Principles of AI Planning

12. Planning as search: potential heuristics

Albert-Ludwigs-Universität Freiburg

Bernhard Nebel and Robert Mattmüller

January 8th, 2020

Motivation

Motivation: declarative heuristics

Previous chapters:

“Procedural” method for obtaining a heuristic

Solve an easier version of the problem.

We have studied two common simplification methods: relaxation and abstraction.

This chapter:

“Declarative” method for obtaining a heuristic

- Declaratively describe the information we want the heuristic estimator to exploit.
- Let a computer find a heuristic that fits the declarative description.

Motivation: potential heuristics

Example (potential heuristic in chess)

Evaluation function for chess position *s*
(from White's perspective; the higher, the better):

$$h(s) = 9 \cdot (\text{♕} - \text{♛}) + 5 \cdot (\text{♖} - \text{♜}) + 3 \cdot (\text{♗} - \text{♝}) + 3 \cdot (\text{♘} - \text{♞}) + 1 \cdot (\text{♙} - \text{♟})$$

where ♕, ♛, ♖, ♜, … is the number of white and black queens, rooks, etc. still on the board.

Question: Can we derive a similar heuristic for planning?

Answer: Yes! (Even declaratively!)

Potential Heuristics

Potential heuristics

Potential heuristics: idea

Heuristic design as an optimization problem:

- Define simple numerical state features $f_1, \dots, f_n$.
- Consider heuristics that are linear combinations of features:

$$h(s) = w_1 f_1(s) + \cdots + w_n f_n(s)$$

  with weights (potentials) $w_i \in \mathbb{R}$.
- Find potentials for which h is admissible and well-informed.

Motivation:

- declarative approach to heuristic design
- heuristic very fast to compute if features are

Potential heuristics

Definition (feature)

A (state) feature for a planning task is a numerical function defined on the states of the task: $f : S \to \mathbb{R}$.

Atomic features test if some atom is true in a state.

Definition (atomic feature)

Let $v = d$ be an atom of an FDR planning task.
Then the atomic feature $f_{v=d}$ is defined as:

$$f_{v=d}(s) = \begin{cases} 1 & \text{if } s \models v = d \\ 0 & \text{otherwise} \end{cases}$$

$\rightsquigarrow$ atomic features $\approx$ facts

Potential heuristics

Definition (potential heuristic)

A potential heuristic for a set of features $\mathscr{F} = \{f_1, \ldots, f_n\}$ is a heuristic function h defined as a linear combination of the features:

$$h(s) = w_1 f_1(s) + \cdots + w_n f_n(s)$$

with weights (potentials) $w_i \in \mathbb{R}$.

- We only consider atomic potential heuristics, which are based on the set of all atomic features.
- Example for a task with state variables v_1 and v_2 and $\mathscr{D}_{v_1} = \mathscr{D}_{v_2} = \{d_1, d_2, d_3\}$:

$$h(s) = 3f_{v_1=d_1} + {}^1\!/{}_2 f_{v_1=d_2} - 2f_{v_1=d_3} + {}^5\!/{}_2 f_{v_2=d_1}$$

How to set the weights?

We want to find good atomic potential heuristics:

- admissible
- consistent
- well-informed

Question: How to achieve this?

Answer: Linear programming.

Linear Programming

Goal: solve a system of linear inequalities over n real-valued variables while optimizing some linear objective function.

Example (Production domain)

Two sorts of items with time requirements and profit per item.

	Cutting	Assembly	Postproc.	Profit per item
(x) sort 1	25	60	68	30
(y) sort 2	75	60	34	40
per day	≤ 450	≤ 480	≤ 476	maximize!

Aim: Find numbers of pieces x of sort 1 and y of sort 2 produced per day such that resource constraints are met and objective function is maximized.

Example (ctd., formalization)

$$\text{maximize } z = 30x + 40y \text{ subject to:} \tag{1}$$

$$x \geq 0, \; y \geq 0 \tag{2}$$

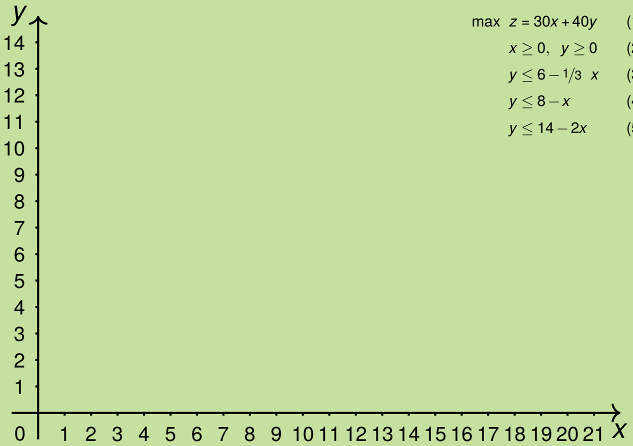
$$25x + 75y \leq 450 \tag{3}$$

$$60x + 60y \leq 480 \tag{4}$$

$$68x + 34y \leq 476 \tag{5}$$

- Line (1): Objective function
- Inequalities (2)–(5): Admissible solutions

Example (ctd., visualization)



$$\max \quad z = 30x + 40y \qquad (1)$$

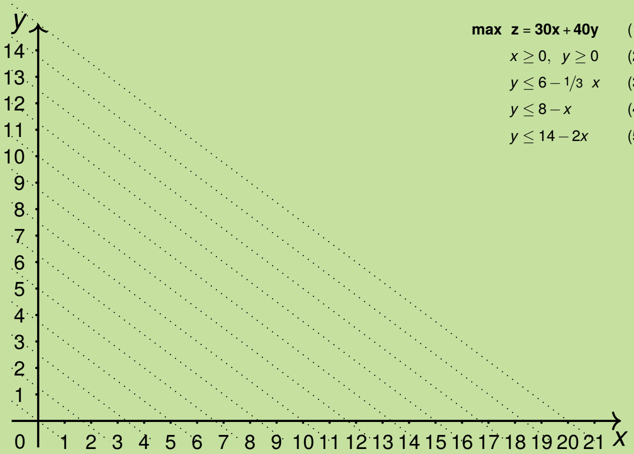
$$x \geq 0, \quad y \geq 0 \qquad (2)$$

$$y \leq 6 - 1/3 \; x \qquad (3)$$

$$y \leq 8 - x \qquad (4)$$

$$y \leq 14 - 2x \qquad (5)$$

Example (ctd., visualization)



$$\textbf{max } \mathbf{z = 30x + 40y} \quad (1)$$

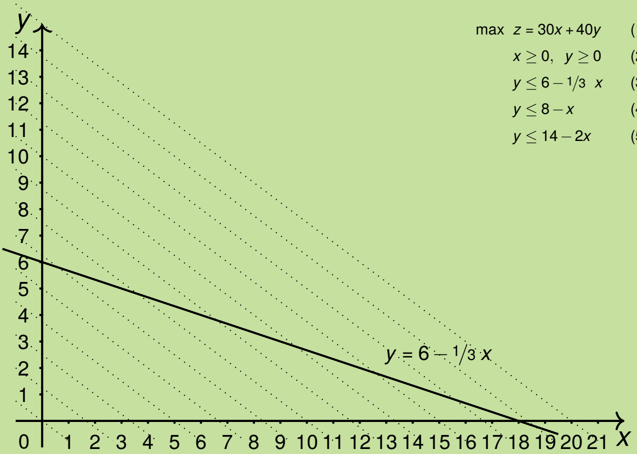
$$x \geq 0, \quad y \geq 0 \quad (2)$$

$$y \leq 6 - 1/3 \; x \quad (3)$$

$$y \leq 8 - x \quad (4)$$

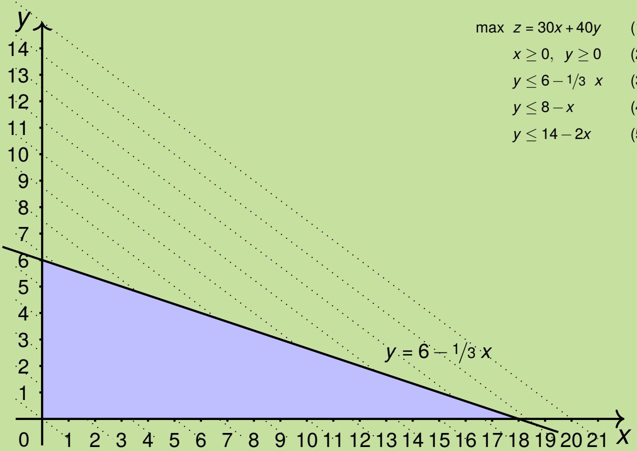
$$y \leq 14 - 2x \quad (5)$$

Example (ctd., visualization)



$$\max\ z = 30x + 40y \quad (1)$$
$$x \geq 0,\ \ y \geq 0 \quad (2)$$
$$y \leq 6 - 1/3\ x \quad (3)$$
$$y \leq 8 - x \quad (4)$$
$$y \leq 14 - 2x \quad (5)$$

Example (ctd., visualization)



$$\max \; z = 30x + 40y \quad (1)$$

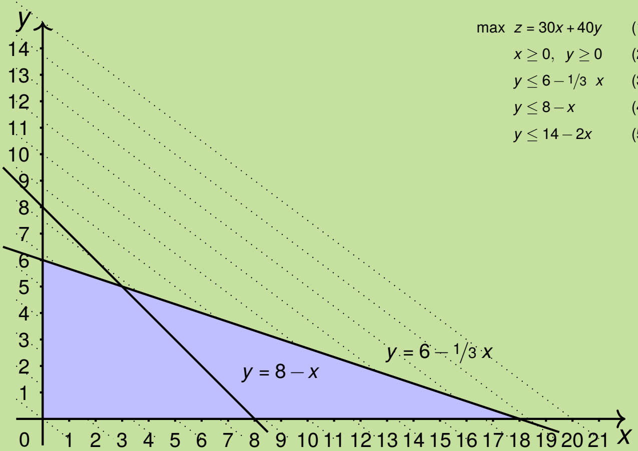
$$x \geq 0, \quad y \geq 0 \quad (2)$$

$$y \leq 6 - {}^{1}/_{3}\; x \quad (3)$$

$$y \leq 8 - x \quad (4)$$

$$y \leq 14 - 2x \quad (5)$$

Example (ctd., visualization)



$$\max \ z = 30x + 40y \qquad (1)$$

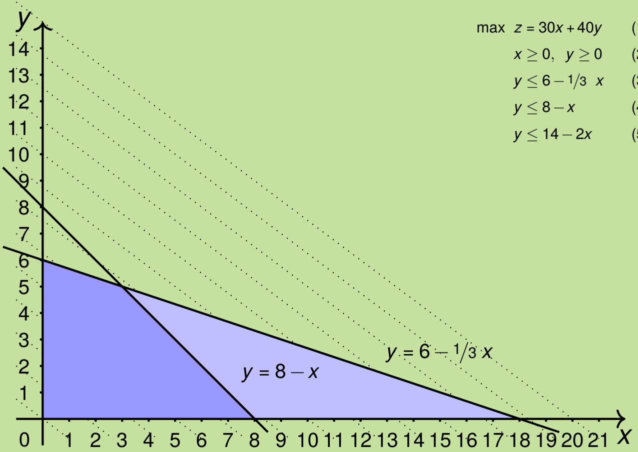
$$x \geq 0, \quad y \geq 0 \qquad (2)$$

$$y \leq 6 - {}^{1}\!/_{3}\ x \qquad (3)$$

$$y \leq 8 - x \qquad (4)$$

$$y \leq 14 - 2x \qquad (5)$$

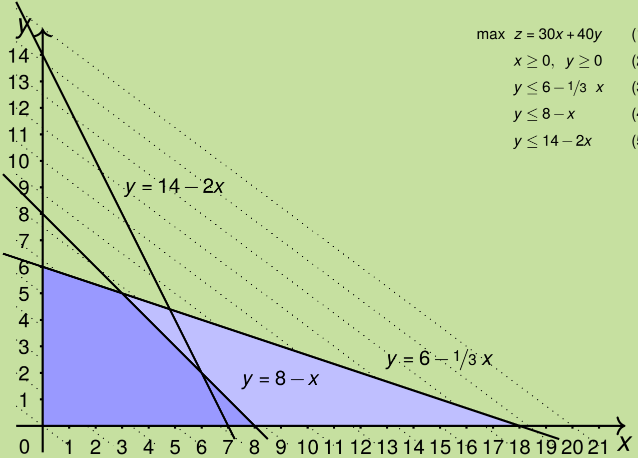
Example (ctd., visualization)



$$\max \quad z = 30x + 40y \qquad (1)$$
$$x \geq 0, \quad y \geq 0 \qquad (2)$$
$$y \leq 6 - 1/3 \ x \qquad (3)$$
$$y \leq 8 - x \qquad (4)$$
$$y \leq 14 - 2x \qquad (5)$$

UNI FREIBURG

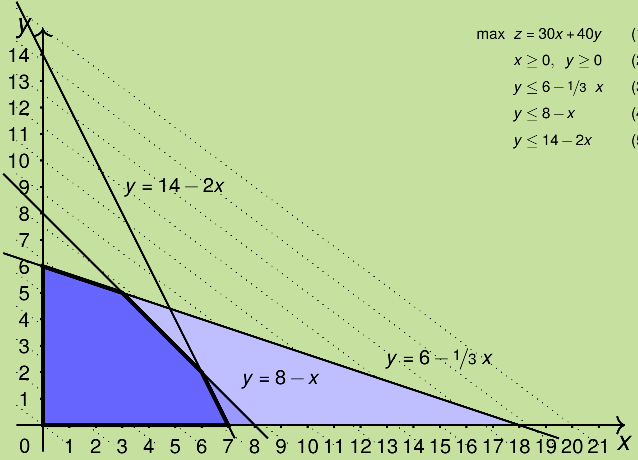
Example (ctd., visualization)



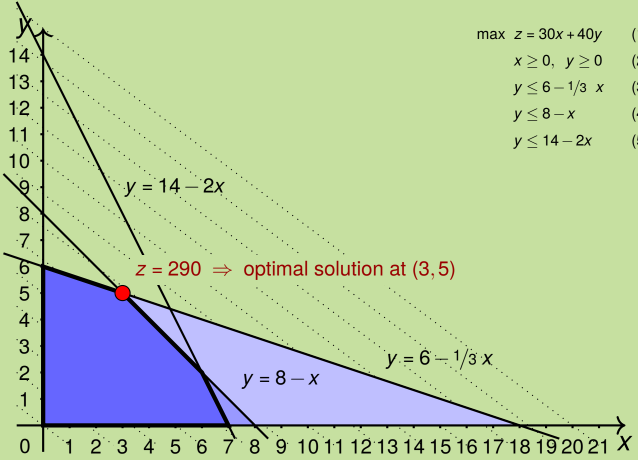
$$\max \quad z = 30x + 40y \qquad (1)$$
$$x \geq 0, \quad y \geq 0 \qquad (2)$$
$$y \leq 6 - {}^{1}/_{3}\ x \qquad (3)$$
$$y \leq 8 - x \qquad (4)$$
$$y \leq 14 - 2x \qquad (5)$$

Example (ctd., visualization)

$$\max \quad z = 30x + 40y \qquad (1)$$
$$x \geq 0, \quad y \geq 0 \qquad (2)$$
$$y \leq 6 - 1/3 \ x \qquad (3)$$
$$y \leq 8 - x \qquad (4)$$
$$y \leq 14 - 2x \qquad (5)$$



Example (ctd., visualization)



$$\max \; z = 30x + 40y \quad (1)$$

$$x \geq 0, \; y \geq 0 \quad (2)$$

$$y \leq 6 - 1/3 \; x \quad (3)$$

$$y \leq 8 - x \quad (4)$$

$$y \leq 14 - 2x \quad (5)$$

Definition (Linear program)

A linear program (LP) over variables $x_1, \ldots, x_n$ consists of

- m linear constraints of the form

$$\sum_{i=1}^{n} a_{ji} x_i \leq b_j$$

  with $a_{ji} \in \mathbb{R}$ for all $j = 1, \ldots, m$ and $i = 1, \ldots, n$, and
- a linear objective function to be maximized ($x_i \geq 0$):

$$\sum_{i=1}^{n} c_i x_i$$

  with $c_i \in \mathbb{R}$ for all $i = 1, \ldots, n$.

Linear Programming

Solution of an LP:
assignment of values to the x_i satisfying the constraints and maximizing the objective function.

Solution algorithms:

- Usually: simplex algorithm (worst-case exponential).
- There are also polynomial-time algorithms.

Transition normal form

Standard description of LP-based derivation of potentials assumes transition normal form.

Assumption (for the rest of the chapter): only SAS$^+$ tasks.

Notation: variables occurring in conditions and effects.

Definition ($vars(\varphi), vars(e)$)

For a logical formula φ over finite-domain variables $\mathcal{V}$, $vars(\varphi)$ denotes the set of finite-domain variables occurring in φ.

For an effect e over finite-domain variables $\mathcal{V}$, $vars(e)$ denotes the set of finite-domain variables occurring in e.

Transition normal form

Definition (transition normal form)

An SAS$^+$ planning task $\Pi = \langle \mathcal{V}, I, O, \gamma \rangle$ is in transition normal form (TNF) if

- for all $o \in O$, $vars(pre(o)) = vars(eff(o))$, and
- $vars(\gamma) = \mathcal{V}$.

In words, an operator in TNF must mention the same variables in the precondition and effect, and a goal in TNF must mention all variables (= specify exactly one goal state).

Converting operators to TNF: violations

There are two ways in which an operator o can violate TNF:

- There exists a variable $v \in vars(pre(o)) \setminus vars(eff(o))$.
- There exists a variable $v \in vars(eff(o)) \setminus vars(pre(o))$.

The first case is easy to address: if $v = d$ is a precondition with no effect on v, just add the effect $v := d$.

Example (TNF: adding effects)

Let $o = \langle x = 0 \land y = 0, y := 1 \rangle$.

Fix: rewrite $o = \langle x = 0 \land y = 0, x := 0 \land y := 1 \rangle$.

Converting operators to TNF: violations

The second case is more difficult: if we have the effect $v := d$ but no precondition on v, how can we add a precondition on v without changing the meaning of the operator (and without introducing exponentially many new operators)?

Example (TNF: adding precondition)

Let $o = \langle \top, y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$ with $\mathscr{D}_{y_i} = \{0, 1\}$ for all i.

One possible fix: rewrite o as set of operators

$$o_{00\ldots 0} = \langle y_1 = 0 \wedge y_2 = 0 \wedge \cdots \wedge y_n = 0,\ y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$$
$$o_{00\ldots 1} = \langle y_1 = 0 \wedge y_2 = 0 \wedge \cdots \wedge y_n = 1,\ y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$$
$$\vdots$$
$$o_{11\ldots 1} = \langle y_1 = 1 \wedge y_2 = 1 \wedge \cdots \wedge y_n = 1,\ y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$$

Problem: 2^n new operators (exponentially many!)

Converting operators to TNF: violations

The second case is more difficult: if we have the effect $v := d$ but no precondition on v, how can we add a precondition on v without changing the meaning of the operator (and without introducing exponentially many new operators)?

Example (TNF: adding precondition (ctd.))

Let $o = \langle \top, y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$ with $\mathscr{D}_{y_i} = \{0, 1\}$ for all i.

Better fix: rewrite $o = \langle y_1 = \mathit{don't\_care} \wedge y_2 = \mathit{don't\_care} \wedge \cdots \wedge y_n = \mathit{don't\_care},\ y_1 := 1 \wedge \cdots \wedge y_n := 1 \rangle$ and make sure that every variable can take its *don't_care* value for free.

Converting Operators to TNF

Formally:

1. For every variable v, add a new auxiliary value u to its domain.
2. For every variable v and value $d \in \mathscr{D}_v \setminus \{u\}$, add a new operator to change the value of v from d to u at no cost: $\langle v = d, v := u \rangle$.
3. For all operators o and all variables $v \in \mathit{vars}(\mathit{eff}(o)) \setminus \mathit{vars}(\mathit{pre}(o))$, add the precondition $v = u$ to $\mathit{pre}(o)$.

Properties:

- Transformation can be computed in linear time.
- Due to the auxiliary values, there are new states and transitions in the induced transition system, but all path costs between original states remain the same.

Converting Goals to TNF

- The auxiliary value idea can also be used to convert the goal γ to TNF.
- For every variable $v \notin vars(\gamma)$, add the condition $v = u$ to γ.

With these ideas, every SAS$^+$ planning task can be converted into transition normal form in linear time.

Producers and consumers

Assume that $\Pi = \langle \mathscr{V}, I, O, \gamma \rangle$ is in TNF.

Definition (producers and consumers)

Fact $v = d$ is produced by operator $o \in O$
if $v = d$ is an effect of o, but not a precondition of o.

Fact $v = d$ is consumed by operator $o \in O$
if $v = d$ is a precondition of o, but not an effect of o.

Admissible and consistent potential heuristics

Assume feature set $\mathscr{F} = \{f_{v=d} \mid v \in \mathscr{V}, d \in \mathscr{D}_v\}$ and corresponding potentials $\mathscr{W} = \{w_{v=d} \mid v \in \mathscr{V}, d \in \mathscr{D}_v\}$.

Constraints on potentials characterize (= are necessary and sufficient for) admissible and consistent atomic potential heuristics:

Goal-awareness constraint

$$\sum_{\text{goal fact } v=d} w_{v=d} = 0$$

Example (Goal-awareness constraint)

$\mathscr{V} = \{x, y\}$, $\mathscr{D}_x = \mathscr{D}_y = \{0, 1, u\}$, $\gamma = (x = 1 \wedge y = u)$.

Goal-awareness constraint: $w_{x=1} + w_{y=u} = 0$.

Admissible and consistent potential heuristics

Theorem

For a task in TNF, a potential heuristic with feature set $\mathscr{F} = \{f_{v=d} \mid v \in \mathscr{V}, d \in \mathscr{D}_v\}$ and corresponding potentials $\mathscr{W} = \{w_{v=d} \mid v \in \mathscr{V}, d \in \mathscr{D}_v\}$ that satisfy the goal-awareness constraint is goal-aware.

Proof.

See blackboard. □

Admissible and consistent potential heuristics

Consistency constraints (for all operators $o \in O$)

$$\sum_{\text{fact } v=d \text{ consumed by } o} w_{v=d} - \sum_{\text{fact } v=d \text{ produced by } o} w_{v=d} \leq cost(o)$$

Example (Consistency constraint)

$\mathscr{V} = \{x, y\}$, $\mathscr{D}_x = \mathscr{D}_y = \{0, 1, u\}$,
$o = \langle x = 0 \wedge y = 0, x := 0 \wedge y := 1 \rangle$ with $cost(o) = 1$.

Then o consumes $y = 0$ and produces $y = 1$.

Consistency constraint for o: $w_{y=0} - w_{y=1} \leq 1$.

Admissible and consistent potential heuristics

Theorem

For a task in TNF, a potential heuristic with feature set $\mathcal{F} = \{f_{v=d} \mid v \in \mathcal{V}, d \in \mathcal{D}_v\}$ and corresponding potentials $\mathcal{W} = \{w_{v=d} \mid v \in \mathcal{V}, d \in \mathcal{D}_v\}$ that satisfy the consistency constraints for all operators o is consistent.

Proof.

Homework exercise. □

Admissible and consistent potential heuristics

Remarks:

- all linear constraints $\rightsquigarrow$ LP
- goal-aware and consistent $\rightsquigarrow$ admissible and consistent

Well-informed potential heuristics

How to find a well-informed potential heuristic?
$\rightsquigarrow$ encode quality metric in the objective function and use LP solver to find a heuristic maximizing it

Examples:

- maximize heuristic value of a given state (e.g., initial state)
- maximize average heuristic value of all states (including unreachable ones)
- maximize average heuristic value of some sample states

Well-informed potential heuristics

LP encoding for maximizing heuristic value of initial state while guaranteeing goal-awareness and consistency:

maximize $\sum_{\text{fact } v=d \text{ satisfied in } s_0} w_{v=d}$ subject to:

goal constraint

consistency constraint for o for all o

Remarks

- Further constraints can be added to the LP to obtain stronger heuristics.
- The hard work is done by the LP solver.

Summary

Summary

- Declarative method for obtaining a heuristic
- Potential heuristics are linear combinations of features.
- Needed: features and weights (potentials)
- Features: facts (for us; can be generalized)
- Potentials: computed by solving an LP, given constraints that encode goal-awareness and consistency, and an objective function to maximize heuristic value.
- Necessary prerequisite: without loss of generality, task is in transition normal form (same variables in preconditions and effects, all variables mentioned in the goal).

Credit

Slides heavily based on those by Gabriele Röger and Thomas Keller (Uni Basel).