# Planning in the propositional logic

- Early work on deductive planning viewed plans as proofs that lead to a desired goal (theorem).
- Planning as satisfiability testing was proposed in 1992.
  1. A propositional formula represents all length $n$ action sequences from the initial state to a goal state.
  2. If the formula is satisfiable then a plan of length $n$ exists.
- Satisfiability planning is the best approach to solve difficult planning problems.

  Heuristic search is often more efficient on very big but easy problems.
- Bounded model-checking in Computer Aided Verification was introduced in 1998 as an extension of satisfiability planning after the success of the latter had been noticed outside the AI community.

# Planning in the propositional logic



- Early work on deductive planning viewed plans as proofs that lead to a desired goal (theorem).
- Planning as satisfiability testing was proposed in 1992.
  1. A propositional formula represents all length $n$ action sequences from the initial state to a goal state.
  2. If the formula is satisfiable then a plan of length $n$ exists.
- Satisfiability planning is the best approach to solve difficult planning problems.

  Heuristic search is often more efficient on very big but easy problems.
- Bounded model-checking in Computer Aided Verification was introduced in 1998 as an extension of satisfiability planning after the success of the latter had been noticed outside the AI community.

# Planning in the propositional logic

- Early work on deductive planning viewed plans as proofs that lead to a desired goal (theorem).
- Planning as satisfiability testing was proposed in 1992.
  1. A propositional formula represents all length $n$ action sequences from the initial state to a goal state.
  2. If the formula is satisfiable then a plan of length $n$ exists.
- Satisfiability planning is the best approach to solve difficult planning problems.
  Heuristic search is often more efficient on very big but easy problems.
- Bounded model-checking in Computer Aided Verification was introduced in 1998 as an extension of satisfiability planning after the success of the latter had been noticed outside the AI community.

# Planning in the propositional logic



- Early work on deductive planning viewed plans as proofs that lead to a desired goal (theorem).
- Planning as satisfiability testing was proposed in 1992.
  1. A propositional formula represents all length $n$ action sequences from the initial state to a goal state.
  2. If the formula is satisfiable then a plan of length $n$ exists.
- Satisfiability planning is the best approach to solve difficult planning problems.
  Heuristic search is often more efficient on very big but easy problems.
- Bounded model-checking in Computer Aided Verification was introduced in 1998 as an extension of satisfiability planning after the success of the latter had been noticed outside the AI community.

# Planning in the propositional logic

## Abstractly



1. Represent actions (= binary relations) as propositional formulae.
2. Construct a formula saying “execute one of the actions”.
3. Construct a formula saying “execute a sequence of $n$ actions, starting from the initial state, ending in a goal state.”
4. Test the satisfiability of this formula by a satisfiability algorithm.
5. If the formula is satisfiable, construct a plan from a satisfying valuation.

# Planning in the propositional logic
## Abstractly



1. Represent actions (= binary relations) as propositional formulae.
2. Construct a formula saying "execute one of the actions".
3. Construct a formula saying "execute a sequence of $n$ actions, starting from the initial state, ending in a goal state."
4. Test the satisfiability of this formula by a satisfiability algorithm.
5. If the formula is satisfiable, construct a plan from a satisfying valuation.

# Planning in the propositional logic

## Abstractly



1. Represent actions (= binary relations) as propositional formulae.
2. Construct a formula saying "execute one of the actions".
3. Construct a formula saying "execute a sequence of $n$ actions, starting from the initial state, ending in a goal state."
4. Test the satisfiability of this formula by a satisfiability algorithm.
5. If the formula is satisfiable, construct a plan from a satisfying valuation.

# Planning in the propositional logic
## Abstractly



1. Represent actions (= binary relations) as propositional formulae.
2. Construct a formula saying "execute one of the actions".
3. Construct a formula saying "execute a sequence of $n$ actions, starting from the initial state, ending in a goal state."
4. Test the satisfiability of this formula by a satisfiability algorithm.
5. If the formula is satisfiable, construct a plan from a satisfying valuation.

# Planning in the propositional logic

## Abstractly



1. Represent actions (= binary relations) as propositional formulae.
2. Construct a formula saying "execute one of the actions".
3. Construct a formula saying "execute a sequence of $n$ actions, starting from the initial state, ending in a goal state."
4. Test the satisfiability of this formula by a satisfiability algorithm.
5. If the formula is satisfiable, construct a plan from a satisfying valuation.

# Sets (of states) as formulae



## Formulae on $A$ as sets of states

We view formulae $\phi$ as representing sets of states $s : A \rightarrow \{0, 1\}$.

## Example

Formula $a \vee b$ on the state variables $a, b, c$ represents the set $\{010, 011, 100, 101, 110, 111\}$.

# Relations/actions as formulae



## Formulae on $A \cup A'$ as binary relations

Let $A = \{a_1, \ldots, a_n\}$ represent state variables in the current state, and $A' = \{a'_1, \ldots, a'_n\}$ state variables in the successor state.
Formulae $\phi$ on $A \cup A'$ represent binary relations on states: a valuation of $A \cup A' \rightarrow \{0, 1\}$ represents a pair of states $s : A \rightarrow \{0, 1\}$, $s' : A' \rightarrow \{0, 1\}$.

## Example

Formula $(a \rightarrow a') \wedge ((a' \vee b) \rightarrow b')$ on $a, b, a', b'$ represents the binary relation
$\{(00, 00), (00, 01), (00, 11), (01, 01), (01, 11), (10, 11), (11, 11)\}$.

# Matrices as formulae



## Example (Formulae as relations as matrices)

Binary relation $\{(00, 00), (00, 01), (00, 11), (01, 01), (01, 11), (10, 11), (11, 11)\}$ can be represented as the adjacency matrix:

| | $a'b'$ 00 | $a'b'$ 01 | $a'b'$ 10 | $a'b'$ 11 |
|---|---|---|---|---|
| $ab$ | | | | |
| 00 | 1 | 1 | 0 | 1 |
| 01 | 0 | 1 | 0 | 1 |
| 10 | 0 | 0 | 0 | 1 |
| 11 | 0 | 0 | 0 | 1 |

## Representation of big matrices is possible

For $n$ state variables a formula (over $2n$ variables) represents an adjacency matrix of size $2^n \times 2^n$.
For $n = 20$, matrix size is $2^{20} \times 2^{20} \sim 10^6 \times 10^6$.

# Actions/relations as propositional formulae

## Example



$\phi = (a_1 \leftrightarrow \neg a'_1) \wedge (a_2 \leftrightarrow \neg a'_2)$ as a matrix

| $a_1 a_2$ \ $a'_1 a'_2$ | 0 0 | 0 1 | 1 0 | 1 1 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 1 |
| 01 | 0 | 0 | 1 | 0 |
| 10 | 0 | 1 | 0 | 0 |
| 11 | 1 | 0 | 0 | 0 |

and as a conventional truth-table:

| $a_1$ | $a_2$ | $a'_1$ | $a'_2$ | $\phi$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

# Actions/relations as propositional formulae
## Example



$\phi = (a_1 \leftrightarrow \neg a'_1) \land (a_2 \leftrightarrow \neg a'_2)$ as a matrix

| $a_1 a_2$ | $a'_1 a'_2$ 0 0 | $a'_1 a'_2$ 0 1 | $a'_1 a'_2$ 1 0 | $a'_1 a'_2$ 1 1 |
|---|---|---|---|---|
| 00 | 0 | 0 | 0 | 1 |
| 01 | 0 | 0 | 1 | 0 |
| 10 | 0 | 1 | 0 | 0 |
| 11 | 1 | 0 | 0 | 0 |

and as a conventional truth-table:

| $a_1$ | $a_2$ | $a'_1$ | $a'_2$ | $\phi$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 |

# Actions/relations as propositional formulae
## Example



$(a_1 \leftrightarrow a_2') \wedge (a_2 \leftrightarrow a_3') \wedge (a_3 \leftrightarrow a_1')$ represents the matrix:

| | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| 000 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 001 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 010 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 011 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 100 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 101 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 110 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 111 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

This action rotates the value of the state variables $a_1, a_2, a_3$ one step forward.

# Deterministic vs. nondeterministic actions

## Expressiveness of propositional logic

- For every operator there is a corresponding formula (see next slides!)
- Our current definition of operators does not allow expressing nondeterministic actions.
- In the propositional logic they can be expressed.

## Example (A nondeterministic action)

The formula $\top$ describes the relation in which any state can be reached from any other state by this action.

## A sufficient (but not necessary) condition for determinism

Formula has the form $(\phi_1 \leftrightarrow a'_1) \wedge \cdots \wedge (\phi_n \leftrightarrow a'_n)$ where $A = \{a_1, \ldots, a_n\}$ and $\phi_i$ have no occurrences of propositions in $A'$.

# Deterministic vs. nondeterministic actions



## Expressiveness of propositional logic

- For every operator there is a corresponding formula (see next slides!)
- Our current definition of operators does not allow expressing nondeterministic actions.
- In the propositional logic they can be expressed.

## Example (A nondeterministic action)

The formula $\top$ describes the relation in which any state can be reached from any other state by this action.

## A sufficient (but not necessary) condition for determinism

Formula has the form $(\phi_1 \leftrightarrow a_1') \wedge \cdots \wedge (\phi_n \leftrightarrow a_n')$ where $A = \{a_1, \ldots, a_n\}$ and $\phi_i$ have no occurrences of propositions in $A'$.

# Deterministic vs. nondeterministic actions



## Expressiveness of propositional logic

- For every operator there is a corresponding formula (see next slides!)
- Our current definition of operators does not allow expressing nondeterministic actions.
- In the propositional logic they can be expressed.

## Example (A nondeterministic action)

The formula $\top$ describes the relation in which any state can be reached from any other state by this action.

## A sufficient (but not necessary) condition for determinism

Formula has the form $(\phi_1 \leftrightarrow a'_1) \wedge \cdots \wedge (\phi_n \leftrightarrow a'_n)$ where $A = \{a_1, \ldots, a_n\}$ and $\phi_i$ have no occurrences of propositions in $A'$.

# Deterministic vs. nondeterministic actions
## Example



### Example

An action that is applicable if $a$ is false, and that randomly sets values to state variables $b$ and $c$:

| $abc$ \ $a'b'c'$ | 000 | 001 | 010 | 011 | 100 | 101 | 110 | 111 |
|---|---|---|---|---|---|---|---|---|
| 000 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 001 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 010 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 011 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 101 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 111 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Corresponding formula: $\neg a \wedge \neg a'$

# Translating operators into formulae



- Any operator can be translated into a propositional formula.
- Translation takes polynomial time.
- Resulting formula has polynomial size.
- Use in planning algorithms. Two main applications are
  1. Planning as Satisfiability
  2. Progression & regression for state sets as used in symbolic state-space traversal, as typically implemented with the help of binary decision diagrams.

# Translating operators into formulae



## Definition

Let $o = \langle c, e \rangle$ be an operator and $A$ a set of state variables. Define $\tau_A(o)$ as the conjunction of

$$c \quad (1)$$

$$\bigwedge_{a \in A}(\mathit{EPC}_a(e) \vee (a \wedge \neg \mathit{EPC}_{\neg a}(e))) \leftrightarrow a' \quad (2)$$

$$\bigwedge_{a \in A} \neg(\mathit{EPC}_a(e) \wedge \mathit{EPC}_{\neg a}(e)) \quad (3)$$

(2) says that the new value of $a$, represented by $a'$, is 1 if the old value was 1 and it did not become 0, or it became 1.
(3) says that none of the state variables is assigned both 0 and 1. This together with $c$ determine whether the operator is applicable.

## Example

Let the state variables be $A = \{a, b, c\}$.
Consider operator $\langle a \vee b, (b \rhd a) \wedge (c \rhd \neg a) \wedge (a \rhd b) \rangle$.
The corresponding propositional formula is

$$\begin{aligned}
(a \vee b) &\wedge ((b \vee (a \wedge \neg c)) \leftrightarrow a') \\
&\wedge ((a \vee (b \wedge \neg \bot)) \leftrightarrow b') \\
&\wedge ((\bot \vee (c \wedge \neg \bot)) \leftrightarrow c') \\
&\wedge \neg (b \wedge c) \wedge \neg (a \wedge \bot) \wedge \neg (\bot \wedge \bot) \\
\equiv& \\
(a \vee b) &\wedge ((b \vee (a \wedge \neg c)) \leftrightarrow a') \\
&\wedge ((a \vee b) \leftrightarrow b') \\
&\wedge (c \leftrightarrow c') \\
&\wedge \neg (b \wedge c)
\end{aligned}$$

# Translating operators into formulae
## Example



### Example

Let $A = \{a, b, c, d, e\}$ be the state variables.
Consider operator $\langle a \wedge b, c \wedge (d \rhd e)\rangle$.
The formula $\tau_A(o)$ after simplifications is

$$(a \wedge b) \wedge (a \leftrightarrow a') \wedge (b \leftrightarrow b') \wedge c' \wedge (d \leftrightarrow d') \wedge ((d \vee e) \leftrightarrow e')$$

# Correctness



**Lemma**

*Let $s$ and $s'$ be states and $o$ an operator. Let $v : A \cup A' \rightarrow \{0, 1\}$ be a valuation such that*

1. *for all $a \in A$, $v(a) = s(a)$, and*
2. *for all $a \in A$, $v(a') = s'(a)$.*

*Then $v \models \tau_A(o)$ if and only if $s' = \text{app}_o(s)$.*

# Planning as satisfiability



1. Encode operator sequences of length 0, 1, 2, ... as formulae $\Phi_0^{seq}, \Phi_1^{seq}, \Phi_2^{seq}, \ldots$ (see next slide...)
2. Test satisfiability of $\Phi_0^{seq}, \Phi_1^{seq}, \Phi_2^{seq}, \ldots$.
3. If a satisfying valuation $v$ is found, a plan can constructed from $v$.

# Planning as satisfiability



## Definition (Transition relation in CPC)

For $\langle A, I, O, G\rangle$ define

$$\mathcal{R}_1(A, A') = \bigvee_{o \in O} \tau_A(o).$$

## Definition (Bounded-length plans in CPC)

Existence of plans length $t$ is represented by a formula over propositions $A^0 \cup \dots \cup A^t$ where $A^i = \{a^i | a \in A\}$ for all $i \in \{0, \dots, t\}$ as

$$\Phi_t^{seq} = \iota^0 \wedge \mathcal{R}_1(A^0, A^1) \wedge \mathcal{R}_1(A^1, A^2) \wedge \dots \wedge \mathcal{R}_1(A^{t-1}, A^t) \wedge G^t$$

where $\iota^0 = \bigwedge \{a^0 | a \in A, I(a) = 1\} \cup \{\neg a^0 | a \in A, I(a) = 0\}$ and $G^t$ is $G$ with propositions $a$ replaced by $a^t$.

## Example

Consider

$$
\begin{aligned}
I &\models b \wedge c \\
G &= (b \wedge \neg c) \vee (\neg b \wedge c) \\
o_1 &= \langle \top, (c \rhd \neg c) \wedge (\neg c \rhd c) \rangle \\
o_2 &= \langle \top, (b \rhd \neg b) \wedge (\neg b \rhd b) \rangle.
\end{aligned}
$$

Formula for plans of length 3 is

$$
\begin{aligned}
&(b^0 \wedge c^0) \\
&\wedge(((b^0 \leftrightarrow b^1) \wedge (c^0 \leftrightarrow \neg c^1)) \vee ((b^0 \leftrightarrow \neg b^1) \wedge (c^0 \leftrightarrow c^1))) \\
&\wedge(((b^1 \leftrightarrow b^2) \wedge (c^1 \leftrightarrow \neg c^2)) \vee ((b^1 \leftrightarrow \neg b^2) \wedge (c^1 \leftrightarrow c^2))) \\
&\wedge(((b^2 \leftrightarrow b^3) \wedge (c^2 \leftrightarrow \neg c^3)) \vee ((b^2 \leftrightarrow \neg b^3) \wedge (c^2 \leftrightarrow c^3))) \\
&\wedge((b^3 \wedge \neg c^3) \vee (\neg b^3 \wedge c^3)).
\end{aligned}
$$

# Planning as satisfiability

## Existence of (optimal) plans

### Theorem

*Let $\Phi_t^{seq}$ be the formula for $\langle A, I, O, G \rangle$ and plan length $t$. The formula $\Phi_t^{seq}$ is satisfiable if and only if there is a sequence of states $s_0, \ldots, s_t$ and operators $o_1, \ldots, o_t$ such that $s_0 = I$, $s_t \models G$ and $s_i = app_{o_i}(s_{i-1})$ for all $i \in \{1, \ldots, t\}$.*

### Consequence

If $\Phi_0^{seq}, \Phi_1^{seq}, \ldots, \Phi_{i-1}^{seq}$ are unsatisfiable and $\Phi_i^{seq}$ is satisfiable, then the length of shortest plans is $i$.
Satisfiability planning with $\Phi_i^{seq}$ yields optimal plans, like heuristic search with admissible heuristics and optimal algorithms like A$*$ or IDA$*$.

All satisfiability algorithms give a valuation $v$ that satisfies $\Phi_i^{seq}$ upon finding out that $\Phi_i^{seq}$ is satisfiable.
This makes it possible to construct a plan.

## Constructing a plan from a satisfying valuation

Let $v$ be a valuation so that $v \models \Phi_t^{seq}$. Then define $s_i(a) = v(a^i)$ for all $a \in A$ and $i \in \{0, \ldots, t\}$.
The $i$th operator in the plan is $o \in O$ if $app_o(s_{i-1}) = s_i$.
Notice: There may be more than one such operator, any of them may be chosen.

# Planning as satisfiability

## Example, continued



### Example

One valuation that satisfies $\Phi_3^{seq}$:

| | time $i$ | | | |
|---|---|---|---|---|
| | 0 | 1 | 2 | 3 |
| $b^i$ | 1 | 1 | 0 | 0 |
| $c^i$ | 1 | 0 | 0 | 1 |

Notice:

1. Also a plan of length 1 exists.
2. Plans of length 2 do not exist.

# Conjunctive normal form





Many satisfiability algorithms require formulas in the conjunctive normal form: transformation by repeated applications of the following equivalences.

$$
\begin{aligned}
\neg(\phi \vee \psi) &\equiv \neg\phi \wedge \neg\psi \\
\neg(\phi \wedge \psi) &\equiv \neg\phi \vee \neg\psi \\
\neg\neg\phi &\equiv \phi \\
\phi \vee (\psi_1 \wedge \psi_2) &\equiv (\phi \vee \psi_1) \wedge (\phi \vee \psi_2)
\end{aligned}
$$

The formula is conjunction of clauses (disjunctions of literals).

**Example**

$(A \vee \neg B \vee C) \wedge (\neg C \vee \neg B) \wedge A$

# The unit resolution rule



## Unit resolution

From $l_1 \vee l_2 \vee \cdots \vee l_n$ (here $n \geq 1$) and $\overline{l_1}$ infer $l_2 \vee \cdots \vee l_n$.

## Example

From $a \vee b \vee c$ and $\neg a$ infer $b \vee c$.

## Unit resolution: a special case

From $A$ and $\neg A$ we get the empty clause $\bot$ ("disjunction consisting of zero disjuncts").

## Unit subsumption

The clause $l_1 \vee l_2 \vee \cdots \vee l_n$ can be eliminated if we have the unit clause $l_1$.

# The Davis-Putnam procedure



- The first efficient decision procedure for any logic (Davis, Putnam, Logemann & Loveland, 1960/62).
- Based on binary search through the valuations of a formula.
- Unit resolution and unit subsumption help pruning the search tree.
- The currently most efficient satisfiability algorithms are variants of the Davis-Putnam procedure
(Although there is currently a shift toward viewing these procedures as performing more general resolution: clause-learning.)

# Satisfiability test by the Davis-Putnam procedure



1. Let $C$ be a set of clauses.
2. For all clauses $l_1 \vee l_2 \vee \cdots \vee l_n \in C$ and $\overline{l_1} \in C$, remove $l_1 \vee l_2 \vee \cdots \vee l_n$ from $C$ and add $l_2 \vee \cdots \vee l_n$ to $C$.
3. For all clauses $l_1 \vee l_2 \vee \cdots \vee l_n \in C$ and $l_1 \in C$, remove $l_1 \vee l_2 \vee \cdots \vee l_n$ from $C$. (UNIT SUBSUMPTION)
4. If $\bot \in C$, return FALSE.
5. If $C$ contains only unit clauses, return TRUE.
6. Pick some $a \in A$ such that $\{a, \neg a\} \cap C = \emptyset$
7. Recursive call: if $C \cup \{a\}$ is satisfiable, return TRUE.
8. Recursive call: if $C \cup \{\neg a\}$ is satisfiable, return TRUE.
9. Return FALSE.

Consider the problem from a previous slide, with two operators each inverting the value of one state variable, for plan length 3.

$$
\begin{aligned}
&(b^0 \wedge c^0)\\
&\wedge(((b^0 \leftrightarrow b^1) \wedge (c^0 \leftrightarrow \neg c^1)) \vee ((b^0 \leftrightarrow \neg b^1) \wedge (c^0 \leftrightarrow c^1)))\\
&\wedge(((b^1 \leftrightarrow b^2) \wedge (c^1 \leftrightarrow \neg c^2)) \vee ((b^1 \leftrightarrow \neg b^2) \wedge (c^1 \leftrightarrow c^2)))\\
&\wedge(((b^2 \leftrightarrow b^3) \wedge (c^2 \leftrightarrow \neg c^3)) \vee ((b^2 \leftrightarrow \neg b^3) \wedge (c^2 \leftrightarrow c^3)))\\
&\wedge((b^3 \wedge \neg c^3) \vee (\neg b^3 \wedge c^3)).
\end{aligned}
$$

# Planning as satisfiability

## Example: plan search with Davis-Putnam



To obtain a short CNF formula, we introduce auxiliary variables $o_1^i$ and $o_2^i$ for $i \in \{1, 2, 3\}$ denoting operator applications.

$$b^0$$
$$c^0$$
$$o_1^1 \vee o_2^1$$
$$o_1^2 \vee o_2^2$$
$$o_1^3 \vee o_2^3$$
$$(b^3 \wedge \neg c^3) \vee (\neg b^3 \wedge c^3)$$

$$o_1^1 \rightarrow ((b^0 \leftrightarrow b^1) \wedge (c^0 \leftrightarrow \neg c^1))$$
$$o_2^1 \rightarrow ((b^0 \leftrightarrow \neg b^1) \wedge (c^0 \leftrightarrow c^1))$$
$$o_1^2 \rightarrow ((b^1 \leftrightarrow b^2) \wedge (c^1 \leftrightarrow \neg c^2))$$
$$o_2^2 \rightarrow ((b^1 \leftrightarrow \neg b^2) \wedge (c^1 \leftrightarrow c^2))$$
$$o_1^3 \rightarrow ((b^2 \leftrightarrow b^3) \wedge (c^2 \leftrightarrow \neg c^3))$$
$$o_2^3 \rightarrow ((b^2 \leftrightarrow \neg b^3) \wedge (c^2 \leftrightarrow c^3))$$

# Planning as satisfiability
## Example: plan search with Davis-Putnam



We rewrite the formulae for operator applications by using the equivalence $\phi \rightarrow (l \leftrightarrow l') \equiv ((\phi \wedge l \rightarrow l') \wedge (\phi \wedge \bar{l} \rightarrow \overline{l'}))$.

$$
\begin{array}{llll}
b^0 & o_1^1 \wedge b^0 \rightarrow b^1 & o_1^2 \wedge b^1 \rightarrow b^2 & o_1^3 \wedge b^2 \rightarrow b^3 \\
c^0 & o_1^1 \wedge \neg b^0 \rightarrow \neg b^1 & o_1^2 \wedge \neg b^1 \rightarrow \neg b^2 & o_1^3 \wedge \neg b^2 \rightarrow \neg b^3 \\
o_1^1 \vee o_2^1 & o_1^1 \wedge c^0 \rightarrow \neg c^1 & o_1^2 \wedge c^1 \rightarrow \neg c^2 & o_1^3 \wedge c^2 \rightarrow \neg c^3 \\
o_1^2 \vee o_2^2 & o_1^1 \wedge \neg c^0 \rightarrow c^1 & o_1^2 \wedge \neg c^1 \rightarrow c^2 & o_1^3 \wedge \neg c^2 \rightarrow c^3 \\
o_1^3 \vee o_2^3 & o_2^1 \wedge b^0 \rightarrow \neg b^1 & o_2^2 \wedge b^1 \rightarrow \neg b^2 & o_2^3 \wedge b^2 \rightarrow \neg b^3 \\
b^3 \vee c^3 & o_2^1 \wedge \neg b^0 \rightarrow b^1 & o_2^2 \wedge \neg b^1 \rightarrow b^2 & o_2^3 \wedge \neg b^2 \rightarrow b^3 \\
\neg c^3 \vee \neg b^3 & o_2^1 \wedge c^0 \rightarrow c^1 & o_2^2 \wedge c^1 \rightarrow c^2 & o_2^3 \wedge c^2 \rightarrow c^3 \\
 & o_2^1 \wedge \neg c^0 \rightarrow c^1 & o_2^2 \wedge \neg c^1 \rightarrow c^2 & o_2^3 \wedge \neg c^2 \rightarrow c^3
\end{array}
$$

# Planning as satisfiability
## Example: plan search with Davis-Putnam



Eliminate implications with $((l_1 \wedge l_2) \rightarrow l_3) \equiv (\overline{l_1} \vee \overline{l_2} \vee l_3)$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | | | | |
| $c^i$ | | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

Identify unit clauses.

$b^0$
$c^0$
$o_1^1 \vee o_2^1$
$o_1^2 \vee o_2^2$
$o_1^3 \vee o_2^3$
$b^3 \vee c^3$
$\neg c^3 \vee \neg b^3$

| | | |
|---|---|---|
| $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

## Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

# Planning as satisfiability
## Example: plan search with Davis-Putnam



Perform unit resolution with $b^0$ and $c^0$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

# Planning as satisfiability
## Example: plan search with Davis-Putnam



Perform unit subsumption with $b^0$ and $c^0$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

# Planning as satisfiability

## Example: plan search with Davis-Putnam



Identify unit clauses. None exist. Must branch.

$b^0$
$c^0$
$o_1^1 \vee o_2^1$
$o_1^2 \vee o_2^2$
$o_1^3 \vee o_2^3$
$b^3 \vee c^3$
$\neg c^3 \vee \neg b^3$

$\neg o_1^1 \vee \neg b^0 \vee b^1$
$\neg o_1^1 \vee b^0 \vee \neg b^1$
$\neg o_1^1 \vee \neg c^0 \vee \neg c^1$
$\neg o_1^1 \vee c^0 \vee c^1$
$\neg o_2^1 \vee \neg b^0 \vee \neg b^1$
$\neg o_2^1 \vee b^0 \vee b^1$
$\neg o_2^1 \vee \neg c^0 \vee c^1$
$\neg o_2^1 \vee c^0 \vee c^1$

$\neg o_1^2 \vee \neg b^1 \vee b^2$
$\neg o_1^2 \vee b^1 \vee \neg b^2$
$\neg o_1^2 \vee \neg c^1 \vee \neg c^2$
$\neg o_1^2 \vee c^1 \vee c^2$
$\neg o_2^2 \vee \neg b^1 \vee \neg b^2$
$\neg o_2^2 \vee b^1 \vee b^2$
$\neg o_2^2 \vee \neg c^1 \vee c^2$
$\neg o_2^2 \vee c^1 \vee c^2$

$\neg o_1^3 \vee \neg b^2 \vee b^3$
$\neg o_1^3 \vee b^2 \vee \neg b^3$
$\neg o_1^3 \vee \neg c^2 \vee \neg c^3$
$\neg o_1^3 \vee c^2 \vee c^3$
$\neg o_2^3 \vee \neg b^2 \vee \neg b^3$
$\neg o_2^3 \vee b^2 \vee b^3$
$\neg o_2^3 \vee \neg c^2 \vee c^3$
$\neg o_2^3 \vee c^2 \vee c^3$

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

We branch on $b^1$, first trying out $b^1 = 1$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

Perform unit resolution and unit subsumption with $b^1$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

## Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | | | |

Perform unit resolution and unit subsumption with $\neg o_2^1$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

## Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | | | |
| $o_2^i$ | 0 | | |

# Planning as satisfiability
## Example: plan search with Davis-Putnam

We obtain unit clause $o_1^1$ and directly after it $\neg c^1$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | 0 | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | 1 | | |
| $o_2^i$ | 0 | | |

Perform unit resolution and unit subsumption with $o_1^1, \neg c^1$.

| | | | |
|---|---|---|---|
| $b^0$ | $\neg o_1^1 \vee \neg b^0 \vee b^1$ | $\neg o_1^2 \vee \neg b^1 \vee b^2$ | $\neg o_1^3 \vee \neg b^2 \vee b^3$ |
| $c^0$ | $\neg o_1^1 \vee b^0 \vee \neg b^1$ | $\neg o_1^2 \vee b^1 \vee \neg b^2$ | $\neg o_1^3 \vee b^2 \vee \neg b^3$ |
| $o_1^1 \vee o_2^1$ | $\neg o_1^1 \vee \neg c^0 \vee \neg c^1$ | $\neg o_1^2 \vee \neg c^1 \vee \neg c^2$ | $\neg o_1^3 \vee \neg c^2 \vee \neg c^3$ |
| $o_1^2 \vee o_2^2$ | $\neg o_1^1 \vee c^0 \vee c^1$ | $\neg o_1^2 \vee c^1 \vee c^2$ | $\neg o_1^3 \vee c^2 \vee c^3$ |
| $o_1^3 \vee o_2^3$ | $\neg o_2^1 \vee \neg b^0 \vee \neg b^1$ | $\neg o_2^2 \vee \neg b^1 \vee \neg b^2$ | $\neg o_2^3 \vee \neg b^2 \vee \neg b^3$ |
| $b^3 \vee c^3$ | $\neg o_2^1 \vee b^0 \vee b^1$ | $\neg o_2^2 \vee b^1 \vee b^2$ | $\neg o_2^3 \vee b^2 \vee b^3$ |
| $\neg c^3 \vee \neg b^3$ | $\neg o_2^1 \vee \neg c^0 \vee c^1$ | $\neg o_2^2 \vee \neg c^1 \vee c^2$ | $\neg o_2^3 \vee \neg c^2 \vee c^3$ |
| | $\neg o_2^1 \vee c^0 \vee c^1$ | $\neg o_2^2 \vee c^1 \vee c^2$ | $\neg o_2^3 \vee c^2 \vee c^3$ |

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | 0 | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | 1 | | |
| $o_2^i$ | 0 | | |

# Planning as satisfiability
## Example: plan search with Davis-Putnam



Identify unit clauses. None exist. Must branch a second time.

$b^0$
$c^0$
$o_1^1 \vee o_2^1$
$o_1^2 \vee o_2^2$
$o_1^3 \vee o_2^3$
$b^3 \vee c^3$
$\neg c^3 \vee \neg b^3$

$\neg o_1^1 \vee \neg b^0 \vee b^1$
$\neg o_1^1 \vee b^0 \vee \neg b^1$
$\neg o_1^1 \vee \neg c^0 \vee \neg c^1$
$\neg o_1^1 \vee c^0 \vee c^1$
$\neg o_2^1 \vee \neg b^0 \vee \neg b^1$
$\neg o_2^1 \vee b^0 \vee b^1$
$\neg o_2^1 \vee \neg c^0 \vee c^1$
$\neg o_2^1 \vee c^0 \vee c^1$

$\neg o_1^2 \vee \neg b^1 \vee b^2$
$\neg o_1^2 \vee b^1 \vee \neg b^2$
$\neg o_1^2 \vee \neg c^1 \vee \neg c^2$
$\neg o_1^2 \vee c^1 \vee c^2$
$\neg o_2^2 \vee \neg b^1 \vee \neg b^2$
$\neg o_2^2 \vee b^1 \vee b^2$
$\neg o_2^2 \vee \neg c^1 \vee c^2$
$\neg o_2^2 \vee c^1 \vee c^2$

$\neg o_1^3 \vee \neg b^2 \vee b^3$
$\neg o_1^3 \vee b^2 \vee \neg b^3$
$\neg o_1^3 \vee \neg c^2 \vee \neg c^3$
$\neg o_1^3 \vee c^2 \vee c^3$
$\neg o_2^3 \vee \neg b^2 \vee \neg b^3$
$\neg o_2^3 \vee b^2 \vee b^3$
$\neg o_2^3 \vee \neg c^2 \vee c^3$
$\neg o_2^3 \vee c^2 \vee c^3$

### Valuation constructed by the Davis-Putnam procedure

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| $b^i$ | 1 | 1 | | |
| $c^i$ | 1 | 0 | | |

| | 1 | 2 | 3 |
|---|---|---|---|
| $o_1^i$ | 1 | | |
| $o_2^i$ | 0 | | |

# Planning as satisfiability with parallel plans



- Efficiency of satisfiability planning is strongly dependent on the plan length because satisfiability algorithms have runtime $O(2^n)$ where $n$ is the formula size, and formula sizes are linearly proportional to plan length.
- Formula sizes can be reduced by allowing several operators in parallel.
- On many problems this leads to big speed-ups.
- However there are no guarantees of optimality.

# Parallel operator application
## Formal definition



We consider the possibility of executing several operators simultaneously.

**Definition**

Let $T$ be a set of operators and $s$ a state.
Define $\textit{app}_T(s)$ as the state that is obtained from $s$ by making the literals in $\bigcup_{\langle c,e\rangle \in T}[e]_s$ true.
For $\textit{app}_T(s)$ to be defined, we require that $s \models c$ for all $o = \langle c, e\rangle \in T$ and $\bigcup_{\langle c,e\rangle \in T}[e]_s$ is consistent.

# Parallel operator application
## Representation in CPC



Consider the formula $\tau_A(o)$ representing operator $o = \langle c, e \rangle$

$$
\begin{aligned}
&c \wedge \\
&\textstyle\bigwedge_{a \in A}((\mathit{EPC}_a(e) \vee (a \wedge \neg \mathit{EPC}_{\neg a}(e))) \leftrightarrow a') \wedge \\
&\textstyle\bigwedge_{a \in A} \neg(\mathit{EPC}_a(e) \wedge \mathit{EPC}_{\neg a}(e)).
\end{aligned}
$$

This can be logically equivalently be written as follows.

$$
\begin{aligned}
&c \wedge \\
&\textstyle\bigwedge_{a \in A}(\mathit{EPC}_a(e) \rightarrow a') \wedge \\
&\textstyle\bigwedge_{a \in A}(\mathit{EPC}_{\neg a}(e) \rightarrow \neg a') \wedge \\
&\textstyle\bigwedge_{a \in A}((a \wedge \neg \mathit{EPC}_{\neg a}(e)) \rightarrow a') \wedge \\
&\textstyle\bigwedge_{a \in A}((\neg a \wedge \neg \mathit{EPC}_a(e)) \rightarrow \neg a')
\end{aligned}
$$

This separates the changes from non-changes. This is the basis of the translation for parallel actions for which we do not say that executing a given operator directly means that unrelated state variables retain their old value.

# The explanatory frame axioms



The formulae say that the only explanation for $a$ changing its value is the application of one operator.

$$\bigwedge_{a \in A}((a \wedge \neg a') \rightarrow \mathit{EPC}_{\neg a}(e))$$
$$\bigwedge_{a \in A}((\neg a \wedge a') \rightarrow \mathit{EPC}_{a}(e))$$

When several operators could be applied in parallel, we have to consider all operators as possible explanations.

$$\bigwedge_{a \in A}((a \wedge \neg a') \rightarrow ((o_1 \wedge \mathit{EPC}_{\neg a}(e_1)) \vee \cdots \vee (o_n \wedge \mathit{EPC}_{\neg a}(e_n))$$
$$\bigwedge_{a \in A}((\neg a \wedge a') \rightarrow ((o_1 \wedge \mathit{EPC}_{a}(e_1)) \vee \cdots \vee (o_n \wedge \mathit{EPC}_{a}(e_n)))$$

where $T = \{o_1, \ldots, o_n\}$ and $e_1, \ldots, e_n$ are the respective effects.

# Parallel actions

## Formula in CPC

### Definition

Let $T$ be a set of operators. Let $\tau_A(T)$ denote the conjunction of formulae

$$(o \rightarrow c) \wedge \\ \bigwedge_{a \in A}(o \wedge \mathit{EPC}_a(e) \rightarrow a') \wedge \\ \bigwedge_{a \in A}(o \wedge \mathit{EPC}_{\neg a}(e) \rightarrow \neg a')$$

for all $\langle c, e \rangle \in T$ and

$$\bigwedge_{a \in A}((a \wedge \neg a') \rightarrow ((o_1 \wedge \mathit{EPC}_{\neg a}(e_1)) \vee \cdots \vee (o_n \wedge \mathit{EPC}_{\neg a}(e_n))) \\ \bigwedge_{a \in A}((\neg a \wedge a') \rightarrow ((o_1 \wedge \mathit{EPC}_a(e_1)) \vee \cdots \vee (o_n \wedge \mathit{EPC}_a(e_n)))$$

where $T = \{o_1, \ldots, o_n\}$ and $e_1, \ldots, e_n$ are the respective effects.

# Correctness



The formula $\tau_A(T)$ exactly matches the definition of *app*$_T(s)$.

**Lemma**

*Let $s$ and $s'$ be states and $T$ a set of operators. Let $v : A \cup A' \cup T \rightarrow \{0, 1\}$ be a valuation such that*

1. *for all $o \in T$, $v(o) = 1$,*
2. *for all $a \in A$, $v(a) = s(a)$, and*
3. *for all $a \in A$, $v(a') = s'(a)$.*

*Then $v \models \tau_A(T)$ if and only if $s' =$ app$_T(s)$.*

# Parallel actions
## Meaning in terms of interleavings



**Example**

The operators $\langle a, \neg b\rangle$ and $\langle b, \neg a\rangle$ may be executed simultaneously resulting in a state satisfying $\neg a \wedge \neg b$.
But this state is not reachable by the two operators sequentially, because executing any one operator makes the precondition of the other false.

### Definition (Step plans)

For a set of operators $O$ and an initial state $I$, a step plan for $O$ and $I$ is a sequence $T = \langle T_0, \ldots, T_{l-1} \rangle$ of sets of operators for some $l \geq 0$ such that there is a sequence of states $s_0, \ldots, s_l$ (the execution of $T$) such that

1. $s_0 = I$,
2. for all $i \in \{0, \ldots, l-1\}$ and every total ordering $o_1, \ldots, o_n$ of $T_i$, $app_{o_1;\ldots;o_n}(s_i)$ is defined and equals $s_{i+1}$.

# Step plans
## Tractable subclass

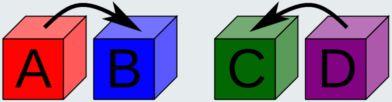


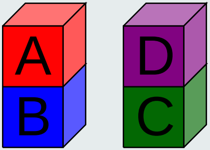
- Finding arbitrary step plans is difficult: even testing whether a set $T$ of operators is executable in all orders is co-NP-hard.
- Representing the executability test exactly as a propositional formula seems complicated: doing this test exactly would seem to cancel the benefits of parallel plans.
- Instead, all work on parallel plans so far has used a sufficient but not necessary condition that can be tested in polynomial-time.
- This is a simple syntactic test: is the result of executing $o_1$ and $o_2$ in any state both in order $o_1; o_2$ and in $o_2; o_1$ the same.

# Interference
## Example



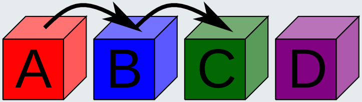
### Actions do not interfere




Actions can be taken simultaneously.

### Actions interfere




If A is moved first, B won't be clear and cannot be moved.

# Interference

## Auxiliary definition: affects



### Definition (Affect)

Let $A$ be a set of state variables and $o = \langle c, e \rangle$ and $o' = \langle c', e' \rangle$ operators over $A$. Then $o$ **affects** $o'$ if there is $a \in A$ such that

1. $a$ is an atomic effect in $e$ and $a$ occurs in a formula in $e'$ or it occurs negatively in $c'$, or
2. $\neg a$ is an atomic effect in $e$ and $a$ occurs in a formula in $e'$ or it occurs positively in $c'$.

### Example

$\langle c, d \rangle$ affects $\langle \neg d, e \rangle$ and $\langle e, d \rhd f \rangle$.
$\langle c, d \rangle$ does not affect $\langle d, e \rangle$ nor $\langle e, \neg c \rangle$.

# Interference



**Definition (Interference)**

Operators $o$ and $o'$ interfere if $o$ affects $o'$ or $o'$ affects $o$.

**Example**

$\langle c, d \rangle$ and $\langle \neg d, e \rangle$ interfere.
$\langle c, d \rangle$ and $\langle e, f \rangle$ do not interfere.

# Interference



## Lemma

*Let $s$ be a state and $T$ a set of operators so that app$_T(s)$ is defined and no two operators interfere.*
*Then app$_T(s) =$ app$_{o_1;\ldots;o_n}(s)$ for any total ordering $o_1, \ldots, o_n$ of $T$.*

# The translation for parallel plans in CPC



**Definition**

Define $\mathcal{R}_2(A, A', O)$ as the conjunction of $\tau_A(O)$ and

$$\neg(o \wedge o')$$

for all $o \in O$ and $o' \in O$ such that $o$ and $o'$ interfere and $o \neq o'$.

### Definition (Bounded-length plans in CPC)

Existence of parallel plans length $t$ is represented by a formula over propositions $A^0 \cup \cdots \cup A^t \cup O^1 \cup \cdots \cup O^t$ where $A^i = \{a^i | a \in A\}$ for all $i \in \{0, \ldots, t\}$ and $O^i = \{o^i | o \in O\}$ for all $i \in \{1, \ldots t\}$ as

$$\Phi_t^{par} = \iota^0 \wedge \mathcal{R}_2(A^0, A^1, O^1) \wedge \cdots \wedge \mathcal{R}_2(A^{t-1}, A^t, O^t) \wedge G^t$$

where $\iota^0 = \bigwedge \{a^0 | a \in A, I(a) = 1\} \cup \{\neg a^0 | a \in A, I(a) = 0\}$ and $G^t$ is $G$ with propositions $a$ replaced by $a^t$.

### Theorem

*Let $\Phi_t^{par}$ be the formula for $\langle A, I, O, G\rangle$ and plan length $t$. The formula $\Phi_t^{par}$ is satisfiable if and only if there is a sequence of states $s_0, \ldots, s_t$ and sets $O_1, \ldots, O_t$ of non-interfering operators such that $s_0 = I$, $s_t \models G$ and $s_i = app_{O_i}(s_{i-1})$ for all $i \in \{1, \ldots, t\}$.*

# Why is optimality lost?



**For parallel plans there is no guarantee for smallest number of operators**

That a plan has the smallest number of time points does not guarantee that it has the smallest number of actions.

- Satisfiability algorithms return any satisfying valuation of $\Phi_i^{par}$, and this does not have to be the one with the smallest number of operators.
- There could be better solutions with more time points.

# Why is optimality lost?



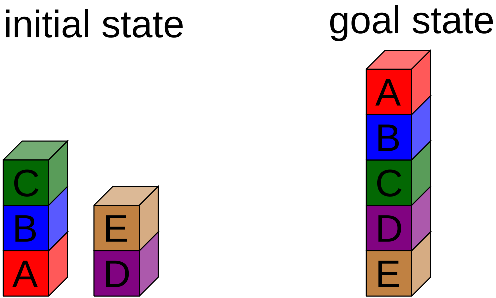
## Example

Let $I$ be a state such that $s \models \neg c \wedge \neg d \wedge \neg e \wedge \neg f$.
Let $G = c \wedge d \wedge e$.
Let

$$
\begin{aligned}
o_1 &= \langle \top, c \rangle \\
o_2 &= \langle \top, d \rangle \\
o_3 &= \langle \top, e \rangle \\
o_4 &= \langle \top, f \rangle \\
o_5 &= \langle f, c \wedge d \wedge e \rangle
\end{aligned}
$$

Now $\{o_1, o_2, o_3\}$ is a plan with one step, and $\{o_4\}; \{o_5\}$ is a plan with two steps. The first one has less time steps and corresponds to a satisfying valuation of both $\Phi_1^{par}$ and $\Phi_2^{par}$.

# Planning as satisfiability
## Example




The Davis-Putnam procedure solves the problem quickly:

- Formulae for lengths 1 to 4 shown unsatisfiable without any search.
- Formula for plan length 5 is satisfiable: 3 nodes in the search tree.
- Plans have 5 to 7 operators, optimal plan has 5.

# Planning as satisfiability

## Example

```
v0.9 13/08/1997 19:32:47
30 propositions 100 operators
Length 1
Length 2
Length 3
Length 4
Length 5
branch on -clear(b)[1] depth 0
branch on clear(a)[3] depth 1
Found a plan.
  0 totable(e,d)
  1 totable(c,b) fromtable(d,e)
  2 totable(b,a) fromtable(c,d)
  3 fromtable(b,c)
  4 fromtable(a,b)
Branches 2 last 2 failed 0; time 0.0
```

# Planning as satisfiability

## Example: valuations after unit propagation, after branching

```
     ON                   ON
  CLEARaaaabbbbccccddddeeeeTABLE
  abcdebcdeacdeabdeabceabcdabcde

0 FFTFTFFFFTFFFFTFFFFFFFFFTTFFTF
1 F TTTFFFFTFFFF FFFFFFFFFFTF TT
2    TFFFFF  FFF FFFFFTFFFFT  FT
3    FF FFF  FFFFTFFFFTFFFF  FFT
4   FFF FFFFTFFFFTFFFFTFFFF FFFT
5  FFFFTFFFFTFFFFTFFFFTFFFFFFFFT


0 FFTFTFFFFTFFFFTFFFFFFFFFTTFFTF
1 FFTTTFFFFTFFFFTFFFFFFFFFFTFFTT
2 F TTFFFFFTFFFF FFFFFTFFFFTF FT
3  TTFFFFFF FFFFFTFFFFTFFFFT FFT
4 TTFFFFFFFFTFFFFTFFFFTFFFFTFFFT
5 TFFFFTFFFFTFFFFTFFFFTFFFFFFFFT


0 FFTFTFFFFTFFFFTFFFFFFFFFTTFFTF
1 FFTTTFFFFTFFFFTFFFFFFFFFFTFFTT
2 FTTTFFFFFTFFFFFFFFFFTFFFFTFTFT
3 TTTFFFFFFFFFFFFTFFFFTFFFFTTFFT
4 TTFFFFFFFFTFFFFTFFFFTFFFFTFFFT
5 TFFFFTFFFFTFFFFTFFFFTFFFFFFFFT
```

# Planning as satisfiability

## Example: valuations after unit propagation, after branching

```
           012345   012345   012345
  clear(a) FF       FFF TT   FFFTTT
  clear(b) F    F   FF TTF   FFTTTF
  clear(c) TT  FF   TTTTFF   TTTTFF
  clear(d) FTTFFF   FTTFFF   FTTFFF
  clear(e) TTFFFF   TTFFFF   TTFFFF
    on(a,b) FFF  T   FFFFFT   FFFFFT
    on(a,c) FFFFFF   FFFFFF   FFFFFF
    on(a,d) FFFFFF   FFFFFF   FFFFFF
    on(a,e) FFFFFF   FFFFFF   FFFFFF
    on(b,a) TT  FF   TTT FF   TTTFFF
    on(b,c) FF  TT   FFFFTT   FFFFTT
    on(b,d) FFFFFF   FFFFFF   FFFFFF
    on(b,e) FFFFFF   FFFFFF   FFFFFF
    on(c,a) FFFFFF   FFFFFF   FFFFFF
    on(c,b) T  FFF   TT FFF   TTFFFF
    on(c,d) FFFTTT   FFFTTT   FFFTTT
    on(c,e) FFFFFF   FFFFFF   FFFFFF
    on(d,a) FFFFFF   FFFFFF   FFFFFF
    on(d,b) FFFFFF   FFFFFF   FFFFFF
    on(d,c) FFFFFF   FFFFFF   FFFFFF
    on(d,e) FFTTTT   FFTTTT   FFTTTT
    on(e,a) FFFFFF   FFFFFF   FFFFFF
    on(e,b) FFFFFF   FFFFFF   FFFFFF
    on(e,c) FFFFFF   FFFFFF   FFFFFF
    on(e,d) TFFFFF   TFFFFF   TFFFFF
ontable(a) TTT  F   TTTTTF   TTTTTF
ontable(b) FF  FF   FFF FF   FFFTFF
ontable(c) F  FFF   FF FFF   FFTFFF
ontable(d) TTFFFF   TTFFFF   TTFFFF
ontable(e) FTTTTT   FTTTTT   FTTTTT
```

# Planning as satisfiability

## Example: valuation for operators after plan has been found

```
                 01234
fromtable(a,b) ....T
fromtable(b,c) ...T.
fromtable(c,d) ..T..
fromtable(d,e) .T...
  totable(b,a) ..T..
  totable(c,b) .T...
  totable(e,d) T....
```