Situation Calculus

Dr. Neil T. Dantam

CSCI-498/598 RPM, Colorado School of Mines

Spring 2018

Outline

Logical Calculi

Propositional Calculus:

- Boolean variables (propositions)
- Logical Operators ($\wedge$, $\vee$, $\neg$, $\implies$, $\iff$, $\oplus$)

Predicate Calculus: Extends the propositional calculus with:

- Objects
- Predicates
- Functions
- Quantifiers

Situation Calculus: Extends the predicate calculus to model actions that change state:

- Fluents
- Actions

Situation Calculus

Predicate Calculus + changing state:

Fluents

- Synonym for state variables of the system
- Example:
  - `closed(suitcase)`
  - `contains(suitcase, laptop)`
- From Latin *fluere* meaning "to flow."

Actions

- **Elements:**

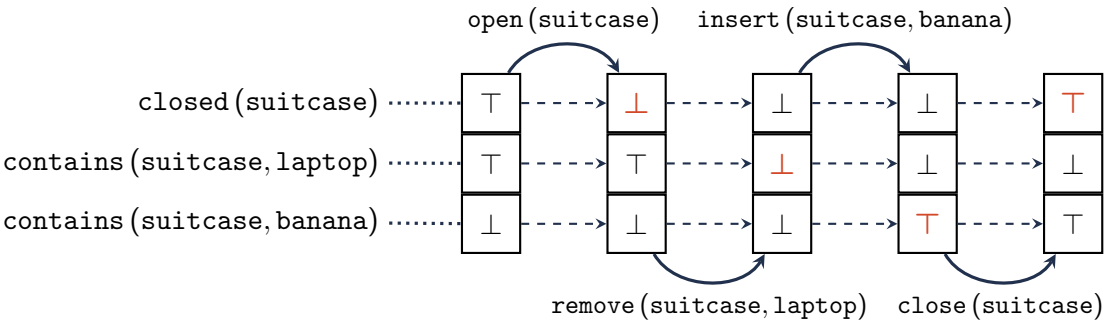
  Label: Name / arguments
  Precondition: States where the action is valid
  Effect: Result of the action

- **Example:**

  Label: `open(suitcase)`
  Precondition: `closed(suitcase)`
  Effect: $\neg$`closed(suitcase)`

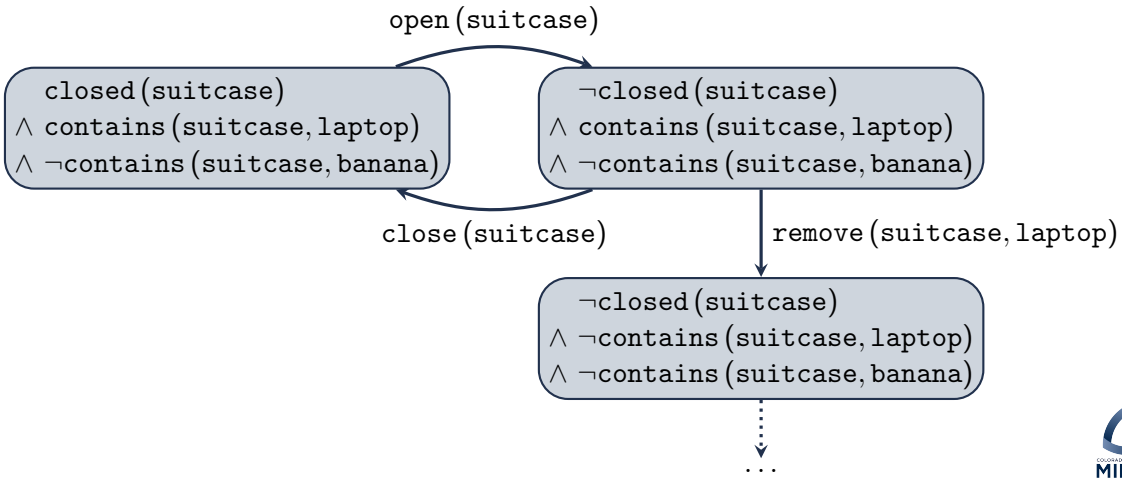
Illustration

State/Action Sequence



Illustration

Automaton



Exercise: State Space Size

Objects:
- $C = \{\texttt{suitcase}, \texttt{backpack}\}$
- $B = \{\texttt{laptop}, \texttt{banana}, \texttt{book}\}$

Predicate: $\texttt{contains} : C \times B \mapsto \mathbb{B}$

Fluents:

Transition System

State Space: $\mathcal{Q} = f_0 \times f_1 \times \ldots \times f_m$, for each fluent f_i

Actions: $\mathcal{U} = \{a_0, \ldots, a_n\}$

Transitions: $\delta : \mathcal{Q} \times \mathcal{U} \mapsto \mathcal{Q}$,
where for $\delta(q_0, a) = q_1$,

- q_0 satisfies the precondition of a
- q_1 is the effect of a applied to q_0

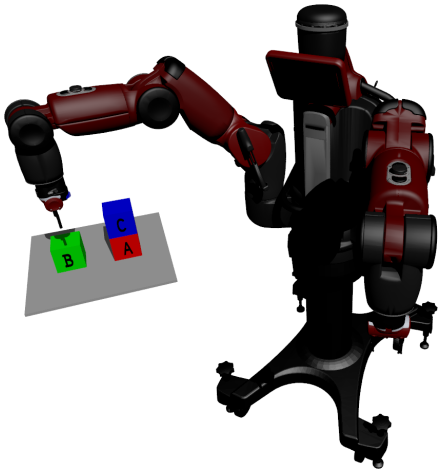
Start: $q_0 \in \mathcal{Q}$ is the initial state

Goal: $G \subseteq \mathcal{Q}$ is the set of goal states

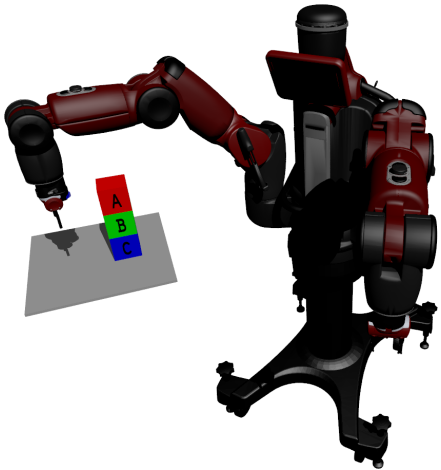
Outline

A Planning Problem

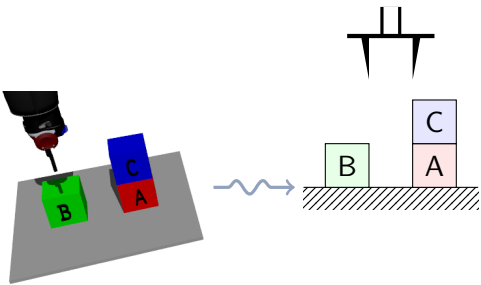
Start



Goal



First-Order Logic Description

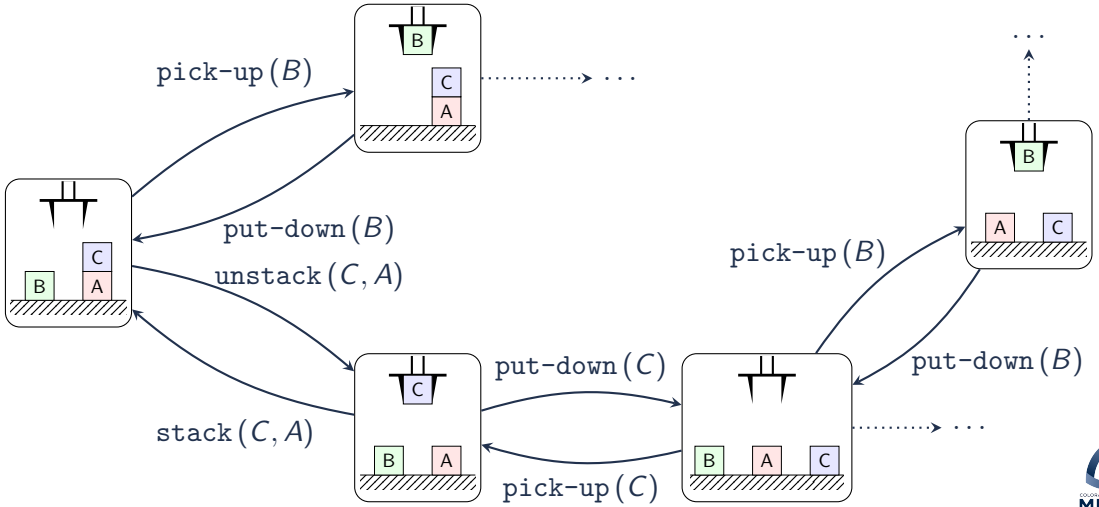


Constants: A, B, C

Predicates:
- $\texttt{on}(?x, ?y)$
- $\texttt{clear}(?x)$
- $\texttt{ontable}(?x)$
- $\texttt{handempty}()$

Fluents:
- $\texttt{clear}(B)$
- $\texttt{clear}(C)$
- $\texttt{ontable}(B)$
- $\texttt{ontable}(A)$
- $\texttt{handempty}()$

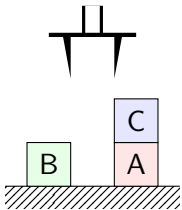
Task Language



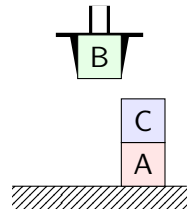
Example: pick-up(?x)

Precondition: $\texttt{ontable}(?x) \wedge \texttt{clear}(?x) \wedge \texttt{handempty}()$

Effect: $\neg\texttt{ontable}(?x) \wedge \neg\texttt{clear}(?x) \wedge \neg\texttt{handempty}() \wedge \texttt{holding}(?x)$



pick-up(B)



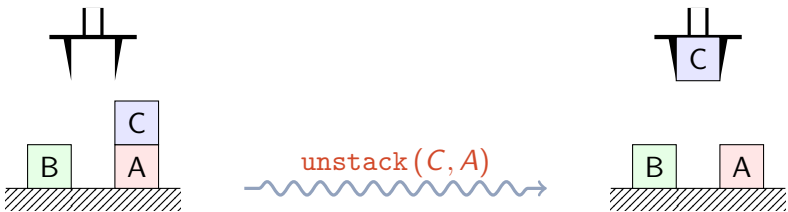
$\texttt{ontable}(B) \wedge \texttt{ontable}(A)$
$\wedge\ \texttt{on}(C, A)$
$\wedge\ \texttt{clear}(B) \wedge \texttt{clear}(C)$
$\wedge\ \texttt{handempty}()$

~~$\texttt{ontable}(B) \wedge$~~ $\texttt{ontable}(A)$
$\wedge\ \texttt{on}(C, A)$
~~$\wedge\ \texttt{clear}(B)$~~ $\wedge\ \texttt{clear}(C)$
~~$\wedge\ \texttt{handempty}()$~~
$\wedge\ \texttt{holding}(B)$

Exercise: $\texttt{unstack}(?x, ?y)$

Precondition: $\texttt{on}(?x, ?y) \wedge \texttt{clear}(?x) \wedge \texttt{handempty}()$

Effect: $\neg\texttt{on}(?x, ?y) \wedge \neg\texttt{clear}(?x) \wedge \neg\texttt{handempty}() \wedge \texttt{holding}(?x) \wedge \texttt{clear}(?y)$



$\texttt{ontable}(B) \ \wedge \ \texttt{ontable}(A)$
$\wedge \ \texttt{on}(C, A)$
$\wedge \ \texttt{clear}(B) \ \wedge \ \texttt{clear}(C)$
$\wedge \ \texttt{handempty}()$

Outline

Example: pick-up(?x)

pick-up(?x)

Precondition: ontable(?x)
∧ clear(?x)
∧ handempty()

Effect: ¬ontable(?x)
∧ ¬clear(?x)
∧ ¬handempty()
∧ holding(?x)

PDDL

```
(:action pick-up
         :parameters (?x)
         :precondition (and (ontable ?x)
                            (clear ?x)
                            (handempty))
         :effect (and (not (ontable ?x))
                      (not (clear ?x))
                      (not (handempty))
                      (holding ?x)))
```

Exercise: unstack(?x, ?y)

unstack(?x, ?y)

Precondition: on(?x, ?y)
∧ clear(?x)
∧ handempty()

Effect: ¬on(?x, ?y)
∧ ¬clear(?x)
∧ ¬handempty()
∧ holding(?x)
∧ clear(?y)

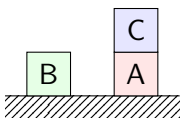
PDDL

```
(:action unstack
         :parameters (?x ?y)
         :precondition (and (on ?x ?y)
                            (clear ?x)
                            (handempty))
         :effect (and (not (on ?x ?y))
                      (not (clear ?x))
                      (not (handempty))
                      (holding ?x)
                      (clear ?y)))
```

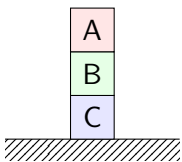
Full Operators File

Example: PDDL Facts

Start



Goal

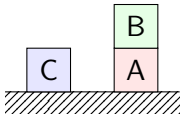


PDDL

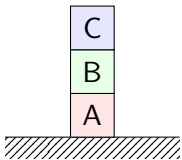
```
(define
 (problem sussman-anomaly)
  (:domain blocks)
  (:objects a b c)
  (:init (on c a)
         (ontable a)
         (ontable b)
         (clear c)
         (clear b)
         (handempty))
  (:goal (and (on b c)
              (on a b))))
```

Exercise: PDDL Facts

Start



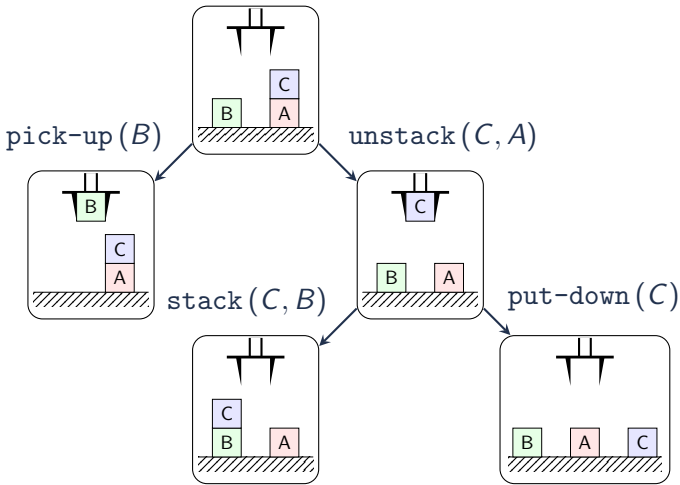
Goal



PDDL

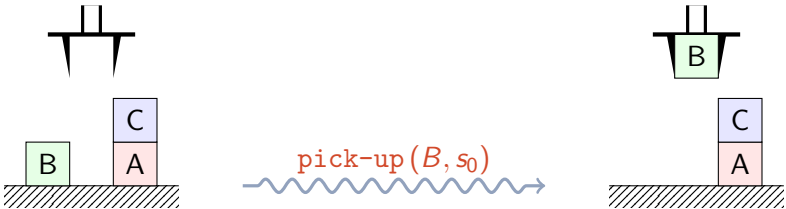
Outline

Heuristic Search



Constraint-Based Planning

aka SATPlan



$$\texttt{pick-up}(B, s_0) \implies \overbrace{\texttt{ontable}(?x, s_0) \wedge \texttt{clear}(?x, s_0) \wedge \texttt{handempty}(s_0)}^{\text{precondition at step } i}$$
$$\wedge \underbrace{\neg\texttt{ontable}(?x, s_1) \wedge \neg\texttt{clear}(?x, s_1) \wedge \neg\texttt{handempty}(s_1) \wedge \texttt{holding}(?x, s_1)}_{\text{effect at step } i+1}$$

Summary