$$\begin{aligned} \mathcal{D}_0 \cup \mathcal{E} \models \forall \vec{z}. \phi_1(\vec{b}_1, \vec{z}, \vec{d}_2) \vee \phi_2(\vec{b}_2, \vec{z}, \vec{d}_2) \equiv & \\ & \bigvee_{i=1}^{n} (\vec{b}_1 = \vec{c}_{1i} \wedge \vec{z} = \vec{e}_{1i} \wedge \chi_{1i}) \vee \bigvee_{i=1}^{m} (\vec{b}_2 = \vec{c}_{2i} \wedge \vec{z} = \vec{e}_{2i} \wedge \chi_{2i}). \end{aligned}$$

By the uniqueness of names for the constants of sort object it follows that each of the atoms of the form $\vec{b}_1 = \vec{c}_{1i}$ and $\vec{b}_2 = \vec{c}_{2i}$ can be simplified to either *true* or *false*. Therefore the previous sentence can be simplified to the form that the theorem requires and the theorem follows. $\square$