Answer Set Programming

Reasoning about Actions

Martin Baláž

Department of Applied Informatics
Comenius University in Bratislava

2009/2010

Outline

1. Semantics
   - Positive Planning Domains
   - General Planning Domains

2. Translation

Grounded Planning Domain

Definition (Legal Action and Fluent Instances)

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a planning domain and M be a unique stable model of Π.

An action (resp. fluent) instance $\theta(p(X_1, \ldots, X_n))$ is *legal* if there exists a θ-instance of an action (resp. fluent) declaration in D of a form

$$\texttt{p(}X_1\texttt{, ..., }X_n\texttt{) requires } t_1\texttt{, ..., } t_m$$

such that $M \models \{\theta(t_1), \ldots, \theta(t_m)\}$.

Example

occupied(a) is a legal fluent instance.
move$(table, b)$ is an action instance which is not legal.

State Transition

Definition (State)

A *state* is any consistent set of legal fluent instances and their negations.

Definition (State Transition)

A *state transition* is a tuple $\langle s, A, s' \rangle$ where s and s' are states and A is a set of legal action instances.

Example

$$\begin{aligned} s &= \{on(a, table), on(b, a), on(c, b)\} \\ A &= \{move(c, table)\} \\ s' &= \{on(a, table), on(b, a), on(c, table), -on(c, b)\} \end{aligned}$$

Initial State

Definition (Initial State)

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a positive planning domain and M be a unique stable model of Π.
A state s is an *initial state* if it is the least set such that for all initial state constraints in R of a form

$$\texttt{initially caused } f \texttt{ if } B$$

holds $s \cup M \models B \Rightarrow s \models f$.

Example

```
initially on(a, table).
initially on(b, a).
```

Executable Actions

Definition (Executable Action Set)

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a positive planning domain and M be a unique stable model of Π.
A set of legal action instances A is *executable* w.r.t. a state s if for all $a \in A$ there exists an executability condition in R of a form

$$\texttt{executable}\ a\ \texttt{if}\ C$$

such that $s \cup A \cup M \models C$.

Legal State Transition

Definition (Legal State Transition)

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a positive planning domain and M be a unique stable model of Π.
A state transition $\langle s, A, s' \rangle$ is *legal* if A is a legal action set executable in s and s' is the least set such that for all causation rules in r of a form

$$\texttt{caused}\ f\ \texttt{if}\ B\ \texttt{after}\ C$$

holds $s \cup A \cup M \models C \wedge s' \cup M \models B \Rightarrow s' \models f$.

Planning Domain Reduction

Definition (Reduction)

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a planning domain, M be a unique model of Π, and $t = \langle s, A, s' \rangle$ be a state transition.
A planning domain reduction of PD by t is a positive planning domain $PD^t = \langle \Pi, \langle D, R^t \rangle \rangle$ where R^t is obtained from R by deleting

- each $r \in R$ where $s' \cup M \not\models \{\sim b_{k+1}, \ldots, \sim b_l\}$
- each $r \in R$ where $s \cup A \cup M \not\models \{\sim c_{m+1}, \ldots, \sim c_n\}$
- remaining default literals

General Planning Domains

Definition

Let $PD = \langle \Pi, \langle D, R \rangle \rangle$ be a planning domain and M be a unique model of Π.

- A state s is an initial state of PD if s is an initial state of $PD^{\langle \emptyset, \emptyset, s \rangle}$.
- A set of action instances A is executable w.r.t. s in PD if A is executable w.r.t. s in $PD^{\langle s, A, \emptyset \rangle}$.
- A state transition $\langle s, A, s' \rangle$ is legal in PD if $\langle s, A, s' \rangle$ is legal in $PD^{\langle s, A, s' \rangle}$.

Definition (Legal Transition Sequence)

A sequence of legal state transitions $T = \langle \langle s_0, A_1, s_1 \rangle, \langle s_1, A_2, s_2 \rangle, \ldots, \langle s_{n-1}, A_n, s_n \rangle \rangle$, $0 \leq n$ is *legal* if s_0 is an initial state.

Plans

Definition (Optimistic Plan)

Let $\mathcal{P} = \langle PD, q \rangle$ be a planning problem.
A sequence of action sets $\langle A_1, \ldots, A_n \rangle$, $n \geq 0$, is an *optimistic plan* if there exists a legal transition sequence
$T = \langle \langle s_0, A_1, s_1 \rangle, \ldots, \langle s_{n-1}, A_n, s_n \rangle \rangle$, $0 \leq n$ such that $s_n \models q$.

Definition (Secure Plan)

An optimistic plan $\langle A_1, \ldots, A_n \rangle$, $n \geq 0$, is *secure* if for every initial state s_0 and legal transition sequence
$T = \langle \langle s_0, A_1, s_1 \rangle, \ldots, \langle s_{m-1}, A_m, s_m \rangle \rangle$, $0 \leq m \leq n$ either $m = n$ and $s_m \models q$ or $m < n$ and there exists some legal transition $\langle s_m, A_{m+1}, s_{m+1} \rangle$.

Translation

0. Macro expansion
1. Background knowledge
2. Auxiliary predicates
3. Causation rules
4. Executability conditions
5. Initial state constraints
6. Goal query

Auxiliary predicates

$$\langle\langle s_0, A_1, s_1\rangle, \ldots, \langle s_{n-1}, A_n, s_n\rangle\rangle$$

Example (Translation)

```
time(0).
...
time(n).
next(0, 1).
...
next(n-1, n).
actiontime(0).
...
actiontime(n-1).
```

Causation Rules

`caused` f `if` B `after` C

- fluent atom f and all fluent atoms from B are expanded with additional parameter T_1
- if $f = \mathit{false}$, the resulting rule is a constraint
- all action and fluent atoms from C are expanded with additional parameter T_0
- type atoms remain unchanged
- we add $\mathit{time}(T_1)$ to the body, if A is empty, $\mathit{next}(T_0, T_1)$ otherwise
- to make a rule safe, for a fluent literal f and for default negated action and fluent literals from $B \cup C$, we add typing information from corresponding declaration to the body

Causation Rules

Example

```
fluents: on(B, L) requires block(B), location(L).
         occupied(B) requires block(B).
actions: move(B, L) requires block(B), location(L).
always:  caused occupied(B) if on(B1, B).
         caused on(B, L) after move(B, L).
```

becomes

Example

```
occupied(B, T1) :- on(B1, B, T1), block(B), time(T1).
   on(B, L, T1) :- move(B, L, T0), block(B),
                   location(L), next(T0, T1).
```

Executability Conditions

`executable` a `if` C

- the head is $a \vee \neg a$ expanded with additional parameter T_0
- all action and fluent atoms from C are expanded with additional parameter T_0
- type atoms remain unchanged
- we add $actiontime(T_0)$ to the body
- to make a rule safe, for an action literal a and for default negated action and fluent literals from C, we add typing information from corresponding declaration to the body

Executability Conditions

Example

```
actions: move(B, L) requires block(B), location(L).
always:  executable move(B, L) if B <> L.
```

becomes

Example

```
move(B, L, T0) v -move(B, L, T0) :- B <> L,
     block(B), location(L), actiontime(T0).
```

Initial State Constraints

`initially caused` f `if` B.

Like static causation rules, but with $T_1 = 0$.

Example

```
initially: on(a, table). on(b, table). on(c, a).
```

becomes

Example

```
on(a, table, 0).
on(b, table, 0).
on(c, a, 0).
```

Goal Query

g_1, ..., g_m, not g_{m+1}, ..., g_n?

- the head is a new predicate symbol `goal`
- the body is g_1, ..., g_m, not g_{m+1}, ..., g_n expanded with new additional parameter i (the length of a plan)
- new constraint `:- goal` is added

Example

```
goal: on(c, b), on(b, a), on(a, table)?
```

Example

```
goal :- on(c, b, 3), on(b, a, 3), on(a, table, 3).
     :- not goal.
```

Completeness and Correctness

Theorem

Let $\mathcal{P}$ be a planning problem and let $lp(\mathcal{P})$ be the generated logic program. For any interpretation S and $j \geq 0$ we define

$$\begin{aligned} A_j^S &= \{a(t) \mid a(t, j-1) \in S, a(t) \text{ is an action atom}\} \\ s_j^S &= \{f(t) \mid f(t, j) \in S, f(t) \text{ is a fluent literal}\} \end{aligned}$$

For each optimistic plan $P = \langle A_1, \ldots, A_n \rangle$ and witnessing sequence $T = \langle \langle s_0, A_1, s_1 \rangle, \ldots, \langle s_{n-1}, A_n, s_n \rangle \rangle$ there exists an answer set S such that $A_j = A_j^S$, $0 < j \leq n$ and $s_j = S_j^S$, $0 \leq j \leq n$.

For each answer set S, $P = \langle A_1, \ldots, A_n \rangle$ is an optimistic plan witnessed by sequence $T = \langle \langle s_0, A_1, s_1 \rangle, \ldots, \langle s_{n-1}, A_n, s_n \rangle \rangle$ where $A_j = A_j^S$, $0 < j \leq n$ and $s_j = s_j^S$, $0 \leq j \leq n$.