Motivation

Similarly to normal forms in propositional logic (DNF, CNF, NNF, ...) we can define normal forms for effects, operators and planning tasks.

This is useful because algorithms (and proofs) then only need to deal with effects (resp. operators or tasks) in normal form.

Equivalence of operators and effects

Definition (equivalent effects)

Two effects $e$ and $e'$ over state variables $A$ are equivalent, written $e \equiv e'$, if for all states $s$ over $A$, $[e]_s = [e']_s$.

Definition (equivalent operators)

Two operators $o$ and $o'$ over state variables $A$ are equivalent, written $o \equiv o'$, if they are applicable in the same states, and for all states $s$ where they are applicable, $app_o(s) = app_{o'}(s)$.

Theorem

Let $o = \langle \chi, e \rangle$ and $o' = \langle \chi', e' \rangle$ be operators with $\chi \equiv \chi'$ and $e \equiv e'$. Then $o \equiv o'$.

Note: The converse is not true. (Why not?)
Equivalence transformations for effects

\[\begin{align*}
e_1 \land e_2 & \equiv e_2 \land e_1 \quad (1) \\
(e_1 \land e_2) \land e_3 & \equiv e_1 \land (e_2 \land e_3) \quad (2) \\
T \land e & \equiv e \\
\chi \triangleright e & \equiv \chi' \triangleright e \quad \text{if } \chi \equiv \chi' \\
\top \land e & \equiv e \\
\bot \triangleright e & \equiv \top \\
\chi_1 \triangleright (\chi_2 \triangleright e) & \equiv (\chi_1 \land \chi_2) \triangleright e \\
\chi \triangleright (\langle e_1 \land \cdots \land e_n \rangle) & \equiv (\chi \triangleright e_1) \land \cdots \land (\chi \triangleright e_n) \\
(\chi_1 \triangleright e) \land (\chi_2 \triangleright e) & \equiv (\chi_1 \lor \chi_2) \triangleright e \\
\end{align*}\]

Normal form for effects

We can define a normal form for effects:

- Nesting of conditionals, as in \( a \triangleright (b \triangleright c) \), can be eliminated.
- Effects \( e \) within a conditional effect \( \phi \triangleright e \) can be restricted to atomic effects (\( a \) or \( \neg a \)).

Transformation to this effect normal form only gives a small polynomial size increase.

**Compare**: transformation to CNF or DNF may increase formula size exponentially.

Normal form for operators and effects

**Definition**
An operator \( \langle \chi, e \rangle \) is in effect normal form (ENF) if for all occurrences of \( \chi' \triangleright e' \) in \( e \) the effect \( e' \) is either \( a \) or \( \neg a \) for some \( a \in A \), and there is at most one occurrence of any atomic effect in \( e \).

**Theorem**
For every operator there is an equivalent one in effect normal form.

Proof is constructive: we can transform any operator into effect normal form using the equivalence transformations for effects.

Effect normal form example

**Example**
\[
(\langle a \triangleright (b \land (c \triangleright (\neg d \land e))) \rangle \land (\neg b \triangleright e))
\]

transformed to effect normal form is
\[
(\langle a \triangleright b \rangle \land (\langle a \land c \rangle \triangleright \neg d) \land ((\neg b \lor (a \land c)) \triangleright e))
\]
Positive normal form

**Example: Freecell**

If we move a card $c$ to a free tableau position, the **good effect** is that the card formerly below $c$ is now available. 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 the entrance door is **good** if we want to keep burglars out.
- Locking the entrance door is **bad** if we want to enter.

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

Positive normal form

**Definition (operators in positive normal form)**

An operator $o = (\chi, e)$ is in **positive normal form** if it is in effect normal form, no negation symbols appear in $\chi$, and no negation symbols appear in any effect condition in $e$.

**Definition (planning tasks in positive normal form)**

A planning task $\langle A, I, O, \gamma \rangle$ is in **positive normal form** if all operators in $O$ are in positive normal form and no negation symbols occur in the goal $\gamma$. 
Positive normal form: existence

**Theorem (positive normal form)**

*Every planning task $\Pi$ has an equivalent planning task $\Pi'$ in positive normal form. Moreover, $\Pi'$ can be computed from $\Pi$ in polynomial time.*

**Note:** Equivalence here means that the represented transition systems of $\Pi$ and $\Pi'$, limited to the states that can be reached from the initial state, 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 $\langle A, I, O, \gamma \rangle$ to positive normal form**

Convert all operators $o \in O$ to effect normal form.

Convert all conditions to negation normal form (NNF). while any condition contains a negative literal $\neg a$:

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

1. $A := A \cup \{\hat{a}\}$ for some new state variable $\hat{a}$
2. $I(\hat{a}) := 1 - I(a)$
3. Replace the effect $a$ by $(a \land \neg \hat{a})$ in all operators $o \in O$.
4. Replace the effect $\neg a$ by $(\neg a \land \hat{a})$ in all operators $o \in O$.
5. Replace $\neg a$ by $\hat{a}$ in all conditions.
6. Convert all operators $o \in O$ to effect normal form (again).

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

---

Positive normal form: example

**Example (transformation to positive normal form)**

$A = \{\text{home}, \text{uni}, \text{lecture}, \text{bike}, \text{bike-locked}\}$

$I = \{\text{home} \rightarrow 1, \text{bike} \rightarrow 1, \text{bike-locked} \rightarrow 1,$

$\text{uni} \rightarrow 0, \text{lecture} \rightarrow 0\}$

$O = \{\text{home} \land \text{bike} \land \neg \text{bike-locked}, \neg \text{home} \land \text{uni}\},$

$\langle \text{bike} \land \neg \text{bike-locked}, \neg \text{bike-locked}\rangle,$

$\langle \text{bike} \land \text{bike-locked}, \text{bike-locked}\rangle,$

$\langle \text{uni}, \text{lecture} \land ((\text{bike} \land \neg \text{bike-locked}) \implies \neg \text{bike})\} \}$

$\gamma = \text{lecture} \land \text{bike}$
Introduce complementary effects for \( \hat{a} \) with complementary initial value.

\[
\begin{align*}
A &= \{ \text{home, uni, lecture, bike, bike-locked, bike-unlocked}\} \\
I &= \{ \text{home} \rightarrow 1, \text{bike} \rightarrow 1, \text{bike-locked} \rightarrow 1, \\
&\quad \text{uni} \rightarrow 0, \text{lecture} \rightarrow 0, \text{bike-unlocked} \rightarrow 0 \} \\
O &= \{ \{ \text{home} \land \text{bike} \land \lnot\text{bike-locked}, \lnot\text{home} \land \text{uni}\}, \\
&\quad \{ \text{bike} \land \lnot\text{bike-locked}, \lnot\text{bike} \land \text{bike-locked}\}, \\
&\quad \{ \text{uni}, \text{lecture} \land \{(\text{bike} \land \lnot\text{bike-locked}) \Rightarrow \lnot\text{bike}\}\} \}
\end{align*}
\]
\( \gamma = \text{lecture} \land \text{bike} \)

Introduce new variable \( \hat{a} \) with complementary initial value.

\[
\begin{align*}
A &= \{ \text{home, uni, lecture, bike, bike-locked, bike-unlocked}\} \\
I &= \{ \text{home} \rightarrow 1, \text{bike} \rightarrow 1, \text{bike-locked} \rightarrow 1, \\
&\quad \text{uni} \rightarrow 0, \text{lecture} \rightarrow 0, \text{bike-unlocked} \rightarrow 0 \} \\
O &= \{ \{ \text{home} \land \text{bike} \land \lnot\text{bike-locked}, \lnot\text{home} \land \text{uni}\}, \\
&\quad \{ \text{bike} \land \lnot\text{bike-locked}, \lnot\text{bike} \land \text{bike-locked}\}, \\
&\quad \{ \text{uni}, \text{lecture} \land \{(\text{bike} \land \lnot\text{bike-locked}) \Rightarrow \lnot\text{bike}\}\} \}
\end{align*}
\]
\( \gamma = \text{lecture} \land \text{bike} \)

Identify effects on variable \( a \).

\[
\begin{align*}
A &= \{ \text{home, uni, lecture, bike, bike-locked, bike-unlocked}\} \\
I &= \{ \text{home} \rightarrow 1, \text{bike} \rightarrow 1, \text{bike-locked} \rightarrow 1, \\
&\quad \text{uni} \rightarrow 0, \text{lecture} \rightarrow 0, \text{bike-unlocked} \rightarrow 0 \} \\
O &= \{ \{ \text{home} \land \text{bike} \land \lnot\text{bike-locked}, \lnot\text{home} \land \text{uni}\}, \\
&\quad \{ \text{bike} \land \lnot\text{bike-locked}, \lnot\text{bike} \land \text{bike-locked}\}, \\
&\quad \{ \text{uni}, \text{lecture} \land \{(\text{bike} \land \lnot\text{bike-locked}) \Rightarrow \lnot\text{bike}\}\} \}
\end{align*}
\]
\( \gamma = \text{lecture} \land \text{bike} \)

Identify negative conditions for \( a \).
Why positive normal form is interesting

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

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

This is of high relevance for some planning techniques that we will see later in this course.

October 20th, 2017
B. Nebel, R. Mattmüller – AI Planning
16 / 21

STRIPS operators

October 20th, 2017
B. Nebel, R. Mattmüller – AI Planning
17 / 21
STRIPS operators

Definition
An operator \( \langle \chi, e \rangle \) is a STRIPS operator if
- \( \chi \) is a conjunction of atoms, and
- \( e \) is a conjunction of atomic effects.

Hence every STRIPS operator is of the form
\[
\langle a_1 \land \cdots \land a_n, l_1 \land \cdots \land l_m \rangle
\]
where \( a_i \) are atoms and \( l_j \) are atomic effects.

Note: Sometimes we allow conjunctions of literals as preconditions. We denote this as STRIPS with negative preconditions.

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 required operators. There are planning tasks for which such a blow-up is unavoidable.
- There are polynomial transformations of planning tasks to STRIPS, but these do not preserve the structure of the transition system (e.g., length of shortest plans may change).

Why STRIPS is interesting

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

STRIPS
STanford Research Institute Planning System
(Fikes & Nilsson, 1971)

Summary

- Effect normal form simplifies structure of operator effects: conditional effects contain only atomic effects; there is at most one occurrence of any atomic effect.
- Positive normal form allows to distinguish good and bad effects.
- The form of STRIPS operators is even more restrictive than effect normal form, forbidding complex preconditions and conditional effects.
- All three forms are expressive enough to capture general planning problems.
- Transformation to effect normal form and positive normal form possible with polynomial size increase.
- Structure preserving transformations of planning tasks to STRIPS can increase the number of operators exponentially.