Solving Non-deterministic Planning Problems with Pattern Database Heuristics

Pascal Bercher
Institute of Artificial Intelligence
University of Ulm, Germany

Robert Mattmüller
Department of Computer Science
University of Freiburg, Germany

KI 2009, Paderborn

Given: A non-deterministic planning problem.

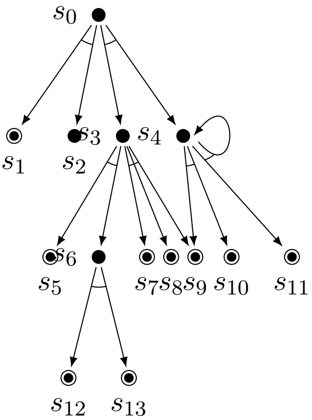
Desired: An acyclic solution, i.e. a strong plan.
(Success, regardless of non-deterministic outcome.)

Given:

Non-deterministic planning problem $\mathcal{P} = (Var, A, s_0, G)$ with:

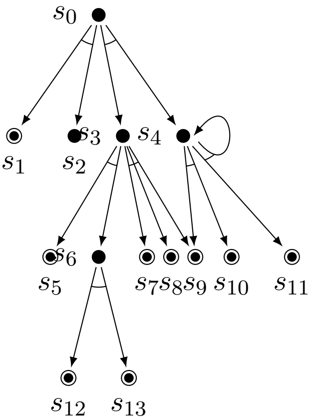
- Var, finite set of *state variables*.
  $S = 2^{Var}$ is the state space.
- A, finite set of *actions* $a = \langle pre(a), eff(a) \rangle$ and:
  - $pre(a) \subseteq Var$ and
  - $eff(a) = \{ \langle add_i, del_i \rangle \mid add_i, del_i \subseteq Var \text{ and } i \in \{1, \dots, n\} \}$.
  - Its application (if $pre(a) \subseteq s$) leads to:
    $app(s, a) = \{ (s \setminus del) \cup add \mid \langle add, del \rangle \in eff(a) \}$
- $s_0 \in S$, the *initial state*.
- $G \subseteq Var$, the *goal description*.

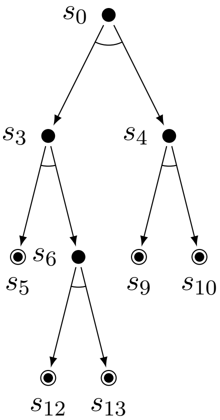
Desired:
Strong plan.



Desired:

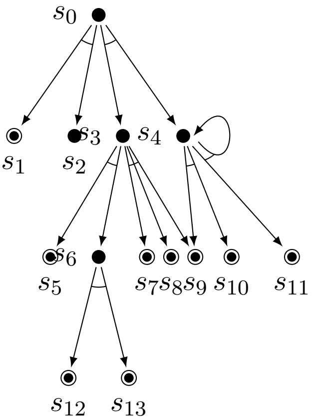
Strong plan.

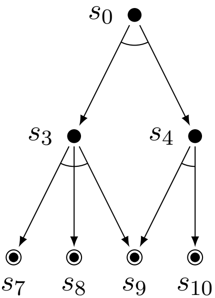




Desired:

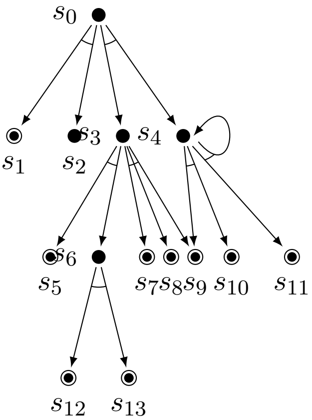
Strong plan.

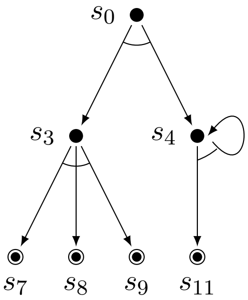




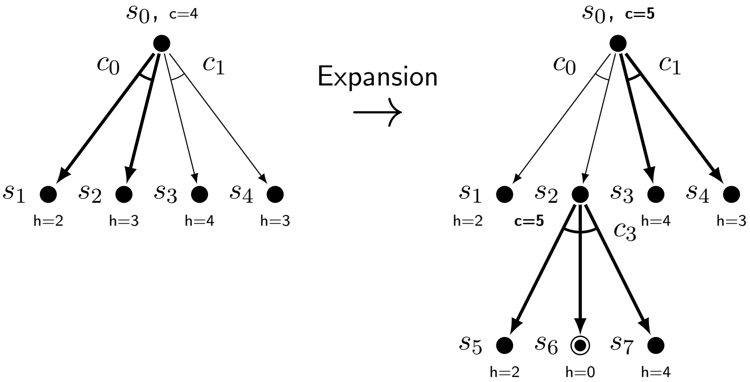
Desired:

Strong plan.

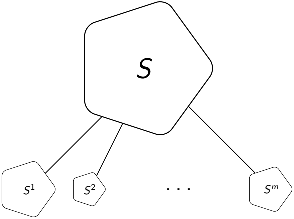




Search Algorithm, modification of AO*



Use abstraction to simplify the problem:



Map the search space S to abstract search spaces S^i with $|S^i| \ll |S|$, f.a. $P_i \in P$, P finite pattern collection.

Compute $h(s), s \in S$, on basis of all $h^i(s^i), P_i \in P$.

The *abstraction* $\mathcal{P}^i = (Var^i, A^i, s_0^i, G^i)$ is the planning problem $\mathcal{P}$, restricted to the pattern $P_i \subseteq Var$:

- $Var^i Var \cap P_i = P_i$,
- For $var \subseteq Var$ let $var^i var \cap P_i$. Then:
  $a^i \langle pre(a)^i, \{ \langle add^i, del^i \rangle \mid \langle add, del \rangle \in eff(a) \} \rangle$ for $a \in A$.
  Now, $A^i \{ a^i \mid a \in A \}$.
- $s_0^i s_0 \cap P_i$
- $G^i G \cap P_i$.

Given: Pattern collection $P = \{P_1 \dots, P_m\}$.

For each pattern $P_i \in P$, calculate true cost value for complete S^i.
Calcuation is done by a complete exhaustive search.

(*True* means: prefer shallow solution graphs.)

Compute: $h(s) \max_{P_i \in P} h^i(s^i)$. ($h$ is admissible.)

Make use of additivity.
A pattern collection P is called *additive*, if for all states $s \in S$:
$\sum_{P_i \in P} h^i(s^i) \leq cost^*(s)$.

Known from classical planning:

Theorem (informal)

If there is no action $a \in A$ that affects variables in more than one pattern from P, then P is additive

Make use of additivity.

A pattern collection P is called *additive*, if for all states $s \in S$:

$$\sum_{P_i \in P} h^i(s^i) \leq cost^*(s).$$

Known from classical planning:

Theorem (formal)

If for all $a \in A$ and for all patterns $P_i \in P$ holds:

If $P_i \cap effvar(a) \neq \emptyset$, then $P_j \cap effvar(a) = \emptyset$ for all $P_j \in P$ with $P_j \neq P_i$, where $effvar(a) = \bigcup_{\langle add, del \rangle \in eff(a)} add \cup del$.

Then P is additive.

$\mathcal{P} = (\{a, b, c, d, e\}, A, \{a\}, \{b, c, d, e\})$ with $A = \{a_1, \ldots, a_9\}$ and:

$$a_1 = \langle \{a\}, \{\langle \{b\}, \{a\} \rangle, \langle \{c\}, \{a\} \rangle\} \rangle$$
$$a_2 = \langle \{b\}, \{\langle \{e\}, \emptyset \rangle, \langle \{d\}, \emptyset \rangle\} \rangle$$
$$a_3 = \langle \{c\}, \{\langle \{e\}, \emptyset \rangle, \langle \{d\}, \emptyset \rangle\} \rangle$$
$$a_4 = \langle \{b, d\}, \{\langle \{c\}, \emptyset \rangle\} \rangle$$
$$a_5 = \langle \{c, d\}, \{\langle \{b\}, \emptyset \rangle\} \rangle$$
$$a_6 = \langle \{b, e\}, \{\langle \{c\}, \emptyset \rangle\} \rangle$$
$$a_7 = \langle \{c, e\}, \{\langle \{b\}, \emptyset \rangle\} \rangle$$
$$a_8 = \langle \{b, c, d\}, \{\langle \{e\}, \emptyset \rangle\} \rangle$$
$$a_9 = \langle \{b, c, e\}, \{\langle \{d\}, \emptyset \rangle\} \rangle$$

Additivity (Example)

$\mathcal{P} = (\{a, b, c, d, e\}, A, \{a\}, \{b, c, d, e\})$ with $A = \{a_1, \ldots, a_9\}$ and:

$$a_1 = \langle\{a\}, \{\langle\{b\}, \{a\}\rangle, \langle\{c\}, \{a\}\rangle\}\rangle$$
$$a_2 = \langle\{b\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_3 = \langle\{c\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_4 = \langle\{b, d\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_5 = \langle\{c, d\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_6 = \langle\{b, e\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_7 = \langle\{c, e\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_8 = \langle\{b, c, d\}, \{\langle\{e\}, \emptyset\rangle\}\rangle$$
$$a_9 = \langle\{b, c, e\}, \{\langle\{d\}, \emptyset\rangle\}\rangle$$

Now, consider the pattern collection $P = \{\{a, b, c\}, \{d, e\}\}$.

$\mathcal{P} = (\{a, b, c, d, e\}, A, \{a\}, \{b, c, d, e\})$ with $A = \{a_1, \ldots, a_9\}$ and:

$$a_1 = \langle\{a\}, \{\langle\{b\}, \{a\}\rangle, \langle\{c\}, \{a\}\rangle\}\rangle$$
$$a_2 = \langle\{b\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_3 = \langle\{c\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_4 = \langle\{b, d\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_5 = \langle\{c, d\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_6 = \langle\{b, e\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_7 = \langle\{c, e\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_8 = \langle\{b, c, d\}, \{\langle\{e\}, \emptyset\rangle\}\rangle$$
$$a_9 = \langle\{b, c, e\}, \{\langle\{d\}, \emptyset\rangle\}\rangle$$

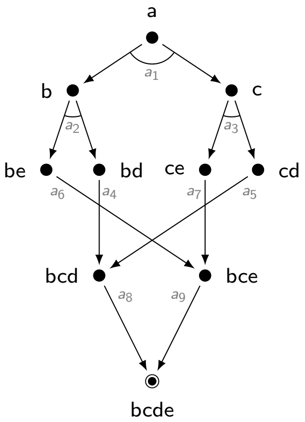
Now, consider the pattern collection $P = \{\{a, b, c\}, \{d, e\}\}$.

$\mathcal{P} = (\{a, b, c, d, e\}, A, \{a\}, \{b, c, d, e\})$ with $A = \{a_1, \ldots, a_9\}$ and:

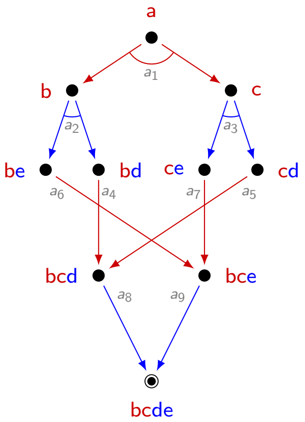
$$a_1 = \langle\{a\}, \{\langle\{b\}, \{a\}\rangle, \langle\{c\}, \{a\}\rangle\}\rangle$$
$$a_2 = \langle\{b\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_3 = \langle\{c\}, \{\langle\{e\}, \emptyset\rangle, \langle\{d\}, \emptyset\rangle\}\rangle$$
$$a_4 = \langle\{b, d\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_5 = \langle\{c, d\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_6 = \langle\{b, e\}, \{\langle\{c\}, \emptyset\rangle\}\rangle$$
$$a_7 = \langle\{c, e\}, \{\langle\{b\}, \emptyset\rangle\}\rangle$$
$$a_8 = \langle\{b, c, d\}, \{\langle\{e\}, \emptyset\rangle\}\rangle$$
$$a_9 = \langle\{b, c, e\}, \{\langle\{d\}, \emptyset\rangle\}\rangle$$

Now, consider the pattern collection $P = \{\{a, b, c\}, \{d, e\}\}$.
Only the effect variables matter!

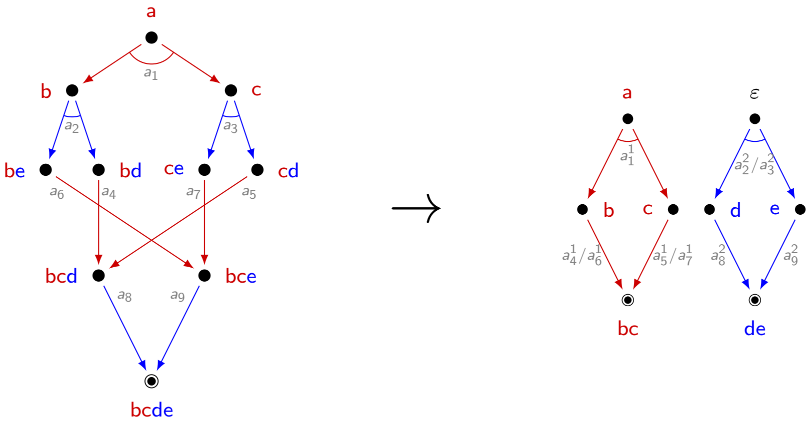
Additivity (Example, cont'd)



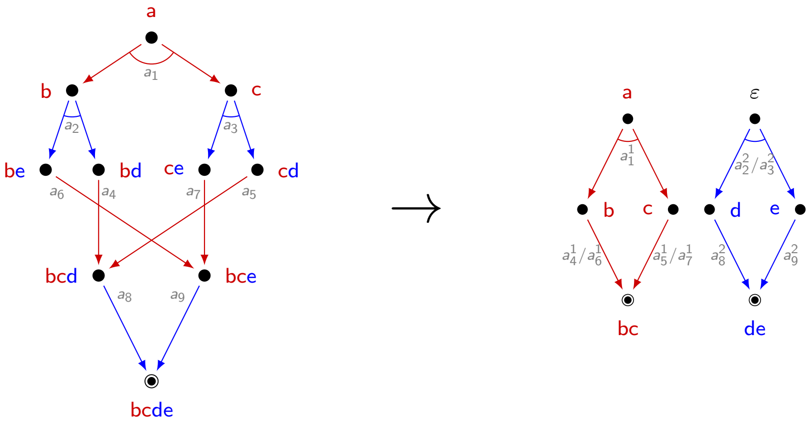
Additivity (Example, cont'd)



Additivity (Example, cont'd)



Additivity (Example, cont'd)



Example:
$h(\{a\}) = h^1(\{a\}^1) + h^2(\{a\}^2) = h^1(\{a\}) + h^2(\emptyset) = 2 + 2 = 4 = cost^*(\{a\})$.

Let $\mathcal{M}$ be a set of additive pattern collections.
$h^{\mathcal{M}}(s) := \max_{P \in \mathcal{M}} \sum_{P_i \in P} h^i(s)$.

$h^{\mathcal{M}}$ (and in particular, every single h^i) is admissible.

How to find $\mathcal{M}$?
Current research. (Here: still domain-dependent by hand.)

Summary

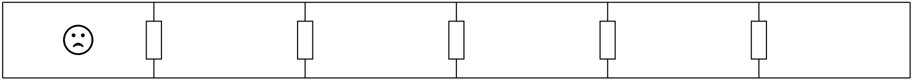
–

Encoded two domains:

- Chain of Rooms
  sequential subproblems, well-informed heuristic.
- Coin Flip
  parallel subproblems, perfectly informed heuristic.

Comparison with GAMER[1]. *Important* differences:

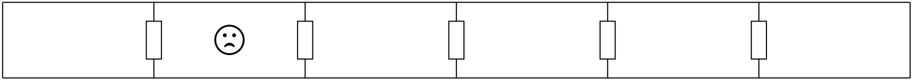
- Optimal solutions[1] vs. suboptimal solutions.
- Regression[1] vs. progression.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

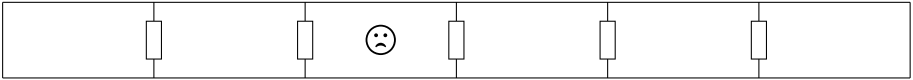
Goal:
Having visited each room at least once.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

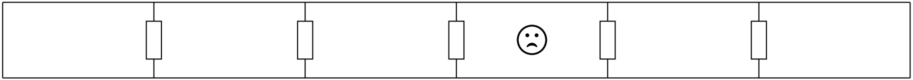
Goal:
Having visited each room at least once.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

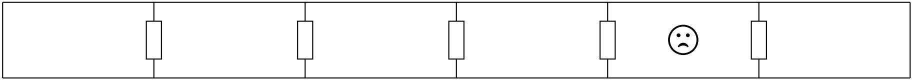
Goal:
Having visited each room at least once.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

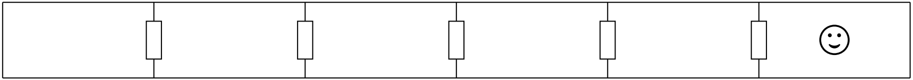
Goal:
Having visited each room at least once.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

Goal:
Having visited each room at least once.



Actions:

- Go into neighboring room, if door is open.
- Turn light on, if it is off
  (non-deterministically: door open/closed).
- Open door.

Goal:
Having visited each room at least once.

Chain of Rooms

#rooms	AO* Planner, FF			AO* Planner, PDB						Gamer	
	search	mem	nodes	pre	search	sum	mem	|PDB|	nodes	search	|BDD|
20	2	3	236	4	1	5	3	315	274	6	16699
40	15	13	873	7	2	9	17	679	1138	23	130657
60	69	33	1909	16	6	22	46	1043	2602	—	—
80	250	69	3346	26	15	41	100	1407	4666	—	—
100	655	133	5183	41	33	74	190	1771	7330	—	—
120	1497	214	7419	61	73	134	319	2135	10594	—	—
140	—	—	—	91	117	208	495	2499	14458	—	—
160	—	—	—	127	194	321	736	2863	18922	—	—
180	—	—	—	177	314	491	1050	3227	23986	—	—

Dashes indicate that the time-out of 30 minutes or the memory bound of 1500 MB was exceeded.

Given:

- n coins, all initially in a bag.
- Actions:
  - Flip coin, if not flipped before.
  - Reverse coin, if flipped before.

Desired:

Strategy, that all coins show heads.

Coin Flip

#coins	AO* Planner, FF			AO* Planner, PDB						Gamer	
	search	mem	nodes	pre	search	sum	mem	|PDB|	nodes	search	|BDD|
20	—	—	—	2	1	3	4	60	1125	—	—
40	—	—	—	3	4	7	27	120	4645	—	—
60	—	—	—	4	19	23	85	180	10565	—	—
80	—	—	—	7	60	67	180	240	18885	—	—
100	—	—	—	11	217	328	366	300	29605	—	—
120	—	—	—	18	306	324	597	360	42725	—	—
140	—	—	—	23	533	556	905	420	58245	—	—
160	—	—	—	34	908	942	1307	480	76165	—	—

Dashes indicate that the time-out of 30 minutes or the memory bound of 1500 MB was exceeded.

- Presented fomalization for domain-independent pattern database heuristics in non-deterministic planning.
- Generalization of additivity criterion.
- Benchmarks look promising.

Future Work

- Automatic pattern selection (Bachelor thesis finished).
- Strong plans $\rightarrow$ strong *cyclic* plans.
  - Search algorithm, LAO*.
  - Pattern database heuristics: Admissibility/Additivity?
- Multi-valued state variables.

Thank you!