Warsaw University of Technology's
Faculty of Mathematics and Information Science

Knowledge Representation and Reasoning

Project number 2:
Deterministic Action With Cost
Supervisor: Dr Anna Radzikowska

Created By
Rishabh Jain, Rahul Tomer, Kuldeep Shankar,
Alaa Abboushi, Haran Dev Murugan,
Bui Tuan Anh.

Contents

1 Introduction

A dynamic system (DS) is viewed as

- a collection of objects, together with their properties, and
- a collection of actions which, while performed, change properties of objects (in consequence, the state of the world).

Let C2 be a class of dynamic systems satisfying the following assumptions:

1. Inertia law
2. Complete information about all actions and fluent.
3. Only Determinism
4. Only sequential actions are allowed.
5. Characterizations of actions:
   - Precondition represented by set of literals(a fluent or its negation);if a precondition does not hold, the action is executed but with empty effect
   - Postcondition (effect of an action) represented by a set of literals.
   - Cost $k \in N$ of an action, actions with empty effects cost 0. Each action has a fixed cost, if it leads to non-empty effects.
6. Effects of an action depends on the state where the action starts.
7. All actions are performed in all states.
8. Partial description of any state of the system are allowed.
9. No constraints are defined.

2 Syntax

A system is defined by a set of fluent **F**, actions **Ac** and Cost $\mathbf{k} \in \mathbf{N}$ and characterized by signature **(F, Ac, k)**
A formula is any propositional combination of fluent:
$\alpha :: \neg\alpha \;|\alpha \wedge \beta \;|\alpha \vee \beta \;|\alpha \rightarrow \beta$
Two specific formulas:

1. $\top$: truth

2. $\perp$: falsity

The system and changes occurring within can be described through a sequence of statements defined in the table:

Statement	Format	Description
Initial Statement	Initially α	Initial condition α of the fluent.
Effect Statement	Ac costing k causes α	Perform action Ac for cost k in any state leads to the effect α.
Value Statement	α after A1 ....An	The condition α always (must) hold after performing the sequence A1 ....An of actions.
Cost Statement	Cost of α after A1 ....An	The condition α holds a constant cost for performing any action in the sequence A1 ....An of actions.

Table 1: Syntax Table

3 Semantics

- A state is a mapping σ : F $\rightarrow$ 0, 1. For any f $\in$ F, if σ(f) = 1, then we say that f holds in σ and write $\sigma \models$ f. If σ(f) = 0, then we write σ $\models \neg$f and say that f does not hold in σ. Let σ stand for the set of all states.

- A state is a mapping σ : F $\rightarrow$ 0, 1. For any f $\in$ F, if σ(f) = 1, then we say that f holds in σ and write $\sigma \models$ f. If σ(f) = 0, then we write σ $\models \neg$f and say that f does not hold in σ. Let σ stand for the set of all states.

- A transition function is a mapping Υ : Ac $\times$ $\sigma \rightarrow \sigma$. For any $\sigma \in \sigma$ and for any A $\in$ Ac, Υ(A, σ) is the state resulting from performing the action A in the state σ.

- A transition function is generalized to the mapping $\Upsilon *$: Ac $*$ $\times$ $\sigma \rightarrow \sigma$ as follows: $\Upsilon*$ (ε , σ) = σ, $\Upsilon*$ ((A1, . . . , An), σ) = Υ(An, $\Upsilon*$ (A1, . . . , An-1)).

- A transition function is generalized to the mapping $\Upsilon* : Ac * \times \sigma \rightarrow \sigma$ as follows: $\Upsilon*$ (ε , σ) = σ, $\Upsilon*$ ((A1, . . . , An), σ) = Υ(An, $\Upsilon*$ (A1, . . . , An-1)).
- Let L be an action language of the class A over the signature Y = (F, Ac). A structure for L is a pair S = (Υ, σ0) where Υ is a transition function and $\sigma 0 \in \sigma$ is the initial state
- Let s = (Υ, σ0) be a structure for L. A statement s is true in S, in symbols S $\vDash$ s, iff if s is of the form f after A1, . . . , An, then Υ((A1, . . . , An), σ0) $\vDash$ f;
- Let s = (Υ, σ0) be a structure for L. A statement s is true in S, in symbols S $\vDash$ s, iff if s is of the form f after A1, . . . , An, then Υ((A1, . . . , An), σ0) $\vDash$ f; if s is of the form A causes f if g1, . . . , gk, then for every $\sigma \in \sigma$ such that $\sigma \vDash$ gi , i = 1, . . . , k, Υ(A, σ) $\vDash$ f.

Let D be an action domain in the language L over the signature Υ=(F,A_c).
A structure S = (Ψ,σ0,k) is a model of D iff
(M1) for every s $\in$ D, S |= s;
(M2) for every A $\in$ Ac, for every f,g1,...,gn $\in$ F, and for every $\sigma \in \sum$, if one of the following conditions holds:

(i) D contains an effect statement

A causes $\overline{f}$ for the cost k if $\overline{g}$,...,$\overline{gn}$,

where k $\neq$ 0 and σ|$\neq$ gi for some i = 1,...,n

(ii) D does not contain an effect statement

A causes $\overline{f}$ if g1,...,gn

then σ |= f iff Ψ(A,σ,k) |= f. where k = 0.

4 Examples

4.1 Example 01

4.1.1 Description

Andrew wants to travel by his car to a place.Travelling costs him 100$ if he uses fuel from the fuel tank of the car.If in case of emergency, Andrew is carrying a bottle of fuel as reserve, which can cost him 150$ for travelling because its a low quality fuel. Buying fuel costs him 200$ and reserve costs him 250$.

4.1.2 Representation

Initially we have:

1. Fuel
2. Reserve

Travel causes ¬fuel if fuel ∨ reserve
Travel causes ¬reserve if ¬fuel ∨ reserve
BuyF causes fuel if ¬fuel
BuyS causes reserve if ¬reserve

4.1.3 Calculation

$\sum = \{ \sigma_0, \sigma_1, \sigma_2, \sigma_3 \}$
$\sigma_0 = \{$ fuel, reserve $\}$ $\sigma_1 = \{$ ¬fuel, reserve $\}$
$\sigma_2 = \{$ ¬fuel, ¬reserve $\}$ $\sigma_3 = \{$ fuel, ¬reserve $\}$

$\Psi(\text{BuyF},\sigma_0)=\sigma_0$ $\Psi(\text{BuyS},\sigma_0)=\sigma_0$
$\Psi(\text{BuyF},\sigma_1)=\sigma_0$ $\Psi(\text{BuyS},\sigma_1)=\sigma_1$
$\Psi(\text{BuyF},\sigma_2)=\sigma_3$ $\Psi(\text{BuyS},\sigma_2)=\sigma_1$
$\Psi(\text{BuyF},\sigma_3)=\sigma_3$ $\Psi(\text{BuyS},\sigma_3)=\sigma_0$

$\Psi(\text{Travel},\sigma_0)=\sigma_1$ $\Psi(\text{Travel},\sigma_1)=\sigma_2$
$\Psi(\text{Travel},\sigma_2)=\sigma_2$ $\Psi(\text{Travel},\sigma_3)=\sigma_2$

4.1.4 Graph

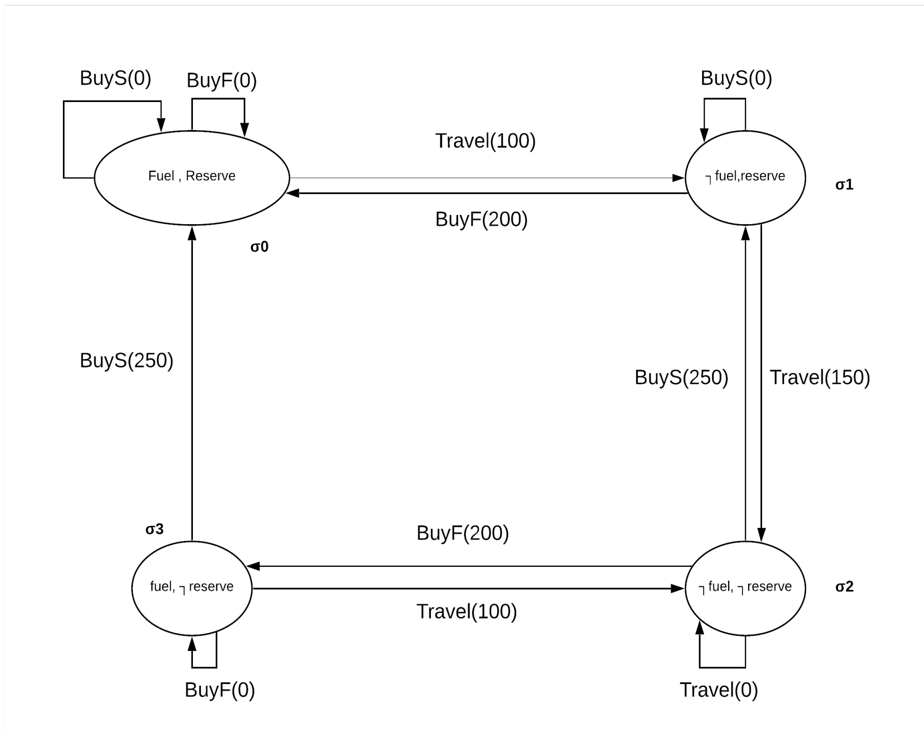

Figure 1: Example 01

4.2 Example 02

4.2.1 Description

John visits a painter to buy a specific painting. The cost of painting is 200$ if its available in the shop. But if painting is not available then John needs to order a new one to be painted and will buy once its available. At any time only one copy of painting is available and another one to be ordered once sold.

4.2.2 Representation:

Fluents: available, sold.

Actions: BUY, ORDER.
BUY costs 200$ **if** available
BUY costs 0$ **if** ¬available
Order Cost: 0 $
initially: ¬available ∧ ¬sold
BUY causes sold if available
ORDER causes available if ¬available
BUY causes ¬available

4.2.3 Calculation:

$\sum$ = { $\sigma 0$, $\sigma 1$, $\sigma 2$, $\sigma 3$}
$\sigma 0$ = { ¬available, ¬sold}
$\sigma 1$ = { ¬available, sold}
$\sigma 2$ = { available, ¬sold}
$\sigma 3$ = { available, sold}
Ψ (BUY, $\sigma 0$) = $\sigma 0$
Ψ (ORDER, $\sigma 0$) = $\sigma 1$
Ψ (BUY, $\sigma 1$) = $\sigma 2$
Ψ (ORDER, $\sigma 1$) = $\sigma 1$
Ψ (BUY, $\sigma 2$) = $\sigma 2$
Ψ (ORDER, $\sigma 2$) = $\sigma 1$
Ψ (BUY, $\sigma 3$) = $\sigma 2$
Ψ (ORDER, $\sigma 3$) = $\sigma 3$

4.2.4 Graph

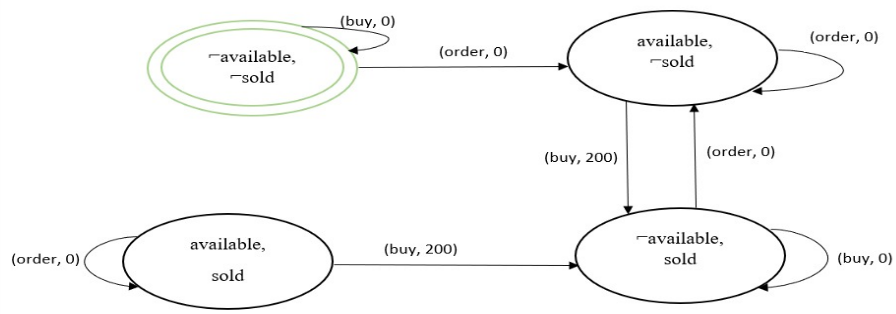

Figure 2: Example 02

4.3 Example 03

4.3.1 Description

There is a man. He can cook, eat, and play. Cooking makes food cooked. he can eat food if it is cooked. After eating he feels not hungry, and food is not cooked again. He can play. Playing makes him hungry. He just can play if he is not hungry. He just cooks when there is no food is cooked. Initially, he is hungry, and no food is cooked. In terms of energy, eating costs 5, cooking costs 15, playing costs 20.

4.3.2 Representation in language

Fluents: cooked, hungry.
Actions: cook, eat, play.

eat cost 5
cooking cost 15
play cost 20
initially $\neg$cooked $\wedge$ hungry
cook causes cook if $\neg$cooked

eat causes (¬cooked ∧ ¬hungry) if cooked
play causes hungry if ¬hungry

4.3.3 Calculation

$\sum = \{\sigma 0\ ,\sigma 1,\ \sigma 2,\ \sigma 3\}$

$\sigma 0 = \{\neg\text{cooked, hungry}\}$
$\sigma 1 = \{\text{cooked, hungry}\}$
$\sigma 2 = \{\neg\text{cooked}, \neg\text{hungry}\}$
$\sigma 3 = \{\text{cooked}, \neg\text{hungry}\}$

$\psi(\text{eat}, \sigma 0) = \sigma 0$
$\psi(\text{cook}, \sigma 0) = \sigma 1$
$\psi(\text{play}, \sigma 0) = \sigma 0$

$\psi(\text{eat}, \sigma 1) = \sigma 2$
$\psi(\text{cook}, \sigma 1) = \sigma 1$
$\psi(\text{play}, \sigma 1) = \sigma 1$

$\psi(\text{eat}, \sigma 2) = \sigma 2$
$\psi(\text{cook}, \sigma 2) = \sigma 3$
$\psi(\text{play}, \sigma 2) = \sigma 1$

$\psi(\text{eat}, \sigma 3) = \sigma 2$
$\psi(\text{cook}, \sigma 3) = \sigma 3$
$\psi(\text{play}, \sigma 3) = \sigma 1$

4.3.4 Graph

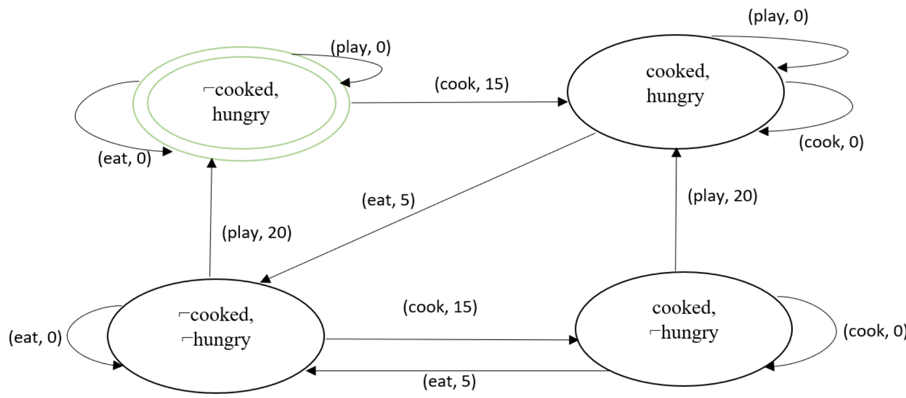

Figure 3: Example 03

5 Appendix

List of Figures

List of Tables