

Planning and Optimization

A6. Positive Normal Form and STRIPS

Gabriele Röger and Thomas Keller

Universität Basel

October 3, 2018

Planning and Optimization

October 3, 2018 — A6. Positive Normal Form and STRIPS

A6.1 Motivation

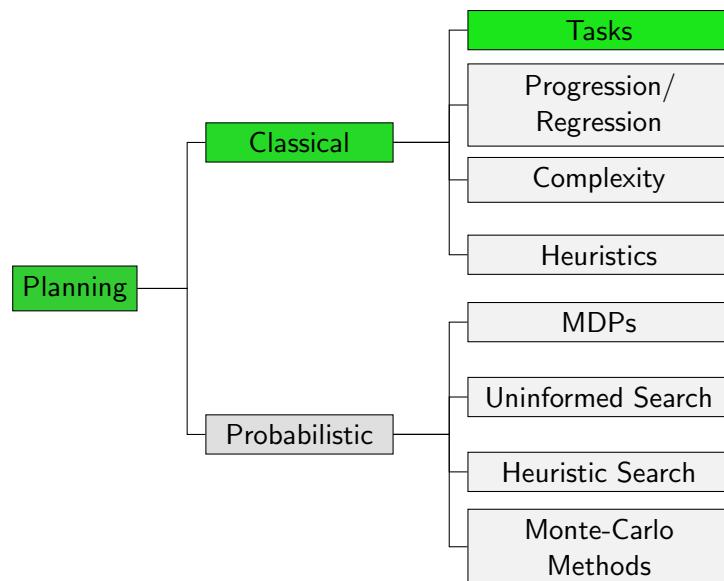
A6.2 Positive Normal Form

A6.3 Example and Discussion

A6.4 STRIPS

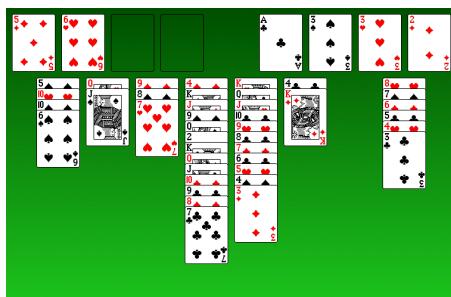
A6.5 Summary

Content of this Course



A6.1 Motivation

Example: Freecell



Example (Good and Bad Effects)

If we move $K\Diamond$ to a free tableau position,
the **good effect** is that $4\clubsuit$ is now accessible.
The **bad effect** is that we lose one free tableau position.

What is a Good or Bad Effect?

Question: Which operator effects are good, and which are bad?

Difficult to answer in general, because it depends on context:

- ▶ Locking our door is **good** if we want to keep burglars out.
- ▶ Locking our door is **bad** if we want to enter.

We now consider a reformulation of propositional planning tasks that makes the distinction between good and bad effects obvious.

A6.2 Positive Normal Form

Positive Formulas, Operators and Tasks

Definition (Positive Formula)

A logical formula φ is **positive** if no negation symbols appear in φ .

Note: This also covers the negation symbols implied by \rightarrow and \leftrightarrow .

Definition (Positive Operator)

An operator o is **positive** if $pre(o)$ and all effect conditions in $eff(o)$ are positive.

Definition (Positive Propositional Planning Task)

A propositional planning task $\langle V, I, O, \gamma \rangle$ is **positive** if all operators in O and the goal γ are positive.

Positive Normal Form

Definition (Positive Normal Form)

A propositional planning task is in **positive normal form** if it is positive and all operator effects are flat.

Positive Normal Form: Existence

Theorem (Positive Normal Form)

For every propositional planning task Π , there is an equivalent propositional planning task Π' in positive normal form.

Moreover, Π' can be computed from Π in polynomial time.

Note: Equivalence here means that the transition systems induced by Π and Π' , restricted to the reachable states, are isomorphic.

We prove the theorem by describing a suitable algorithm.
(However, we do not prove its correctness or complexity.)

Positive Normal Form: Algorithm

Transformation of $\langle V, I, O, \gamma \rangle$ to Positive Normal Form

Replace all operator effects by equivalent conflict-free effects.

Convert all conditions to negation normal form (NNF).

while any condition contains a negative literal $\neg v$:

Let v be a variable which occurs negatively in a condition.

$V := V \cup \{\hat{v}\}$ for some new state variable \hat{v}

$$I(\hat{v}) := \begin{cases} \mathbf{F} & \text{if } I(v) = \mathbf{T} \\ \mathbf{T} & \text{if } I(v) = \mathbf{F} \end{cases}$$

Replace the effect v by $(v \wedge \neg \hat{v})$ in all operators $o \in O$.

Replace the effect $\neg v$ by $(\neg v \wedge \hat{v})$ in all operators $o \in O$.

Replace $\neg v$ by \hat{v} in all conditions.

Convert all operators $o \in O$ to flat operators.

Here, **all conditions** refers to all operator preconditions, operator effect conditions and the goal.

A6.3 Example and Discussion

Positive Normal Form: Example

Example (Transformation to Positive Normal Form)

$$\begin{aligned}
 V &= \{ \text{home}, \text{uni}, \text{lecture}, \text{bike}, \text{bike-locked} \} \\
 I &= \{ \text{home} \mapsto \mathbf{T}, \text{bike} \mapsto \mathbf{T}, \text{bike-locked} \mapsto \mathbf{T}, \\
 &\quad \text{uni} \mapsto \mathbf{F}, \text{lecture} \mapsto \mathbf{F} \} \\
 O &= \{ \langle \text{home} \wedge \text{bike} \wedge \neg \text{bike-locked}, \neg \text{home} \wedge \text{uni} \rangle, \\
 &\quad \langle \text{bike} \wedge \text{bike-locked}, \neg \text{bike-locked} \rangle, \\
 &\quad \langle \text{bike} \wedge \neg \text{bike-locked}, \text{bike-locked} \rangle, \\
 &\quad \langle \text{uni}, \text{lecture} \wedge ((\text{bike} \wedge \neg \text{bike-locked}) \triangleright \neg \text{bike}) \rangle \} \\
 \gamma &= \text{lecture} \wedge \text{bike}
 \end{aligned}$$

Positive Normal Form: Example

Example (Transformation to Positive Normal Form)

$$\begin{aligned}
 V &= \{ \text{home}, \text{uni}, \text{lecture}, \text{bike}, \text{bike-locked} \} \\
 I &= \{ \text{home} \mapsto \mathbf{T}, \text{bike} \mapsto \mathbf{T}, \text{bike-locked} \mapsto \mathbf{T}, \\
 &\quad \text{uni} \mapsto \mathbf{F}, \text{lecture} \mapsto \mathbf{F} \} \\
 O &= \{ \langle \text{home} \wedge \text{bike} \wedge \neg \text{bike-locked}, \neg \text{home} \wedge \text{uni} \rangle, \\
 &\quad \langle \text{bike} \wedge \text{bike-locked}, \neg \text{bike-locked} \rangle, \\
 &\quad \langle \text{bike} \wedge \neg \text{bike-locked}, \text{bike-locked} \rangle, \\
 &\quad \langle \text{uni}, \text{lecture} \wedge ((\text{bike} \wedge \neg \text{bike-locked}) \triangleright \neg \text{bike}) \rangle \} \\
 \gamma &= \text{lecture} \wedge \text{bike}
 \end{aligned}$$

Identify state variable v occurring negatively in conditions.

Positive Normal Form: Example

Example (Transformation to Positive Normal Form)

$$\begin{aligned}
 V &= \{ \text{home}, \text{uni}, \text{lecture}, \text{bike}, \text{bike-locked}, \text{bike-unlocked} \} \\
 I &= \{ \text{home} \mapsto \mathbf{T}, \text{bike} \mapsto \mathbf{T}, \text{bike-locked} \mapsto \mathbf{T}, \\
 &\quad \text{uni} \mapsto \mathbf{F}, \text{lecture} \mapsto \mathbf{F}, \text{bike-unlocked} \mapsto \mathbf{F} \} \\
 O &= \{ \langle \text{home} \wedge \text{bike} \wedge \neg \text{bike-locked}, \neg \text{home} \wedge \text{uni} \rangle, \\
 &\quad \langle \text{bike} \wedge \text{bike-locked}, \neg \text{bike-locked} \rangle, \\
 &\quad \langle \text{bike} \wedge \neg \text{bike-locked}, \text{bike-locked} \rangle, \\
 &\quad \langle \text{uni}, \text{lecture} \wedge ((\text{bike} \wedge \neg \text{bike-locked}) \triangleright \neg \text{bike}) \rangle \} \\
 \gamma &= \text{lecture} \wedge \text{bike}
 \end{aligned}$$

Introduce new variable \hat{v} with complementary initial value.

Positive Normal Form: Example

Example (Transformation to Positive Normal Form)

$$\begin{aligned}
 V &= \{ \text{home}, \text{uni}, \text{lecture}, \text{bike}, \text{bike-locked}, \text{bike-unlocked} \} \\
 I &= \{ \text{home} \mapsto \mathbf{T}, \text{bike} \mapsto \mathbf{T}, \text{bike-locked} \mapsto \mathbf{T}, \\
 &\quad \text{uni} \mapsto \mathbf{F}, \text{lecture} \mapsto \mathbf{F}, \text{bike-unlocked} \mapsto \mathbf{F} \} \\
 O &= \{ \langle \text{home} \wedge \text{bike} \wedge \neg \text{bike-locked}, \neg \text{home} \wedge \text{uni} \rangle, \\
 &\quad \langle \text{bike} \wedge \text{bike-locked}, \neg \text{bike-locked} \rangle, \\
 &\quad \langle \text{bike} \wedge \neg \text{bike-locked}, \text{bike-locked} \rangle, \\
 &\quad \langle \text{uni}, \text{lecture} \wedge ((\text{bike} \wedge \neg \text{bike-locked}) \triangleright \neg \text{bike}) \rangle \} \\
 \gamma &= \text{lecture} \wedge \text{bike}
 \end{aligned}$$

Identify effects on variable v .

Why Positive Normal Form is Interesting

In positive normal form, good and bad effects are easy to distinguish:

- ▶ Effects that make state variables true (**add effects**) are good.
- ▶ Effects that make state variables false (**delete effects**) are bad.

This is particularly useful for planning algorithms based on **delete relaxation**, which we will study later in this course.

A6.4 STRIPS

STRIPS Operators and Planning Tasks

Definition (STRIPS Operator)

An operator o is a **STRIPS operator** if

- ▶ $pre(o)$ is a conjunction of state variables, and
- ▶ $eff(o)$ is a conflict-free conjunction of atomic effects.

Definition (STRIPS Planning Task)

A propositional planning task $\langle V, O, I, \gamma \rangle$ is a **STRIPS planning task** if all operators $o \in O$ are STRIPS operators and γ is a conjunction of state variables.

STRIPS Operators: Remarks

- ▶ Every STRIPS operator is of the form

$$\langle v_1 \wedge \dots \wedge v_n, \ell_1 \wedge \dots \wedge \ell_m \rangle$$

where v_i are state variables and ℓ_j are atomic effects.

- ▶ Often, STRIPS operators o are described via three **sets** of state variables:
 - ▶ the **preconditions** (state variables occurring in $pre(o)$)
 - ▶ the **add effects** (state variables occurring positively in $eff(o)$)
 - ▶ the **delete effects** (state variables occurring negatively in $eff(o)$)
- ▶ There exists a variant called **STRIPS with negation** where negative literals are also allowed in conditions.

Why STRIPS is Interesting

- ▶ STRIPS operators are **particularly simple**, yet expressive enough to capture general planning tasks.
- ▶ In particular, STRIPS planning is **no easier** than planning in general.
- ▶ Most algorithms in the planning literature are **only presented for STRIPS operators** (generalization is often, but not always, obvious).

STRIPS

STanford Research Institute Problem Solver
(Fikes & Nilsson, 1971)

Transformation to STRIPS

- ▶ Not every operator is equivalent to a STRIPS operator.
- ▶ However, each operator can be transformed into a **set** of STRIPS operators whose “combination” is equivalent to the original operator. (How?)
- ▶ However, this transformation may exponentially increase the number of operators. There are planning tasks for which such a blow-up is unavoidable.
- ▶ There are polynomial transformations of propositional planning tasks to STRIPS, but these do not lead to isomorphic transition systems (auxiliary states are needed). (They are, however, equivalent in a weaker sense.)

A6.5 Summary

Summary

- ▶ A **positive** task allows distinguishing good and bad effects.
- ▶ A positive task with flat operators is in **positive normal form**.
- ▶ **STRIPS** is even more restrictive than positive normal form, forbidding complex preconditions and conditional effects.
- ▶ Both forms are expressive enough to capture general propositional planning tasks.
- ▶ Transformation to positive normal form is possible with polynomial size increase.
- ▶ Isomorphic transformations of propositional planning tasks to STRIPS can increase the number of operators exponentially; non-isomorphic polynomial transformations exist.