

Planning and Optimization

A4. Propositional Planning Tasks

Gabriele Röger and Thomas Keller

Universität Basel

October 1, 2018

Planning and Optimization

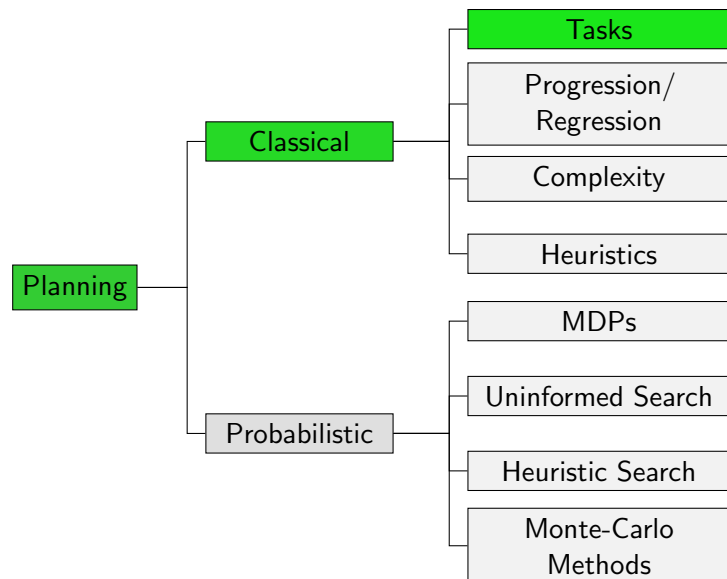
October 1, 2018 — A4. Propositional Planning Tasks

A4.1 Introduction

A4.2 Operators

A4.3 Propositional Planning Tasks

Content of this Course



A4.1 Introduction

State Variables

How to specify huge transition systems without enumerating the states?

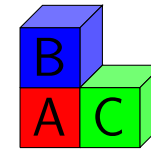
- ▶ represent different aspects of the world in terms of different **Boolean state variables**
- ▶ treat state variables as atomic propositions
 \rightsquigarrow a state is a **valuation of state variables**
- ▶ n state variables induce 2^n states
 \rightsquigarrow **exponentially more compact** than “flat” representations

Example: $O(n^2)$ variables suffice for blocks world with n blocks

Blocks World State with Boolean State Variables

Example

$s(A\text{-on-}B) = \mathbf{F}$
 $s(A\text{-on-}C) = \mathbf{F}$
 $s(A\text{-on-table}) = \mathbf{T}$
 $s(B\text{-on-}A) = \mathbf{T}$
 $s(B\text{-on-}C) = \mathbf{F}$
 $s(B\text{-on-table}) = \mathbf{F}$
 $s(C\text{-on-}A) = \mathbf{F}$
 $s(C\text{-on-}B) = \mathbf{F}$
 $s(C\text{-on-table}) = \mathbf{T}$



Boolean State Variables

Problem:

- ▶ How to **succinctly** represent **transitions** and **goal states**?

Idea: Use **logical formulas** to describe sets of states

- ▶ **state variables:** atomic propositions
- ▶ **states:** all valuations of the state variables
- ▶ **goal states:** defined by a logical formula
- ▶ **transitions:** defined by **operators** (see following section)

A4.2 Operators

Syntax of Operators

Definition (Operator)

An **operator** o over state variables V is an object with three properties:

- ▶ a **precondition** $pre(o)$, a logical formula over V
- ▶ an **effect** $eff(o)$ over V , defined on the following slides
- ▶ a **cost** $cost(o) \in \mathbb{R}_0^+$

Notes:

- ▶ Operators are also called **actions**.
- ▶ Operators are often written as triples $\langle pre(o), eff(o), cost(o) \rangle$.
- ▶ This can be abbreviated to pairs $\langle pre(o), eff(o) \rangle$ when the cost of the operator is irrelevant.

Operators: Intuition

Intuition for operators o :

- ▶ The operator precondition describes the set of states in which a transition labeled with o can be taken.
- ▶ The operator effect describes how taking such a transition changes the state.
- ▶ The operator cost describes the cost of taking a transition labeled with o .

Syntax of Effects

Definition (Effect)

Effects over state variables V are inductively defined as follows:

- ▶ If $v \in V$ is a state variable, then v and $\neg v$ are effects (**atomic effect**).
- ▶ If e_1, \dots, e_n are effects, then $(e_1 \wedge \dots \wedge e_n)$ is an effect (**conjunctive effect**).
The special case with $n = 0$ is the **empty effect** \top .
- ▶ If χ is a logical formula and e is an effect, then $(\chi \triangleright e)$ is an effect (**conditional effect**).

Parentheses can be omitted when this does not cause ambiguity.

Effects: Intuition

Intuition for effects:

- ▶ **Atomic effects** v and $\neg v$ can be understood as assignments " $v := \mathbf{T}$ " and " $v := \mathbf{F}$ ".
- ▶ A **conjunctive effect** $e = (e_1 \wedge \dots \wedge e_n)$ means that all subeffects e_1, \dots, e_n take place simultaneously.
- ▶ A **conditional effect** $e = (\chi \triangleright e')$ means that subeffect e' takes place iff χ is true in the state where e takes place.

Semantics of Effects

Definition (Effect Condition for an Effect)

Let $\ell = v$ or $\ell = \neg v$ be an atomic effect.

The **effect condition** $effcond(\ell, e)$ under which ℓ triggers given the effect e is a propositional formula defined as follows:

- ▶ $effcond(\ell, \ell) = \top$
- ▶ $effcond(\ell, \ell') = \perp$ for atomic effects $\ell' \neq \ell$
- ▶ $effcond(\ell, (e_1 \wedge \dots \wedge e_n)) = effcond(\ell, e_1) \vee \dots \vee effcond(\ell, e_n)$
- ▶ $effcond(\ell, (\chi \triangleright e)) = \chi \wedge effcond(\ell, e)$

Intuition: $effcond(\ell, e)$ represents the condition that must be true in the current state for the effect e to lead to the atomic effect ℓ

Semantics of Operators

Definition (Applicable, Resulting State)

Let V be a set of state variables.

Let s be a state over V , and let o be an operator over V .

Operator o is **applicable** in s if $s \models pre(o)$.

If o is applicable in s , the **resulting state** of applying o in s , written $s[o]$, is the state s' defined as follows for all $v \in V$:

$$s'(v) = \begin{cases} \mathbf{T} & \text{if } s \models effcond(v, e) \\ \mathbf{F} & \text{if } s \models effcond(\neg v, e) \wedge \neg effcond(v, e) \\ s(v) & \text{if } s \not\models effcond(v, e) \vee effcond(\neg v, e) \end{cases}$$

where $e = eff(o)$.

Add-after-Delete Semantics

Note:

- ▶ The definition implies that if a variable is simultaneously “added” (set to \mathbf{T}) and “deleted” (set to \mathbf{F}), the value \mathbf{T} takes precedence.
- ▶ This is called **add-after-delete semantics**.
- ▶ This detail of semantics is somewhat arbitrary, and other definitions are sometimes used.
- ▶ We will later also consider **conflict semantics** where “contradictory” effects are forbidden.

Applying Operators: Example

Example

Consider the operator $o = \langle a, \neg a \wedge (\neg c \triangleright \neg b) \rangle$ and the state $s = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

The operator o is applicable in s because $s \models a$.

Effect conditions of $eff(o)$:

$$\begin{aligned} effcond(a, eff(o)) &= effcond(a, \neg a \wedge (\neg c \triangleright \neg b)) \\ &= effcond(a, \neg a) \vee effcond(a, \neg c \triangleright \neg b) \\ &= \perp \vee (\neg c \wedge effcond(a, \neg b)) \\ &= \perp \vee (\neg c \wedge \perp) \\ &\equiv \perp \quad \rightsquigarrow \text{false in state } s \end{aligned}$$

Applying Operators: Example

Example

Consider the operator $o = \langle a, \neg a \wedge (\neg c \triangleright \neg b) \rangle$
and the state $s = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

The operator o is applicable in s because $s \models a$.

Effect conditions of $\text{eff}(o)$:

$$\begin{aligned} \text{effcond}(\neg a, \text{eff}(o)) &= \text{effcond}(\neg a, \neg a \wedge (\neg c \triangleright \neg b)) \\ &= \text{effcond}(\neg a, \neg a) \vee \text{effcond}(\neg a, \neg c \triangleright \neg b) \\ &= \mathbf{T} \vee \text{effcond}(\neg a, \neg c \triangleright \neg b) \\ &\equiv \mathbf{T} \quad \rightsquigarrow \text{true in state } s \end{aligned}$$

Applying Operators: Example

Example

Consider the operator $o = \langle a, \neg a \wedge (\neg c \triangleright \neg b) \rangle$
and the state $s = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

The operator o is applicable in s because $s \models a$.

Effect conditions of $\text{eff}(o)$:

$$\begin{aligned} \text{effcond}(b, \text{eff}(o)) &= \text{effcond}(a, \neg a \wedge (\neg c \triangleright \neg b)) \\ &= \text{effcond}(b, \neg a) \vee \text{effcond}(b, \neg c \triangleright \neg b) \\ &= \perp \vee (\neg c \wedge \text{effcond}(b, \neg b)) \\ &= \perp \vee (\neg c \wedge \perp) \\ &\equiv \perp \quad \rightsquigarrow \text{false in state } s \end{aligned}$$

Applying Operators: Example

Example

Consider the operator $o = \langle a, \neg a \wedge (\neg c \triangleright \neg b) \rangle$
and the state $s = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

The operator o is applicable in s because $s \models a$.

Effect conditions of $\text{eff}(o)$:

$$\begin{aligned} \text{effcond}(\neg b, \text{eff}(o)) &= \text{effcond}(\neg b, \neg a \wedge (\neg c \triangleright \neg b)) \\ &= \text{effcond}(\neg b, \neg a) \vee \text{effcond}(\neg b, \neg c \triangleright \neg b) \\ &= \perp \vee (\neg c \wedge \text{effcond}(\neg b, \neg b)) \\ &= \perp \vee (\neg c \wedge \mathbf{T}) \\ &\equiv \neg c \quad \rightsquigarrow \text{false in state } s \end{aligned}$$

Applying Operators: Example

Example

Consider the operator $o = \langle a, \neg a \wedge (\neg c \triangleright \neg b) \rangle$
and the state $s = \{a \mapsto \mathbf{T}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

The operator o is applicable in s because $s \models a$.

Effect conditions of $\text{eff}(o)$:

$$\begin{aligned} \text{effcond}(c, \text{eff}(o)) &\equiv \perp \quad \rightsquigarrow \text{false in state } s \\ \text{effcond}(\neg c, \text{eff}(o)) &\equiv \perp \quad \rightsquigarrow \text{false in state } s \\ \text{effcond}(d, \text{eff}(o)) &\equiv \perp \quad \rightsquigarrow \text{false in state } s \\ \text{effcond}(\neg d, \text{eff}(o)) &\equiv \perp \quad \rightsquigarrow \text{false in state } s \end{aligned}$$

The resulting state of applying o in s is the state
 $\{a \mapsto \mathbf{F}, b \mapsto \mathbf{T}, c \mapsto \mathbf{T}, d \mapsto \mathbf{T}\}$.

Example Operators: Blocks World

Example (Blocks World Operators)

To model blocks world operators conveniently, we use auxiliary state variables A -clear, B -clear, and C -clear to express that there is nothing on top of a given block.

Then blocks world operators can be modeled as:

- ▶ $\langle A\text{-clear} \wedge A\text{-on-table} \wedge B\text{-clear}, A\text{-on-B} \wedge \neg A\text{-on-table} \wedge \neg B\text{-clear} \rangle$
- ▶ $\langle A\text{-clear} \wedge A\text{-on-table} \wedge C\text{-clear}, A\text{-on-C} \wedge \neg A\text{-on-table} \wedge \neg C\text{-clear} \rangle$
- ▶ $\langle A\text{-clear} \wedge A\text{-on-B}, A\text{-on-table} \wedge \neg A\text{-on-B} \wedge B\text{-clear} \rangle$
- ▶ $\langle A\text{-clear} \wedge A\text{-on-C}, A\text{-on-table} \wedge \neg A\text{-on-C} \wedge C\text{-clear} \rangle$
- ▶ $\langle A\text{-clear} \wedge A\text{-on-B} \wedge C\text{-clear}, A\text{-on-C} \wedge \neg A\text{-on-B} \wedge B\text{-clear} \wedge \neg C\text{-clear} \rangle$
- ▶ $\langle A\text{-clear} \wedge A\text{-on-C} \wedge B\text{-clear}, A\text{-on-B} \wedge \neg A\text{-on-C} \wedge C\text{-clear} \wedge \neg B\text{-clear} \rangle$
- ▶ ...

Example Operator: 4-Bit Counter

Example (Incrementing a 4-Bit Counter)

Operator to increment a 4-bit number $b_3b_2b_1b_0$ represented by 4 state variables b_0, \dots, b_3 :

precondition:

$$\neg b_0 \vee \neg b_1 \vee \neg b_2 \vee \neg b_3$$

effect:

$$\begin{aligned} & (\neg b_0 \triangleright b_0) \wedge \\ & ((\neg b_1 \wedge b_0) \triangleright (b_1 \wedge \neg b_0)) \wedge \\ & ((\neg b_2 \wedge b_1 \wedge b_0) \triangleright (b_2 \wedge \neg b_1 \wedge \neg b_0)) \wedge \\ & ((\neg b_3 \wedge b_2 \wedge b_1 \wedge b_0) \triangleright (b_3 \wedge \neg b_2 \wedge \neg b_1 \wedge \neg b_0)) \end{aligned}$$

A4.3 Propositional Planning Tasks

Propositional Planning Tasks

Definition (Propositional Planning Task)

A **propositional planning task** is a 4-tuple $\Pi = \langle V, I, O, \gamma \rangle$ where

- ▶ V is a finite set of propositions called **state variables**,
- ▶ I is a valuation over V called the **initial state**,
- ▶ O is a finite set of **operators** over V , and
- ▶ γ is a formula over V called the **goal**.

Notes:

- ▶ Next week, we will introduce other kinds of planning tasks (with non-propositional state variables).
- ▶ We say **planning task** (without “propositional”) when we do not care about the specific kind of task.

Mapping Prop. Planning Tasks to Transition Systems

Definition (Transition System Induced by a Prop. Planning Task)

The propositional planning task $\Pi = \langle V, I, O, \gamma \rangle$ induces the transition system $\mathcal{T}(\Pi) = \langle S, L, c, T, s_0, S_* \rangle$, where

- ▶ S is the set of all valuations of V ,
- ▶ L is the set of operators O ,
- ▶ $c(o) = \text{cost}(o)$ for all operators $o \in O$,
- ▶ $T = \{ \langle s, o, s' \rangle \mid s \in S, o \text{ applicable in } s, s' = s[o] \}$,
- ▶ $s_0 = I$, and
- ▶ $S_* = \{ s \in S \mid s \models \gamma \}$.

Planning Tasks: Terminology

- ▶ Terminology for transitions systems is also applied to the planning tasks Π that induce them.
- ▶ For example, when we speak of the **states of Π** , we mean the states of $\mathcal{T}(\Pi)$.
- ▶ A sequence of operators that forms a solution of $\mathcal{T}(\Pi)$ is called a **plan** of Π .

Satisficing and Optimal Planning

By **planning**, we mean the following two algorithmic problems:

Definition (Satisficing Planning)

Given: a planning task Π

Output: a plan for Π , or **unsolvable** if no plan for Π exists

Definition (Optimal Planning)

Given: a planning task Π

Output: a plan for Π with minimal cost among all plans for Π , or **unsolvable** if no plan for Π exists

Summary

- ▶ **Propositional planning tasks** compactly represent transition systems and are suitable as inputs for planning algorithms.
- ▶ They are based on concepts from **propositional logic**, enhanced to model state change.
- ▶ **States** of propositional planning tasks are propositional valuations.
- ▶ **Operators** of propositional planning tasks describe **in which situations** (precondition), **how** (effect) and at which **cost** the state of the world can be changed.
- ▶ In **satisficing planning**, we must find a solution for a planning task (or show that no solution exists).
- ▶ In **optimal planning**, we must additionally guarantee that generated solutions are of minimal cost.