

Progressing Basic Action Theories with Non-Local Effect Actions

Stavros Vassos¹, Sebastian Sardina², Hector J. Levesque¹

¹Department of Computer Science
University of Toronto
Toronto, Canada



²School of Computer Science
RMIT University
Melbourne, Australia



Commonsense 2009



Introduction

The problem we examine lies in the field of *reasoning about action and change*.

Given a logical formalism that is able to:

- 1 represent the current state of the world;
- 2 represent the dynamics of the world;
- 3 answer queries about the current state and the possible future states of the world,

we want to *update* the representation of the current state after action execution.

This is necessary when the formalism is used *online* by a *long-lived* agent.



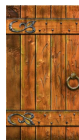
Introduction



Think of a non-player character (NPC) in a video game equipped with such a logical formalism.

1 Representing the current state:

- ▶ “the door is locked”
- ▶ “a bomb is located near the door”



2 Representing the dynamics of the world:

- ▶ “pushing a door opens the door provided that it is unlocked”
- ▶ “the explosion of a bomb breaks all objects near the bomb”

3 Answering queries about the future:

- ▶ “is there an action such that after executing it the door will be open?”

Logical formalism: the situation calculus language

The *situation calculus* is a first-order logical language with limited second-order features:

- S_0 is the initial situation;
- $do(a, S_0)$ is the resulting situation after action a has been performed in S_0 , e.g.,

$do(push(door), S_0),$

$do(trigger(bomb), S_0);$

- *fluents* are like normal predicates but also depend on a situation argument, e.g.,

$Status(door, locked, S_0),$

$Near(bomb, door, S_0).$



Logical formalism: the basic action theories

A *basic action theory* *BAT* consists essentially of:

- **KB**: a first-order knowledge base that represents what holds in the initial situation S_0 ;
- **DYN**: a set of action precondition and effect axioms that represent the dynamics of the world.

1 Representing the current state: KB

2 Represent the dynamics of the world: DYN

3 Answering queries about the future based on entailment:

- ▶ $BAT \models \neg \text{Status}(\text{door}, \text{open}, \text{do}(\text{push}(\text{door}), S_0))$,
- ▶ $BAT \models \text{Status}(\text{door}, \text{broken}, \text{do}(\text{trigger}(\text{bomb}), S_0))$.



Logical formalism: the basic action theories

A *basic action theory* *BAT* consists essentially of:

- **KB**: a first-order knowledge base that represents what holds in the initial situation S_0 ;
- **DYN**: a set of action precondition and effect axioms that represent the dynamics of the world.

1 Representing the current state: KB

2 Represent the dynamics of the world: DYN

3 Answering queries about the future based on entailment:

- ▶ $BAT \models \neg \text{Status}(\text{door}, \text{open}, \text{do}(\text{push}(\text{door}), S_0))$,
- ▶ $BAT \models \text{Status}(\text{door}, \text{broken}, \text{do}(\text{trigger}(\text{bomb}), S_0))$.

Problem: update, i.e., *progress*, the KB after action execution.



Problem: BAT progression

BAT : KB (knowledge base about S_0) + DYN (axioms for dynamics),
action a is executed by the robot.

BAT': *new* KB' + *same* DYN, such that *BAT'* characterizes the
future of *BAT* after a has been executed [Lin & Reiter 97].

BAT	BAT' (progression)
unrestricted KB unrestricted DYN	KB' needs to be second-order! [Lin & Reiter 97], [Vassos & Levesque 08]
unrestricted KB <i>local-effect</i> DYN	KB' first-order representable [Vassos & Lakemeyer & Levesque 08]



Problem: BAT progression

BAT : KB (knowledge base about S_0) + DYN (axioms for dynamics),
action a is executed by the robot.

BAT': *new* KB' + *same* DYN, such that *BAT'* characterizes the
future of *BAT* after a has been executed [Lin & Reiter 97].

BAT	BAT' (progression)
unrestricted KB unrestricted DYN	KB' needs to be second-order! [Lin & Reiter 97], [Vassos & Levesque 08]
unrestricted KB <i>local-effect</i> DYN	KB' first-order representable [Vassos & Lakemeyer & Levesque 08]

Local-effect is often *not practical*, can we *relax* this assumption?



Local-effect actions and beyond

Local-effect DYN: a ground action may only affect finitely many ground atoms that are *fixed* by the *arguments* of the action:

- the action *setstatus(door, open)* that affects only the ground fluent *Status(door, open)* is local-effect;
- the action *push(door)* that affects only the ground fluent *Status(door, open)* is *not* local-effect.
- the action *trigger(bomb)* that affects the fluents *Status(x, broken)* for all those objects *x* that are near the bomb is *not* local-effect.



Local-effect actions and beyond

Local-effect DYN: a ground action may only affect finitely many ground atoms that are *fixed* by the *arguments* of the action:

- the action *setstatus(door, open)* that affects only the ground fluent *Status(door, open)* is local-effect;
- the action *push(door)* that affects only the ground fluent *Status(door, open)* is *not* local-effect.
- the action *trigger(bomb)* that affects the fluents *Status(x, broken)* for all those objects *x* that are near the bomb is *not* local-effect.

Observation: if we can ensure that *a* affects only finitely many ground atoms (not necessarily fixed by *a*), then we can progress using the same method that works for a local-effect DYN.



Result: a case of non-local-effect BAT progression

Our assumptions:

- 1 KB is a finite *database of possible closures*;
 - 2 DYN is *non-local-effect* but *range-restricted*;
 - 3 BAT is *just-in-time* wrt the action.
- 1 and 2 are *syntactical* restrictions that specify a design strategy for building a basic action theory.
 - 3 is a *semantical* condition: it requires that the agent has *disjunctive information* for all the fluents that need to be evaluated in order to progress.

Result: in this case progression is first-order and finite.



1. KB is a finite database of possible closures (DBPC)

- Every (relational) fluent $F(z_1, \dots, z_n, x, s)$ is treated as a multi-valued function:
 - ▶ z_1, \dots, z_n is the *input*;
 - ▶ x is the *output*.
- A *closure* ϕ implies complete information for the output of one or more fluents with a ground input:

$$\phi_1 : \forall x. \text{Near}(\text{bomb}, x, S_0) \equiv (x = \text{door} \vee x = \text{box}_1),$$

$$\phi_2 : \forall x. \text{Near}(\text{bomb}, x, S_0) \equiv (x = \text{door} \vee x = \text{box}_2).$$



1. KB is a finite database of possible closures (DBPC)

- Every (relational) fluent $F(z_1, \dots, z_n, x, s)$ is treated as a multi-valued function:
 - ▶ z_1, \dots, z_n is the *input*;
 - ▶ x is the *output*.
- A *closure* ϕ implies complete information for the output of one or more fluents with a ground input:

$$\phi_1 : \forall x. \text{Near}(\text{bomb}, x, S_0) \equiv (x = \text{door} \vee x = \text{box}_1),$$

$$\phi_2 : \forall x. \text{Near}(\text{bomb}, x, S_0) \equiv (x = \text{door} \vee x = \text{box}_2).$$

- A *possible closures axiom (PCA)* implies disjunctive information for the output of a fluent with a ground input:

$$\phi_1 \vee \phi_2.$$

- A *DBPC* is a set of PCAs that do not overlap.



2. DYN is range-restricted (RR)

- Consider the successor state axiom for $Status(x, y, s)$:

$$Status(x, y, do(a, s)) \equiv \gamma^+(x, y, a, s) \vee \\ (Status(x, y, s) \wedge \neg \gamma^-(x, y, a, s)),$$

- The axiom is RR iff each of the variables x, y is “*guarded*” by a *fluent atom* with a *ground input* or a *constant*:

- ▶ the following $\gamma^+(x, y, a, S_0)$ is *not* RR:

$$a = trigger(bomb) \wedge y = broken,$$

- ▶ the following $\gamma^+(x, y, a, S_0)$ is RR:

$$a = trigger(bomb) \wedge Near(bomb, x, s) \wedge y = broken$$

- The formal definition is inspired by the safe-range queries of the database theory and includes rules for \exists, \neg, \vee .



3. BAT is just-in-time (JIT) wrt the action

- A range-restricted DYN does *not* guarantee that there are finitely many ground fluents that may be affected by an action.
- Consider the following $\gamma^+(x, y, a, S_0)$ that is RR:

$$a = \text{trigger}(\text{bomb}) \wedge \text{Near}(\text{bomb}, x, s) \wedge y = \text{broken}.$$

- ▶ if there is *no PCA* for $\text{Near}(\text{bomb}, x, S_0)$ in the DBPC then $\text{trigger}(\text{bomb})$ may possibly affect the status of *all objects*;
- ▶ if the *PCA* $\phi_1 \vee \phi_2$ that we saw earlier is included, then only the following ground fluent atoms may be affected:

$$\begin{aligned} & \text{Status}(\text{door}, \text{broken}, S_0), \\ & \text{Status}(\text{box}_1, \text{broken}, S_0), \text{Status}(\text{box}_2, \text{broken}, S_0). \end{aligned}$$

- The JIT assumption captures the requirement that the DBPC includes all the necessary PCAs in the previous sense.



Result: a case of non-local-effect BAT progression

BAT : *finite, first-order, DBPC* KB + *range-restricted* DYN,
action *a* is executed, BAT is *just-in-time* wrt *a*.

BAT': *finite, first-order DBPC* KB' + *same* DYN.

Theorem: Let BAT be a theory that has a database of possible closures as a KB and a range-restricted DYN. Let *a* be a ground action such that BAT is just-in-time wrt *a*. Then, we can always specify a finite first-order representation of KB'.

- ▶ KB' can be obtained by updating the original KB in a systematic way similar to the local-effect case.
- ▶ The method is based on the notion of a *possible answer* to a formula wrt the DBPC.
- ▶ We provide an algorithm for the case that the axioms in DYN consist of conjunctive queries.



Conclusions

- We investigated the problem of *progression* of basic action theories in the *situation calculus*.
- A recent result shows that the *local-effect* assumption guarantees a finite first-order progression.
- As this assumption is often not practical, we considered a case where actions are non-local-effect but *range-restricted*.
- We showed that if we also assume that the knowledge base is a *database of possible closures* and the theory is *just-in-time* wrt the action, then the technique that works for local-effect actions can be used to obtain a finite first-order progression.
- We presented a simple algorithm for the core procedure of the progression method that works for a practical case.



Related and future work

- Recent work by Liu and Lakemeyer [2009] about the progression of non-local-effect theories:
 - ▶ assume a *proper*⁺ *KB* and *normal actions*.
- Both approaches have limitations:
 - ▶ an action *trigger(bomb)* that affects the status of the objects nearby as well as the fluent *Near(bomb, x)* is not normal;
 - ▶ our just-in-time assumption can be problematic in *offline* settings.
- In particular for *online* settings we intend to extend the current account of sensing, so that new possible closure axioms may be incorporated into the theory.
- We intend to investigate the general case for the core procedure of our progression method and identify cases where the size of new theory is practical.

