State-Dependent Cost Partitionings for Cartesian Abstractions in Classical Planning

Thomas Keller[1] Florian Pommerening[1] Jendrik Seipp[1]
Florian Geißer[2] Robert Mattmüller[2]

[1]University of Basel
[2]University of Freiburg

September 28, 2016

KI 2016, Klagenfurt


The World of
Classical Planning

The World of **Classical Planning**

- Given:
  - Initial state
  - Goal
  - Actions
- Find: Plan from initial state to goal

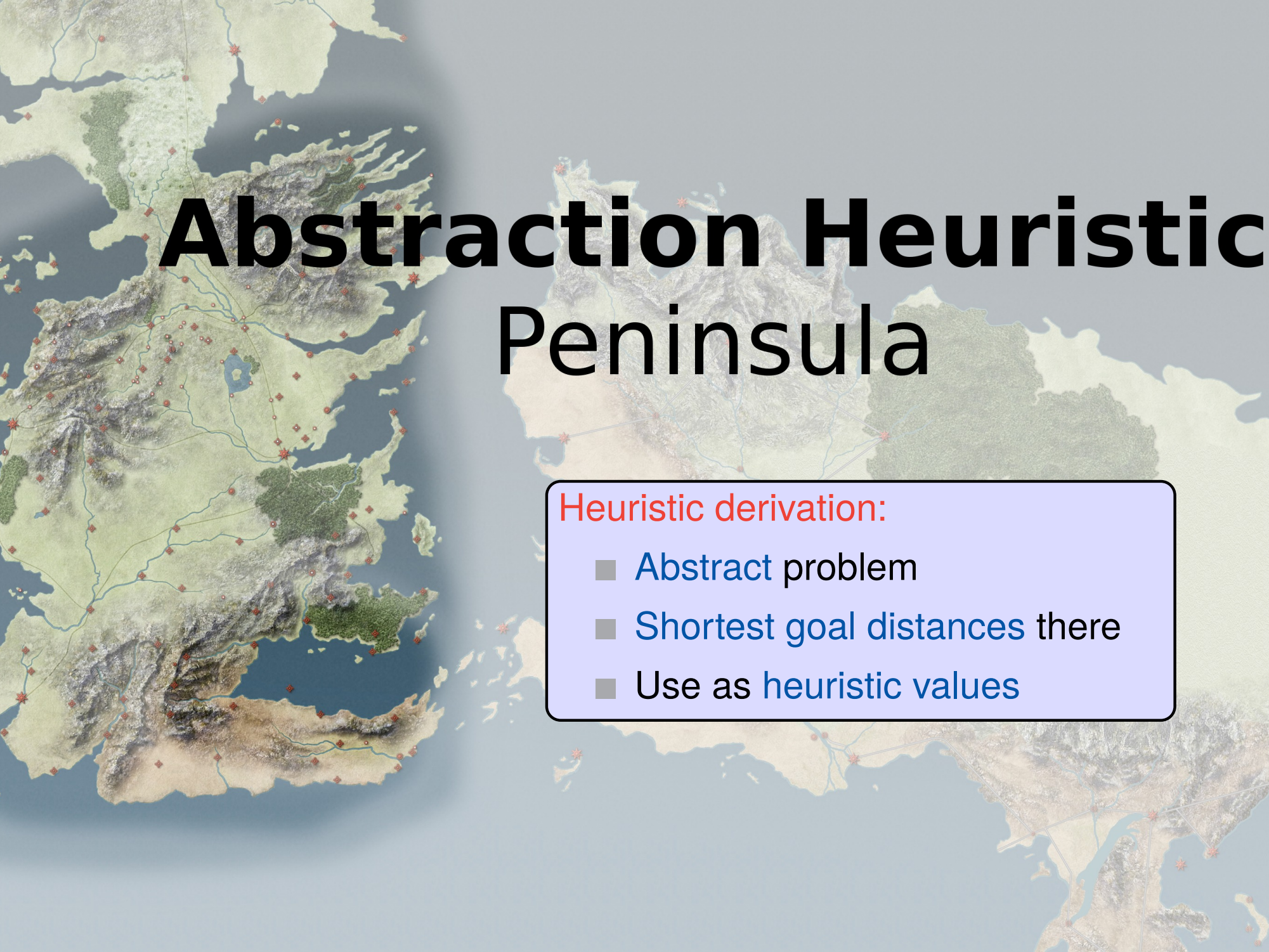

Planning as Heuristic
Search Continent

Planning as Heuristic Search Continent

Approach:

- State-space search for a plan
- Guided by a distance heuristic …
- …that is automatically derived


Abstraction Heuristic
Peninsula



Abstraction Heuristic
Peninsula

Heuristic derivation:

- Abstract problem
- Shortest goal distances there
- Use as heuristic values


Cost Partitioning
Bay

Cost Partitioning

Bay

- Challenge: single abstraction uninformative
- Solution: multiple abstractions

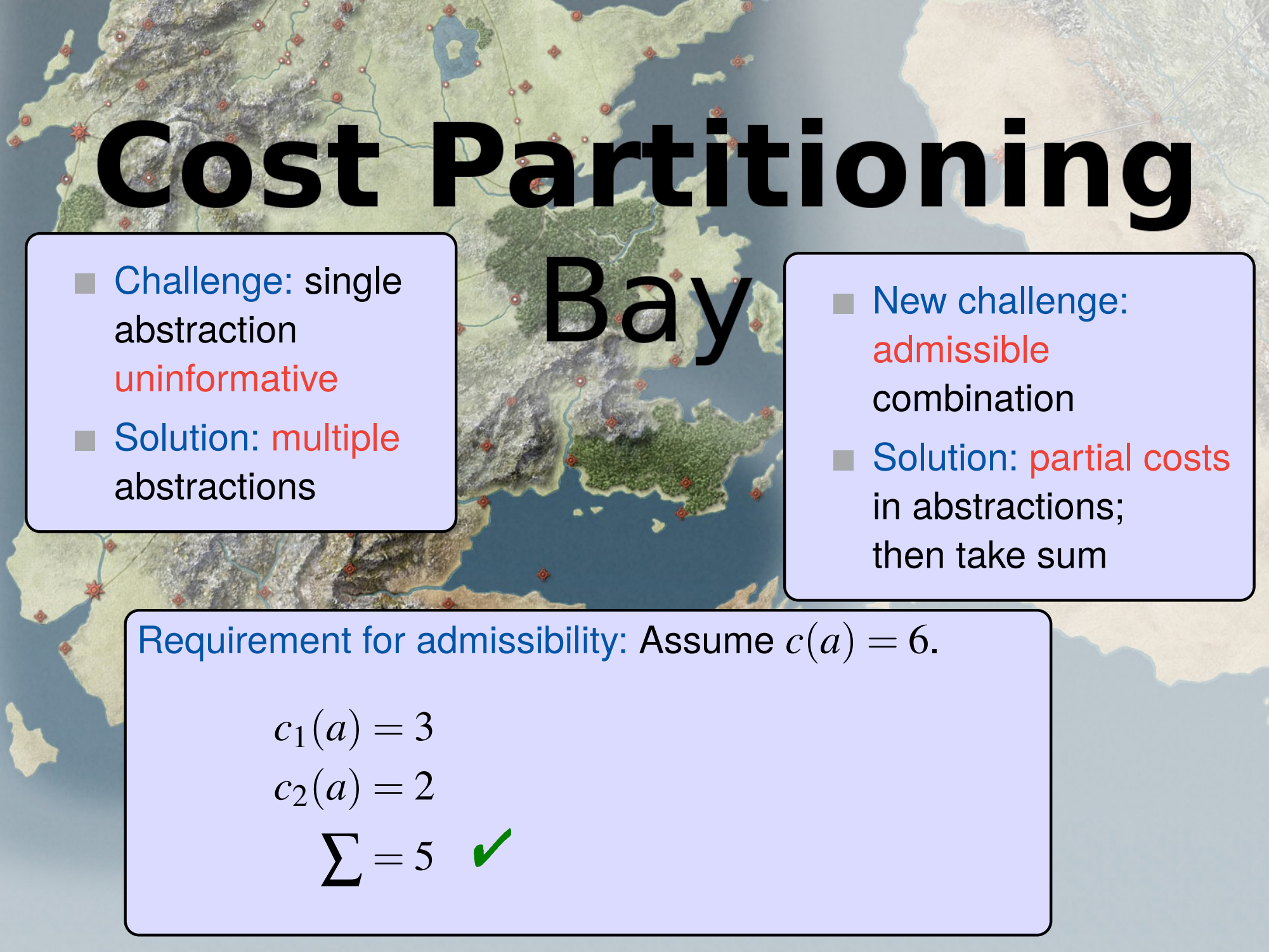
Cost Partitioning

Bay

- Challenge: single abstraction uninformative
- Solution: multiple abstractions

- New challenge: admissible combination
- Solution: partial costs in abstractions; then take sum

Cost Partitioning



- Challenge: single abstraction uninformative
- Solution: multiple abstractions

- New challenge: admissible combination
- Solution: partial costs in abstractions; then take sum

Requirement for admissibility: Assume $c(a) = 6$.

$$c_1(a) = 3$$
$$c_2(a) = 2$$
$$\sum = 5 \quad \checkmark$$

Cost Partitioning

Bay

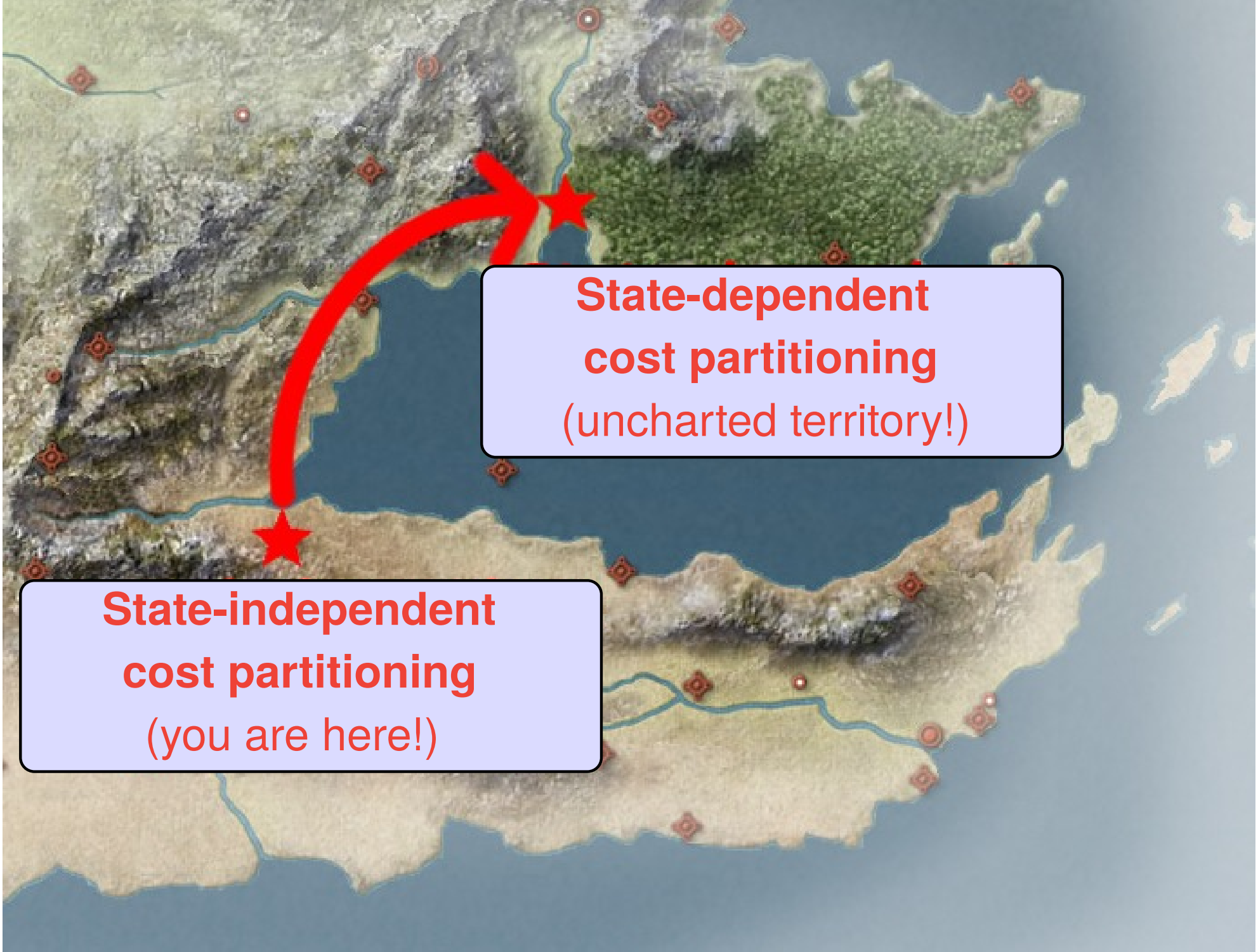
- Challenge: single abstraction uninformative
- Solution: multiple abstractions

- New challenge: admissible combination
- Solution: partial costs in abstractions; then take sum

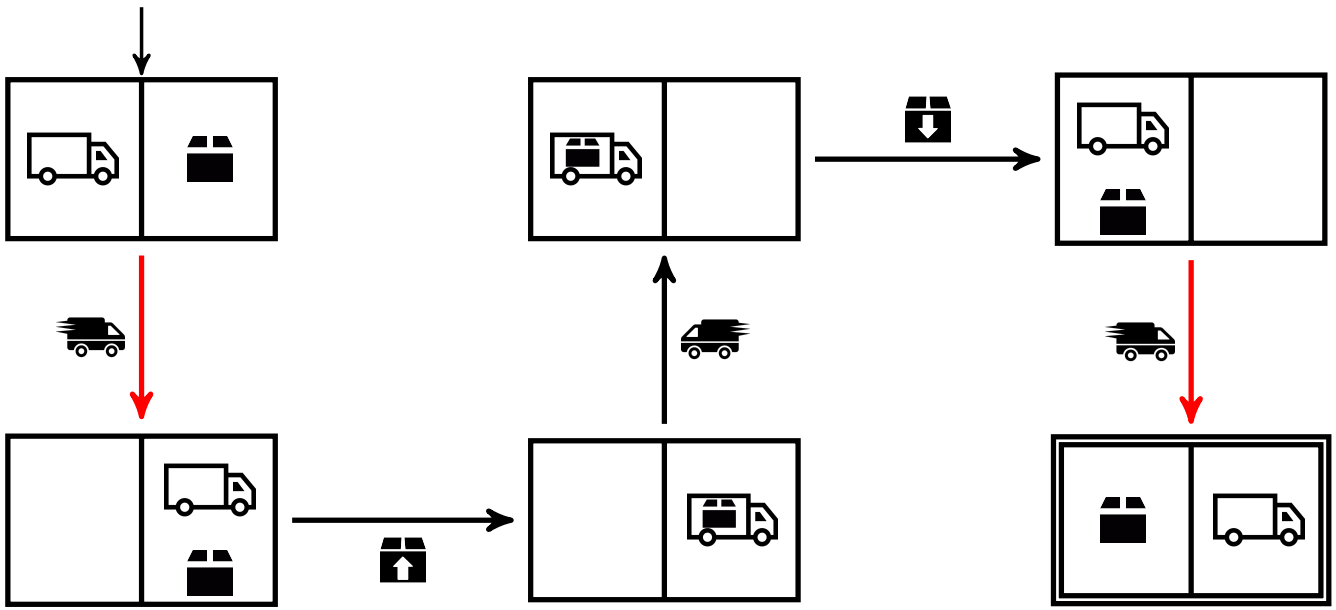
Requirement for admissibility: Assume $c(a) = 6$.

$c_1(a) = 3$
$c_2(a) = 2$
$\sum = 5$ ✔

$c_1(a) = 4$
$c_2(a) = 3$
$\sum = 7$ ✗


State-dependent
cost partitioning
(uncharted territory!)
State-independent
cost partitioning
(you are here!)

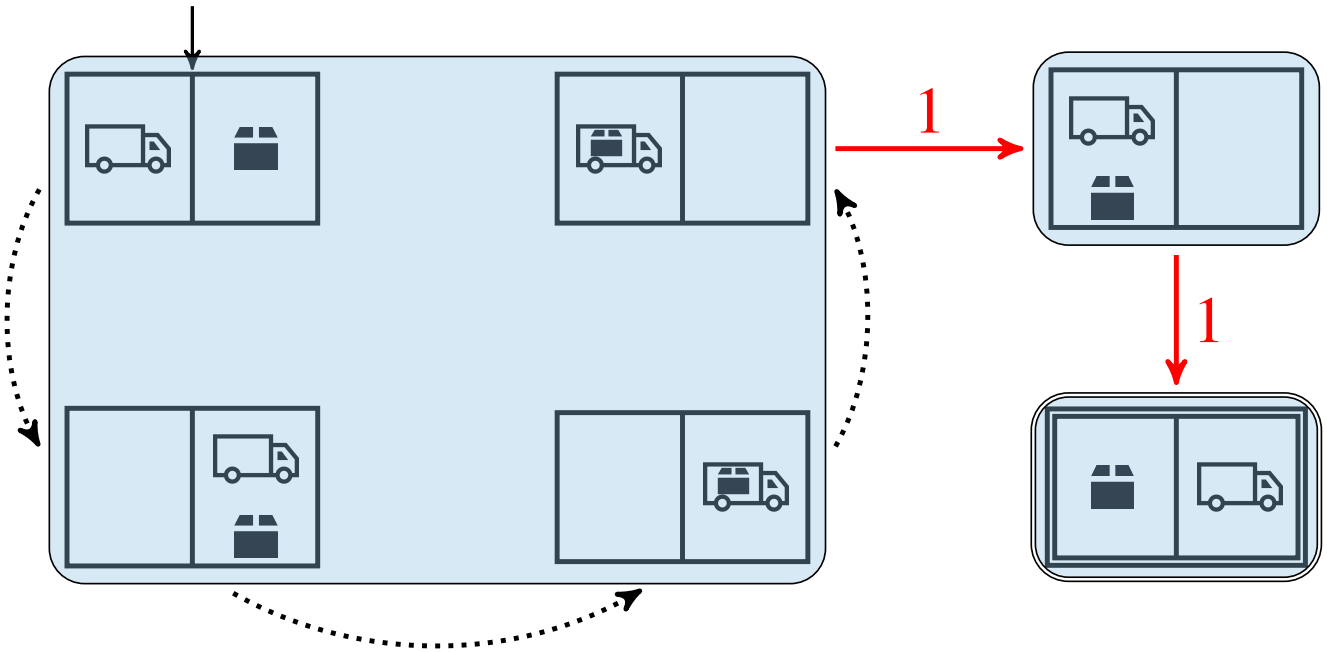
Running Example: Logistics Task



Cost of optimal plan: 5

Running Example: Logistics Task

Abstraction



Cost of abstract plan: 2

Cost Partitioning

A cost partitioning is a tuple $P = \langle c_1, \ldots, c_n \rangle$ where

- each $c_i : A \to \mathbb{R}$ is a cost function,
- sum of partial action costs bounded by original action cost

$$\sum_i c_i(a) \leq c(a) \qquad \text{for all } a \in A.$$

Theorem [Katz and Domshlak, 2010; Pommerening et al., 2015]

For admissible heuristics $h_1, \ldots, h_n$ and cost partitioning P, the cost partitioning heuristic $h^P(s) = \sum_i h_i(s, c_i)$ is admissible.

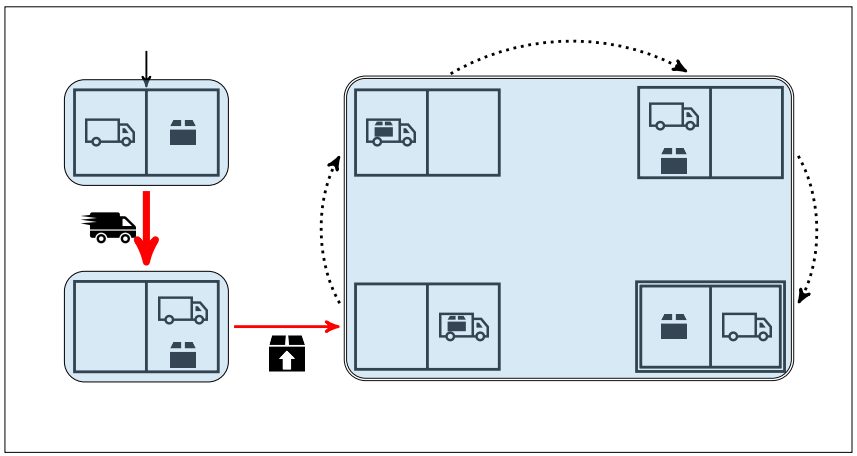
Optimal Cost Partitioning

An optimal cost partitioning for a state s is one which maximizes the heuristic estimate,

$$h^{ocp}(s) = \max_{P} \; h_P(s).$$

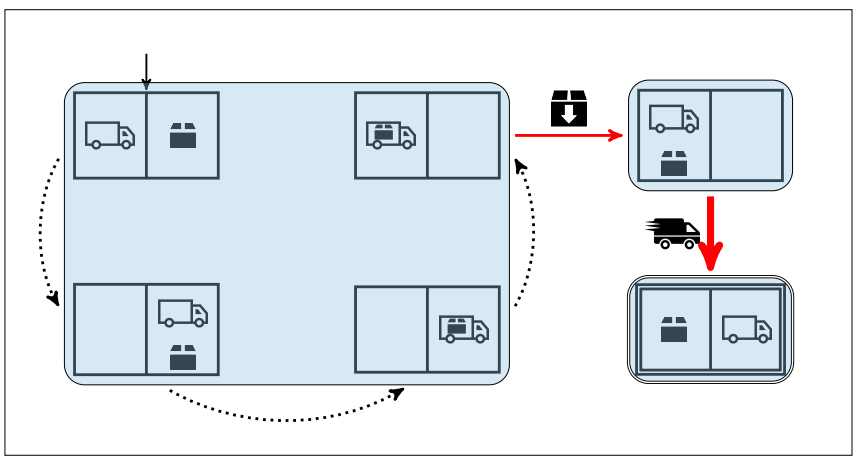
Running Example: Logistics Task

Optimal Cost Partitioning



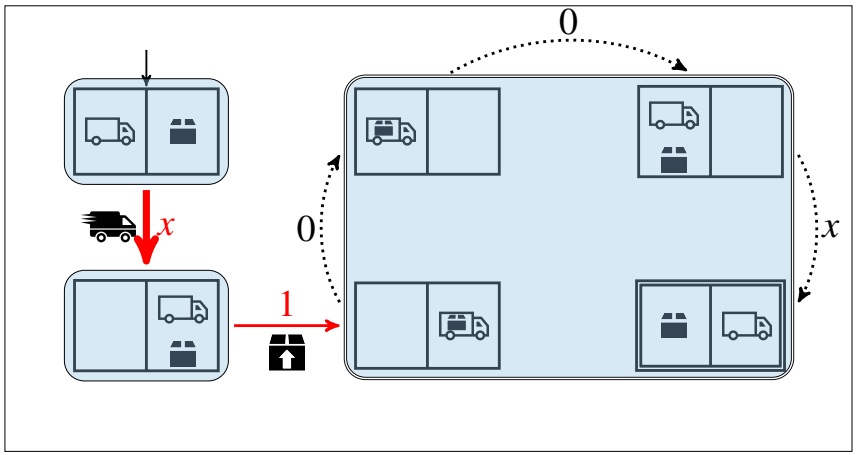
$$h_1(s_0) = c_1(\text{truck}) + c_1(\text{package})$$

$$h_2(s_0) = c_2(\text{package}) + c_2(\text{truck})$$



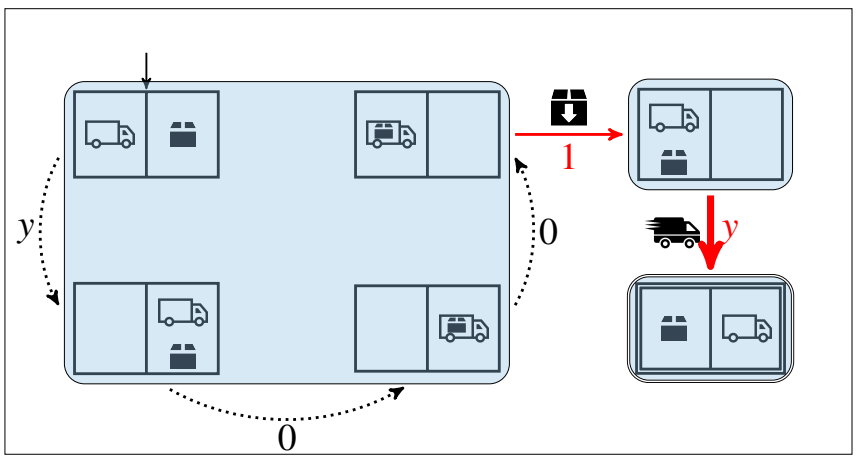
Running Example: Logistics Task

Optimal Cost Partitioning



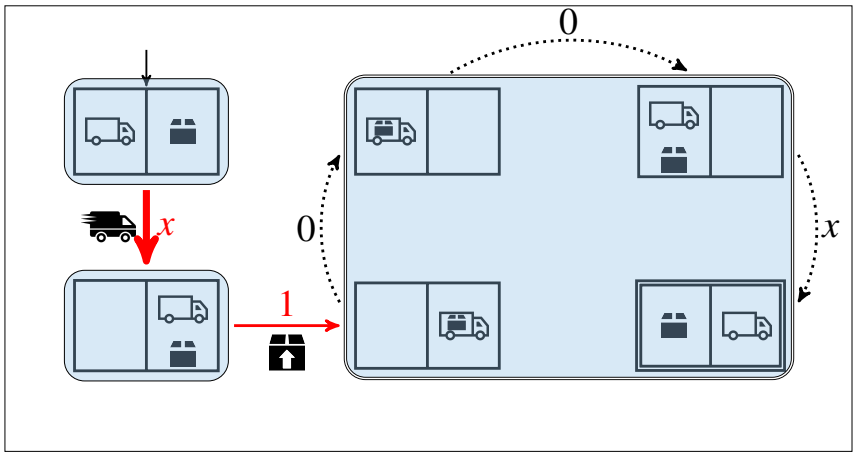
$$h_1(s_0) = c_1(\text{truck}) + c_1(\text{package})$$
$$h_2(s_0) = c_2(\text{package}) + c_2(\text{truck})$$

$$h^{ocp}(s_0) = 1 + c_1(\text{truck}) + c_2(\text{truck}) + 1$$
$$= 2 + x + y$$



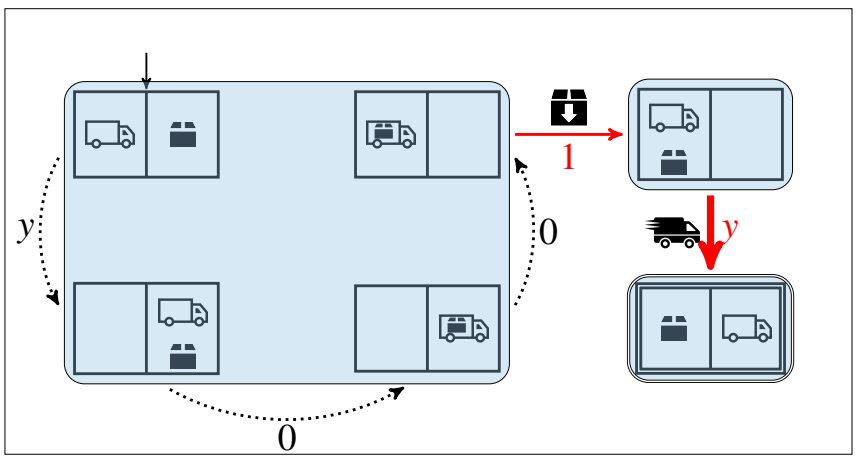
Running Example: Logistics Task

Optimal Cost Partitioning



$$h_1(s_0) = c_1(\text{truck}) + c_1(\text{package})$$
$$h_2(s_0) = c_2(\text{package}) + c_2(\text{truck})$$

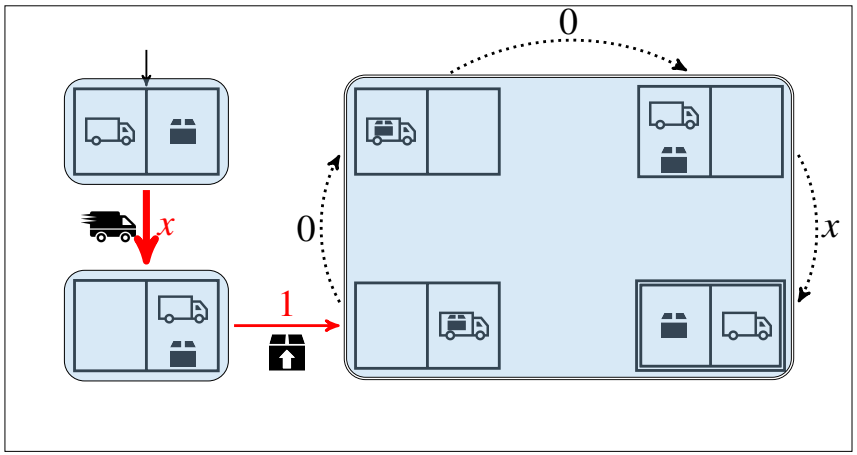
$$h^{ocp}(s_0) = 1 + c_1(\text{truck}) + c_2(\text{truck}) + 1$$
$$= 2 + x + y$$



maximize this subject to $x + y \leq 1$

Running Example: Logistics Task

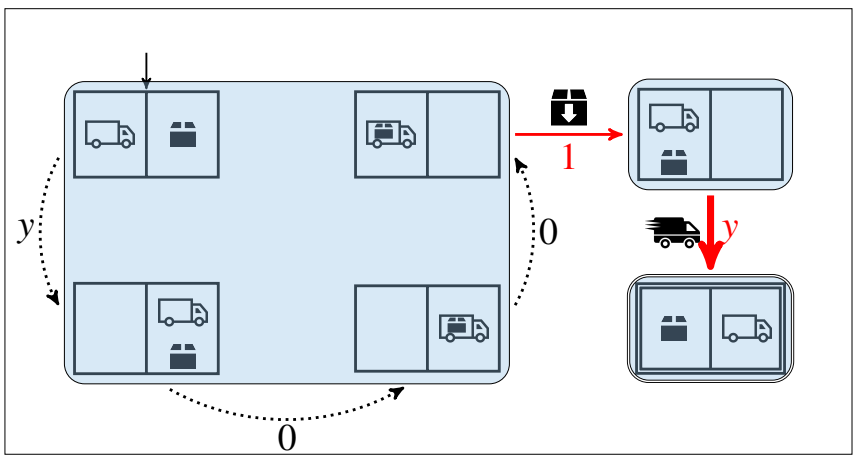
Optimal Cost Partitioning



$$h_1(s_0) = c_1(\text{truck}) + c_1(\text{package})$$

$$h_2(s_0) = c_2(\text{package}) + c_2(\text{truck})$$

$$h^{ocp}(s_0) = 1 + c_1(\text{truck}) + c_2(\text{truck}) + 1$$
$$= 2 + x + y$$

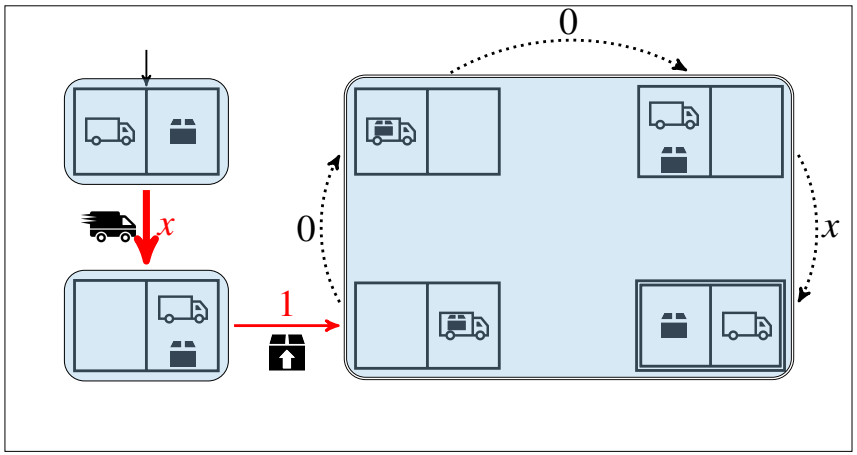


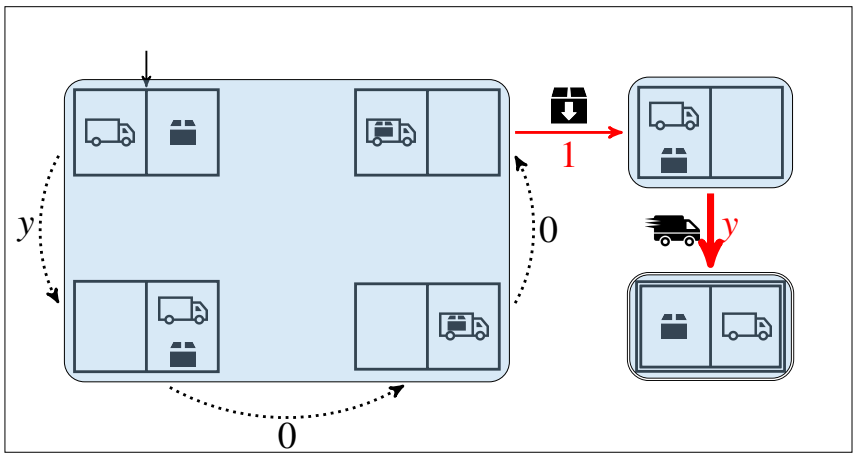
maximize this subject to $x + y \leq 1$

$$\Rightarrow \quad h^{ocp}(s_0) = 3$$

Running Example: Logistics Task

Optimal Cost Partitioning



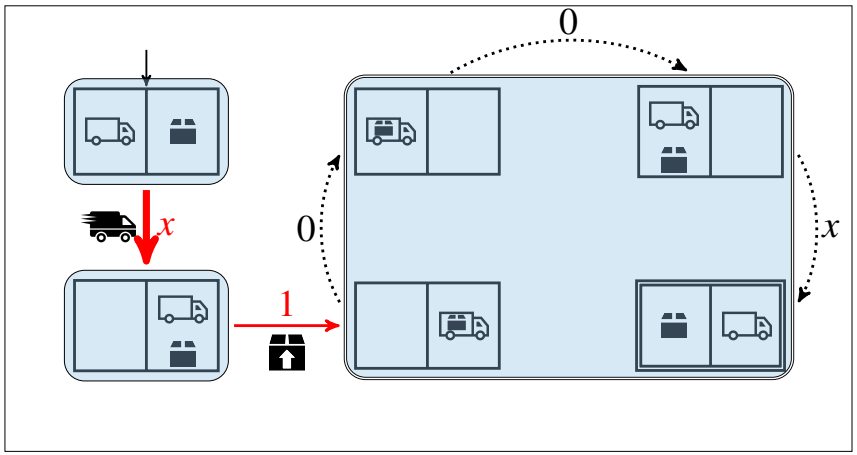


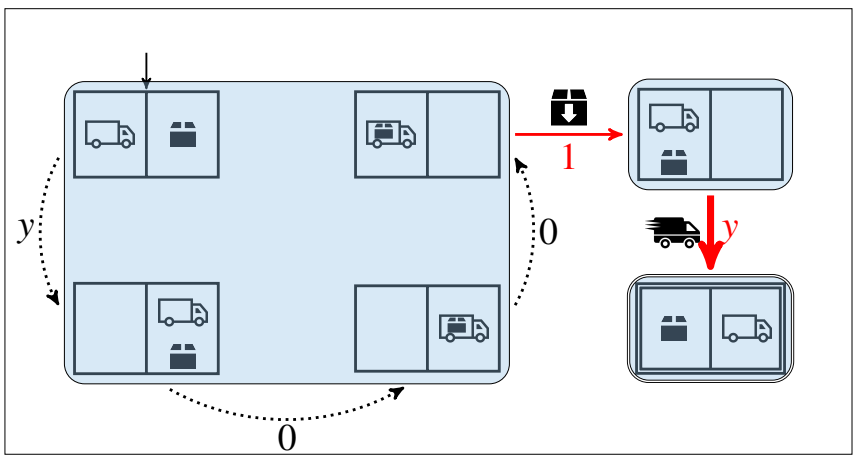
Problem: Action

- counted only once,
- but could be counted twice.

Running Example: Logistics Task

Optimal Cost Partitioning





Problem: Action

- counted only once,
- but could be counted twice.

Idea:

- Distinguish states in which action is applied.
  (= Assign partial costs state-wise.)
- If different, allow counting it both times.

State-Dependent Cost Partitioning

A state-dependent cost partitioning is a tuple $P = \langle c_1, \ldots, c_n \rangle$ where

- each $c_i : A \times S \to \mathbb{R}$ is a state-dependent cost function
- sum of partial action costs bounded by original action cost

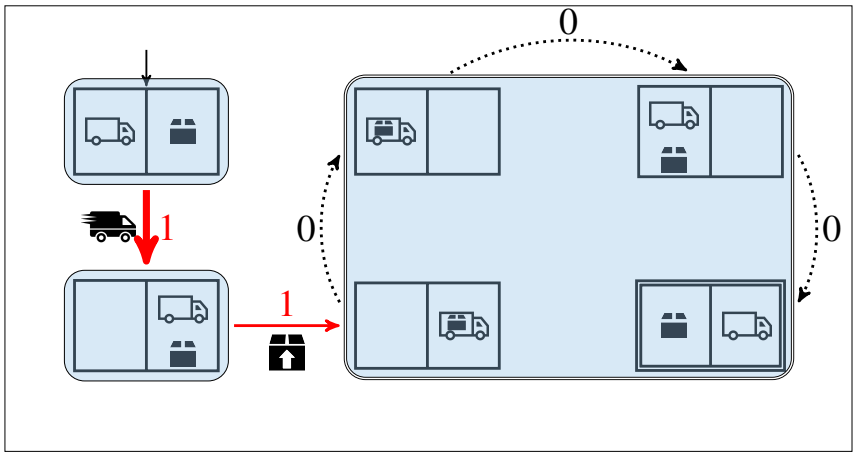
$$\sum_i c_i(a, s) \leq c(a) \qquad \text{for all } a \in A, s \in S.$$

Theorem

For admissible heuristics $h_1, \ldots, h_n$ and state-dependent cost partitioning P, the cost partitioning heuristic $h_P(s) = \sum_i h_i(s, c_i)$ is admissible.

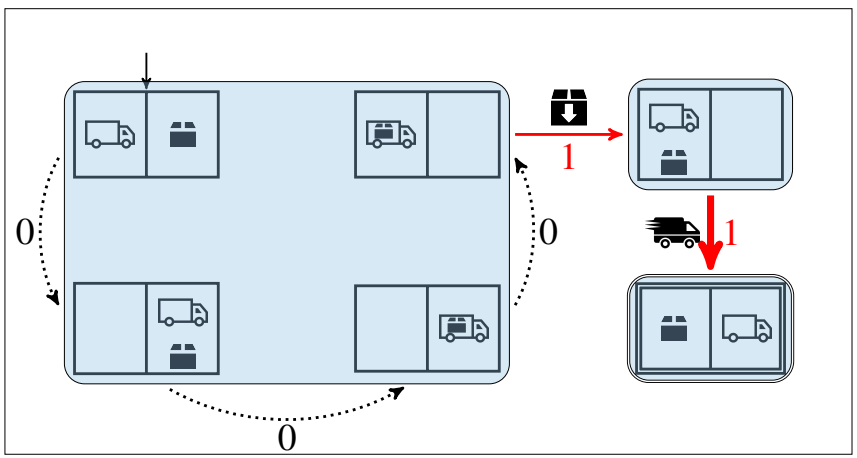
Running Example: Logistics Task

Optimal State-Dependent Cost Partitioning



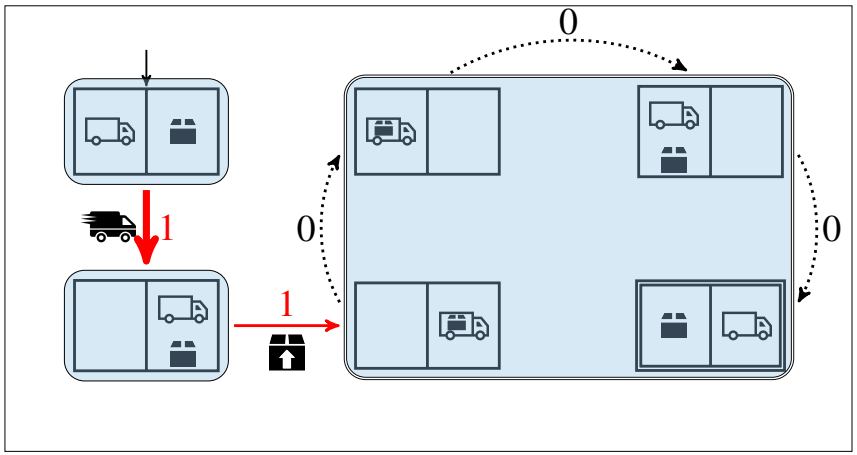
$$c_1(\text{truck}, s_0) = 1$$
$$c_1(\text{truck}, s') = 0 \quad (s' \neq s_0)$$

$$c_2(\text{truck}, s) = 1 - c_1(\text{truck}, s)$$
for all s



Running Example: Logistics Task

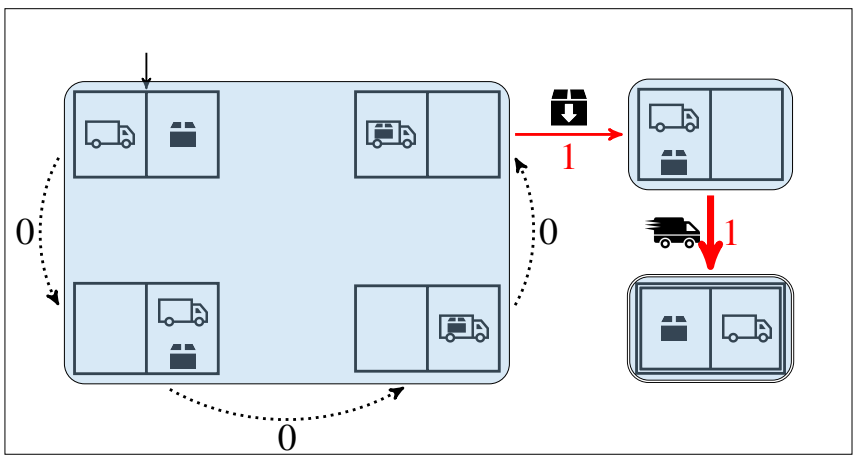
Optimal State-Dependent Cost Partitioning



$$c_1(\text{truck}, s_0) = 1$$
$$c_1(\text{truck}, s') = 0 \quad (s' \neq s_0)$$

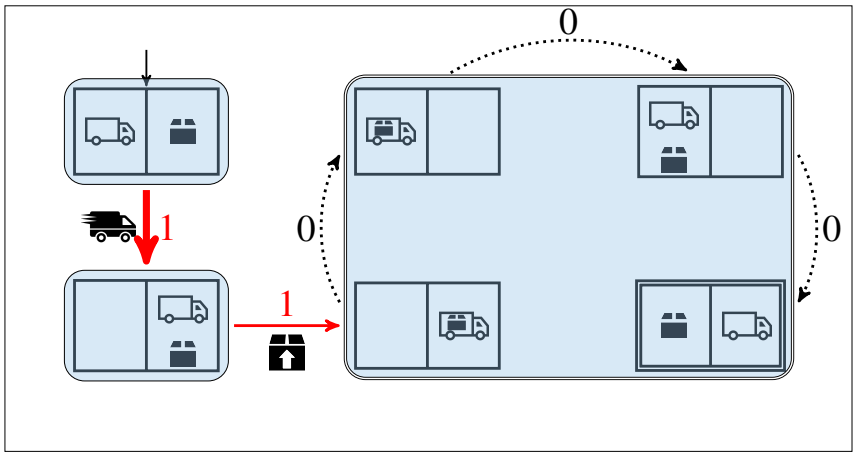
$$c_2(\text{truck}, s) = 1 - c_1(\text{truck}, s)$$
for all s

$$\Rightarrow \quad h^{\text{ocp-dep}}(s_0) = 1 + 1 + 1 + 1 = 4$$



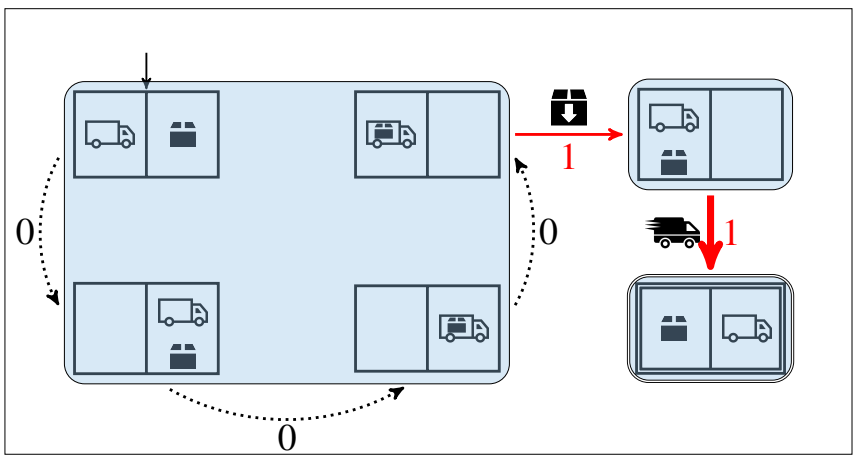
Running Example: Logistics Task

Optimal State-Dependent Cost Partitioning



$$c_1(\text{truck}, s_0) = 1$$
$$c_1(\text{truck}, s') = 0 \quad (s' \neq s_0)$$

$$c_2(\text{truck}, s) = 1 - c_1(\text{truck}, s)$$
$$\text{for all } s$$



$$\Rightarrow \quad h^{\text{ocp-dep}}(s_0) = 1 + 1 + 1 + 1 = 4$$
$$> 3 = h^{\text{ocp}}(s_0).$$

Saturated Cost Partitioning

State-Independent or State-Dependent

Problem: Optimal state-dependent cost partitioning too expensive to compute (exponential).

Remedy: Consider saturated cost-partitioning as an alternative.

Idea: In current abstraction, leave as much remaining costs for subsequent abstractions as possible without making current abstraction less informative.

Details: See IJCAI 2016 paper [KPSGM16].

Theoretical Results

Dominance and Incomparability

OPT-DEP

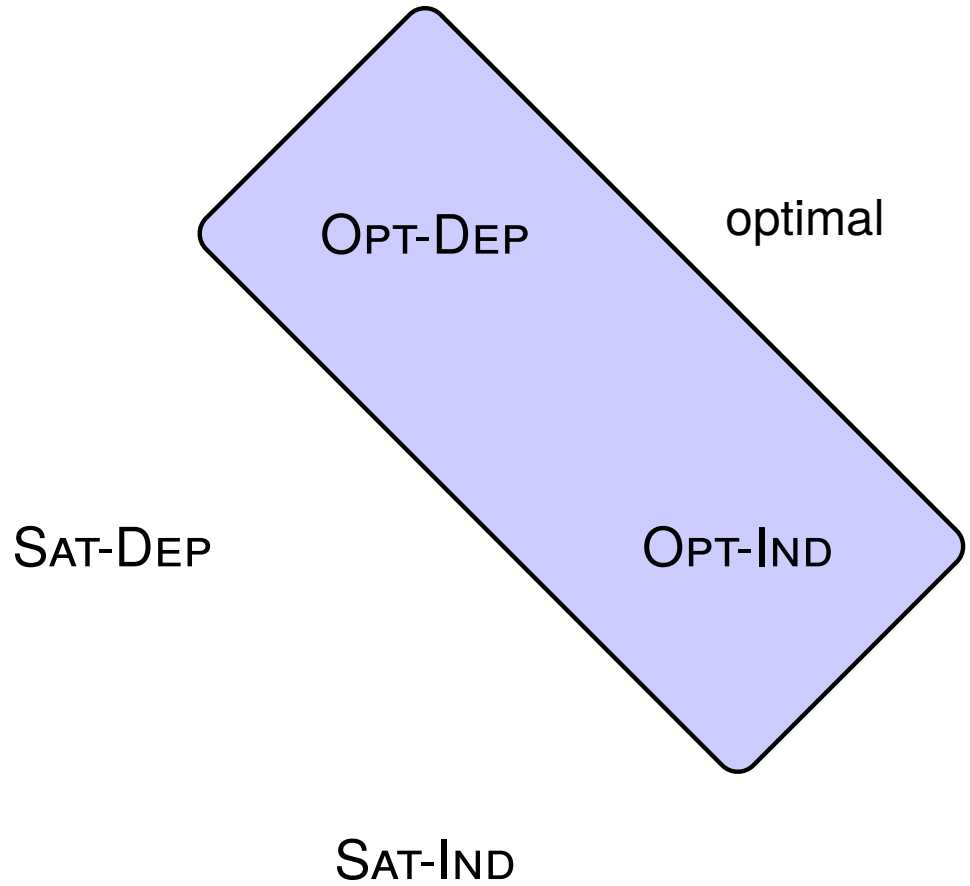
SAT-DEP

OPT-IND

SAT-IND

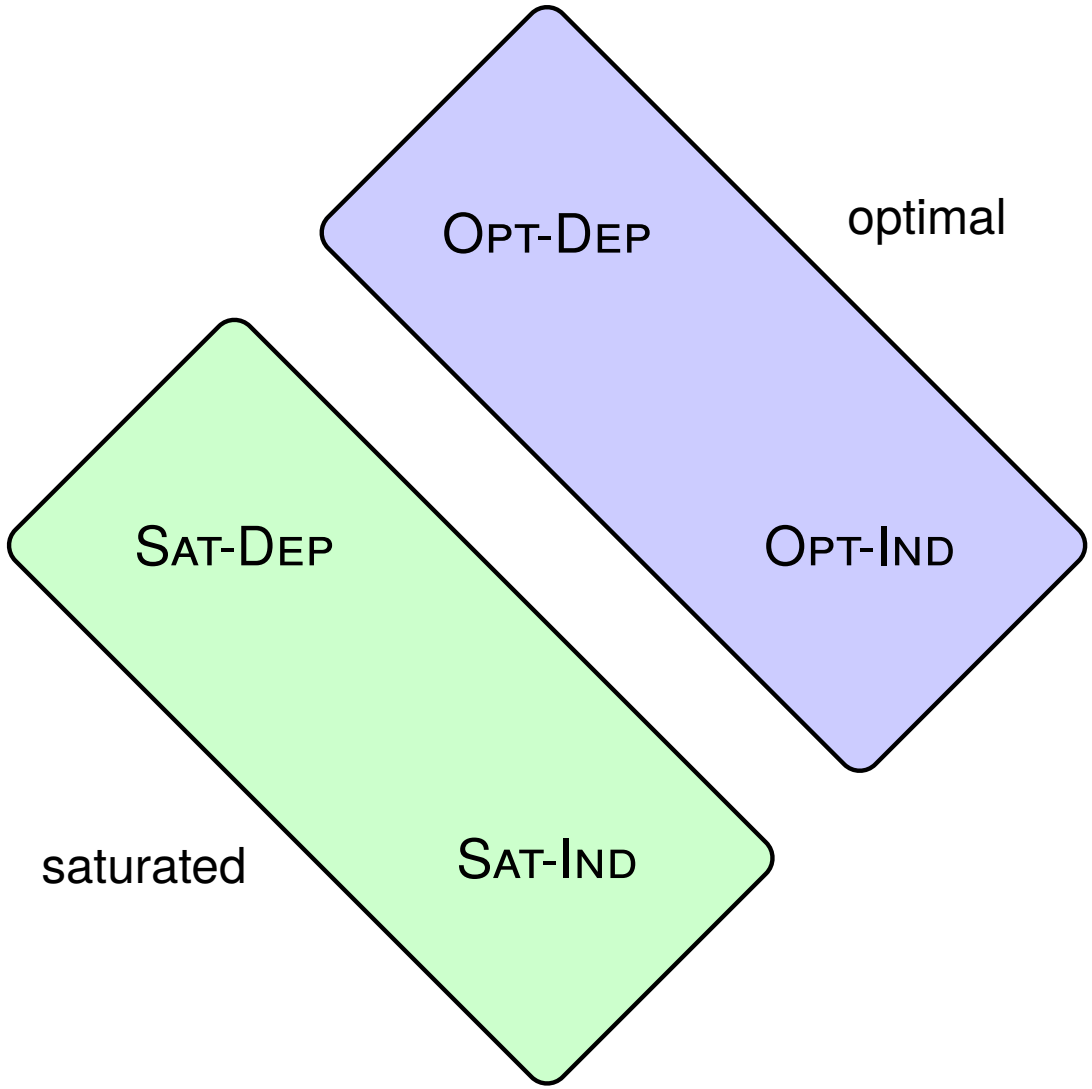
Theoretical Results

Dominance and Incomparability



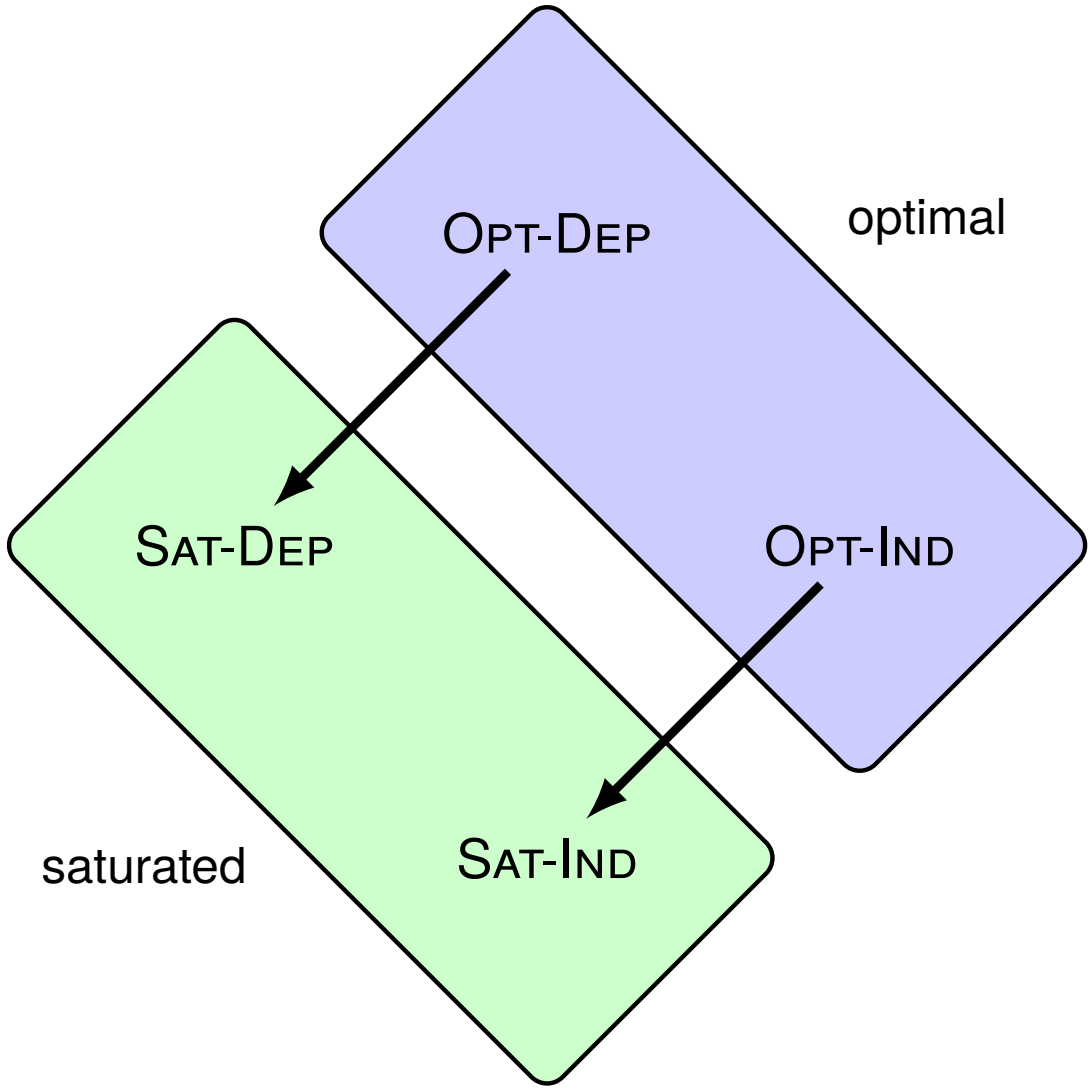
Theoretical Results

Dominance and Incomparability



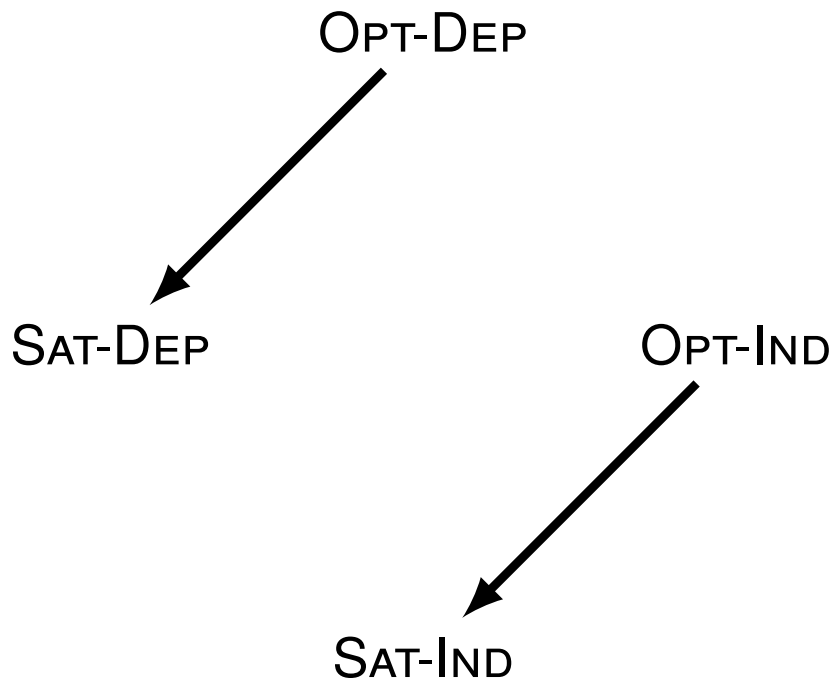
Theoretical Results

Dominance and Incomparability



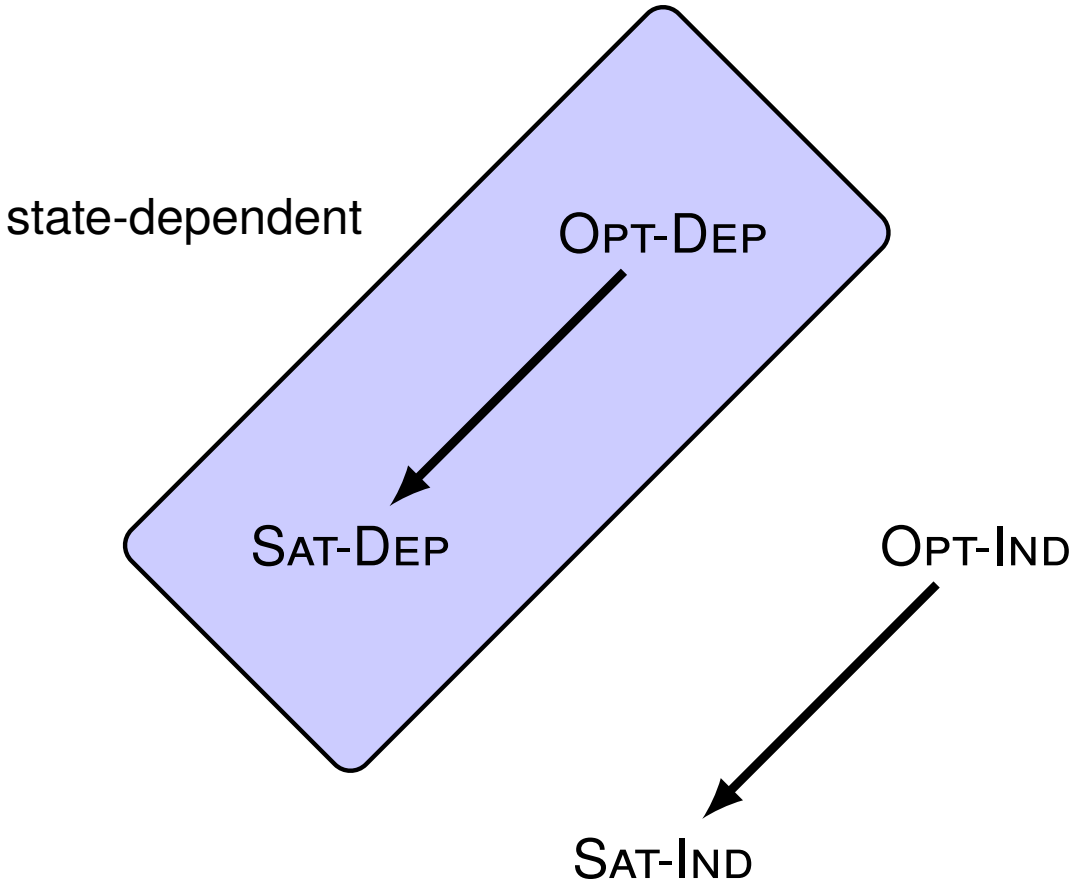
Theoretical Results

Dominance and Incomparability



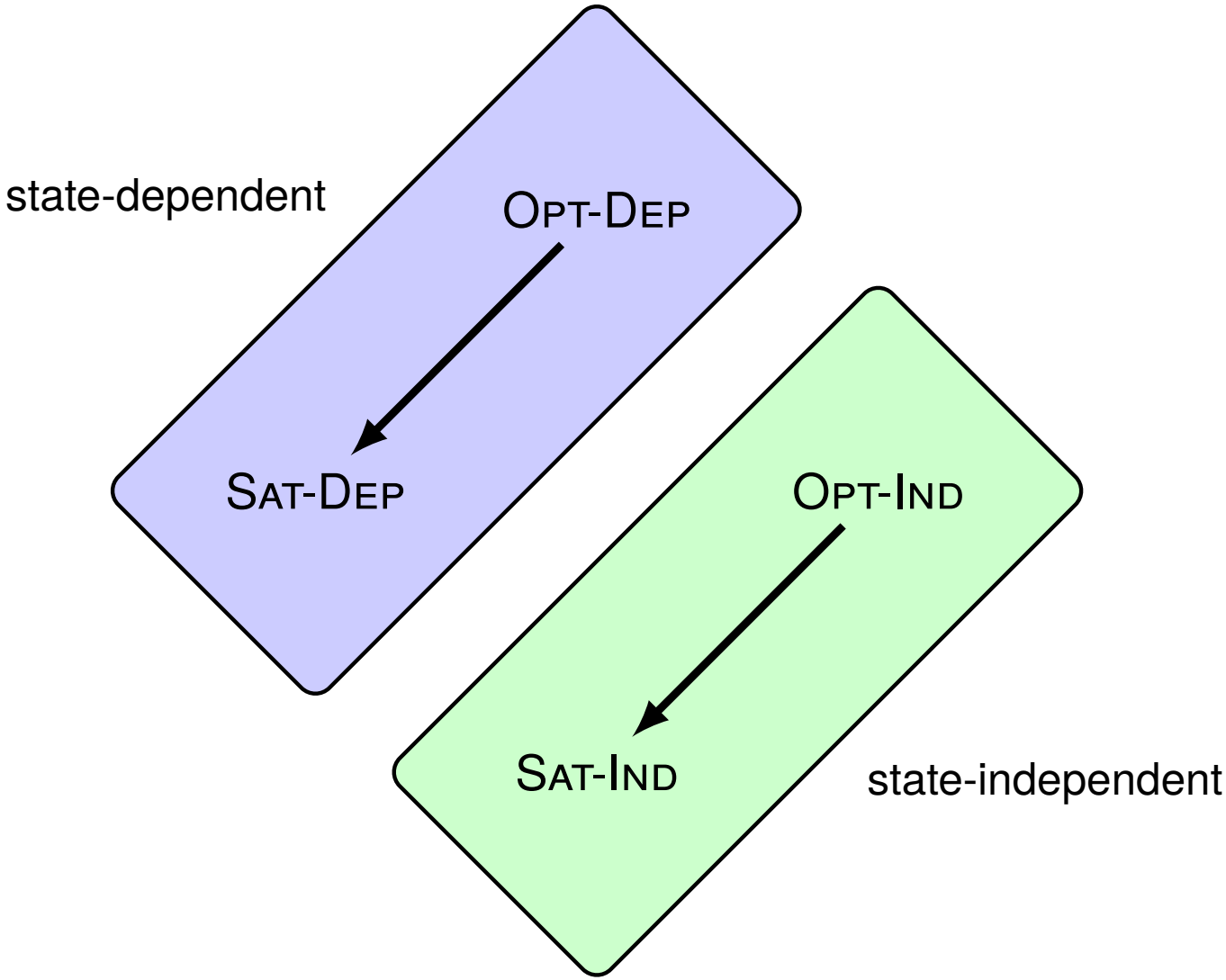
Theoretical Results

Dominance and Incomparability



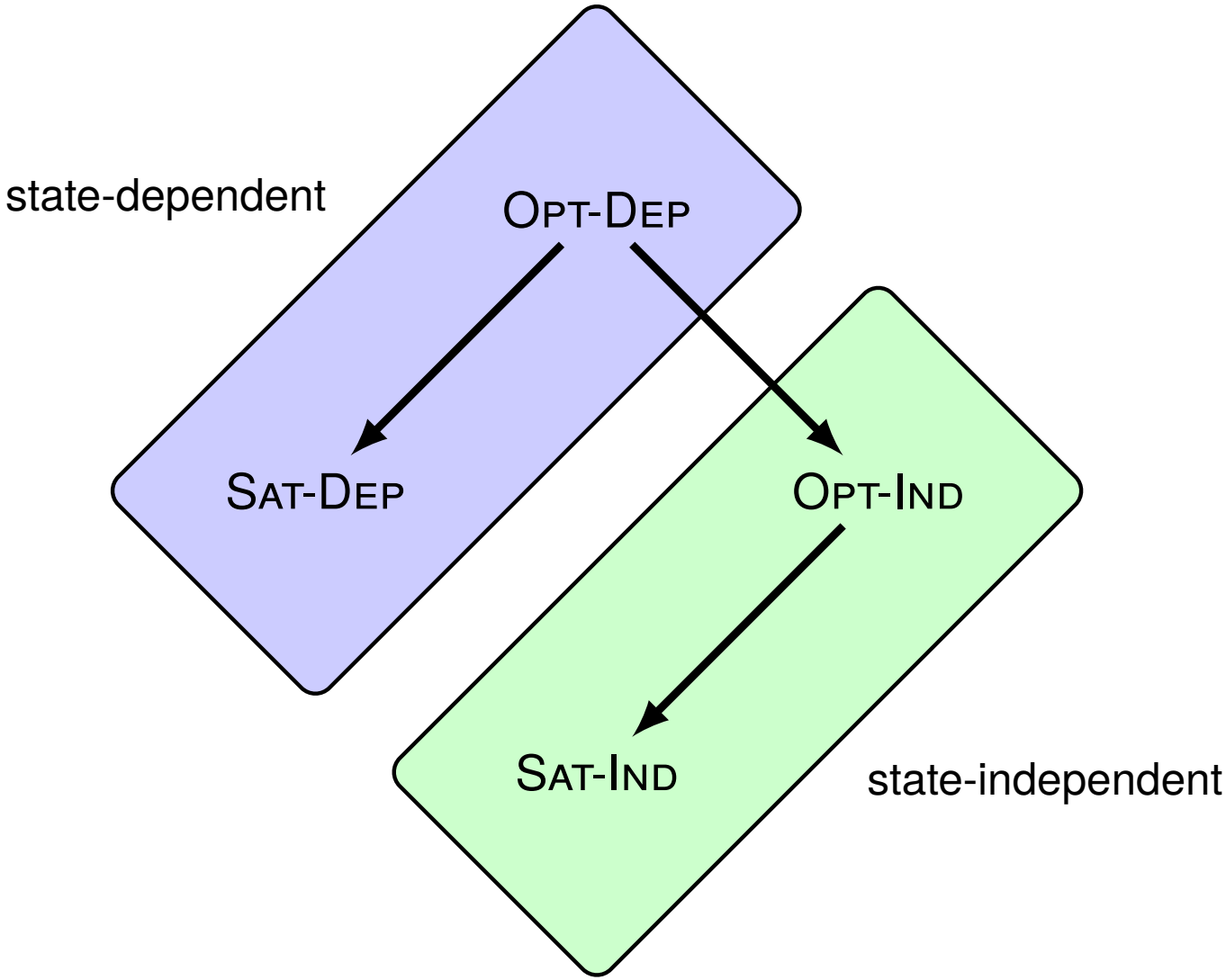
Theoretical Results

Dominance and Incomparability



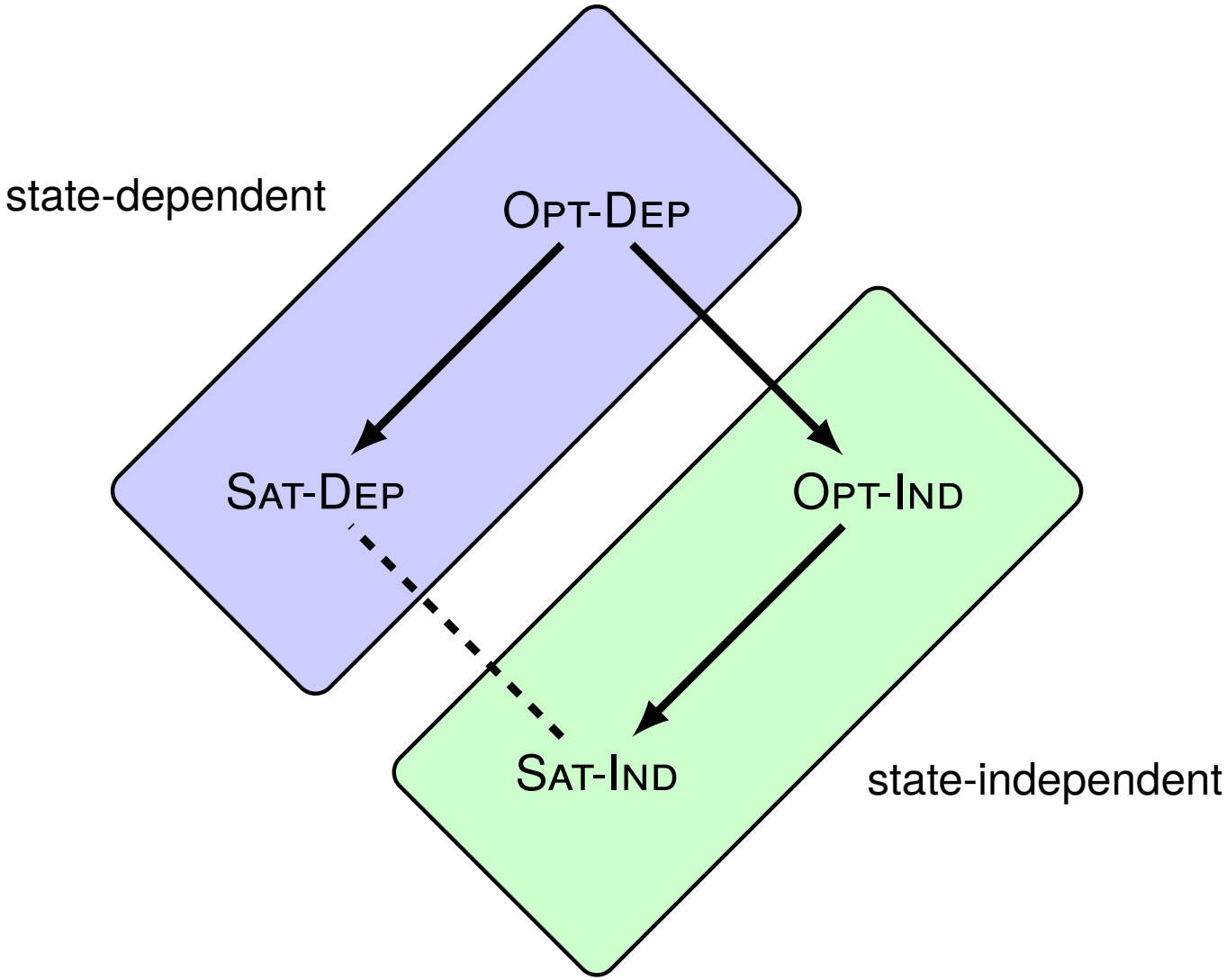
Theoretical Results

Dominance and Incomparability



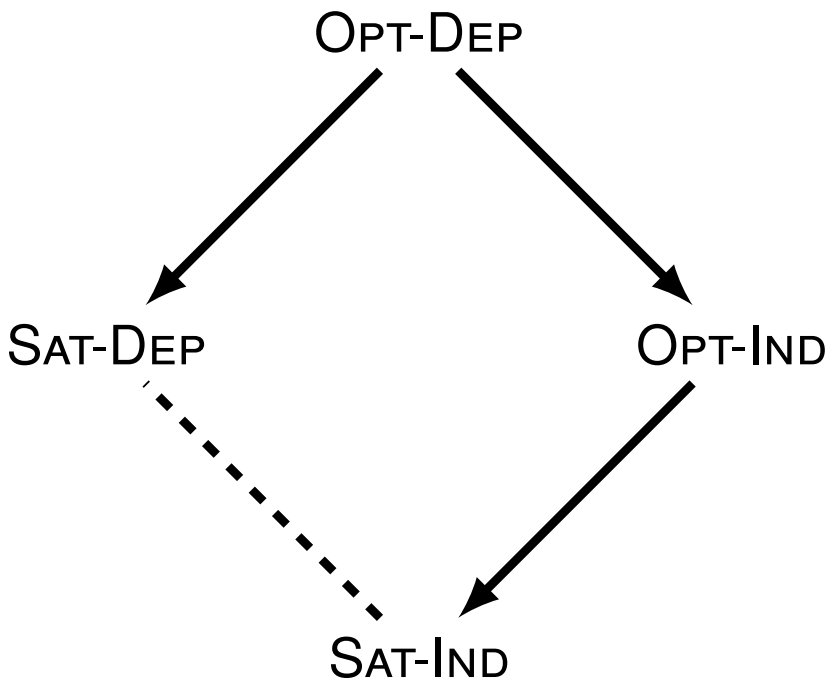
Theoretical Results

Dominance and Incomparability



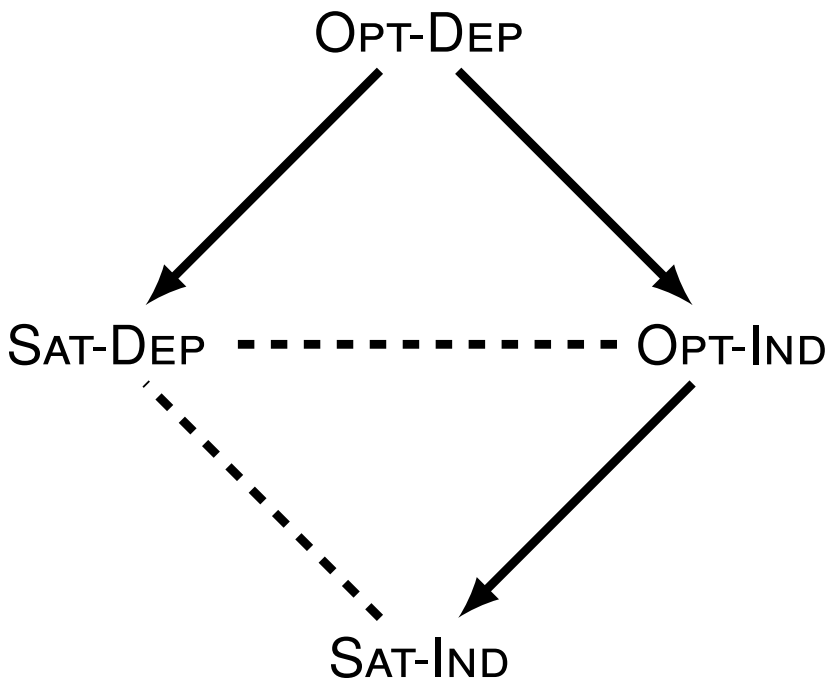
Theoretical Results

Dominance and Incomparability



Theoretical Results

Dominance and Incomparability



Conclusion

- State-dependent more fine-grained than state-independent cost partitioning.
- Complete classification of dominance among $\{\textsc{Opt}, \textsc{Sat}\} \times \{\textsc{Dep}, \textsc{Ind}\}$.
- Only preliminary empirical results.

Appendix

Empirical Results

Preliminary but promising.

- Opt-Dep really too expensive.
- Though theoretically incomparable,
  Sat-Dep often more accurate than Sat-Ind.
- Conjecture:
  Sat-Dep and Opt-Ind have potential to complement each other.