Optimal Search with Inadmissible Heuristics

Erez Karpas   Carmel Domshlak

Faculty of Industrial Engineering and Management,
Technion — Israel Institute of Technology

May 13, 2012

Outline

Search Problems

Almost every problem in AI can be seen as a search problem

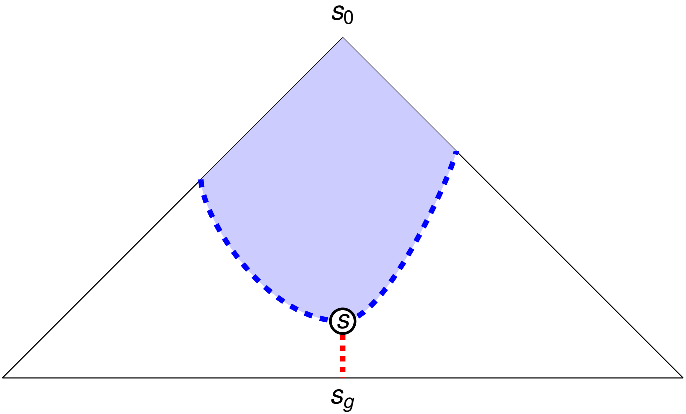
- A search problem contains:
  - Initial world state
  - Set of goal states
  - Set of deterministic actions
- A solution is a sequence of actions:
  - Transforms the initial world state into a goal state
- We are interested in optimal search:
  - Find (one of) the cheapest possible solutions

Heuristic Forward Search

- Heuristic forward search:
  1. Maintains a list of candidate states (open list)
  2. At each iteration, a state is removed from the list
  3. If it is not a goal state, all of its successors are added to the list
- The choice of which state to remove usually involves a heuristic evaluation function
  - Evaluates the merit of each state

Outline

Admissibility of Heuristics



Admissible

A heuristic is admissible iff $h(s) \leq h^*(s)$ for any state s.

Optimality and Admissibility

- We know that A^* search with an admissible heuristic guarantees an optimal solution
- Is this a necessary condition?

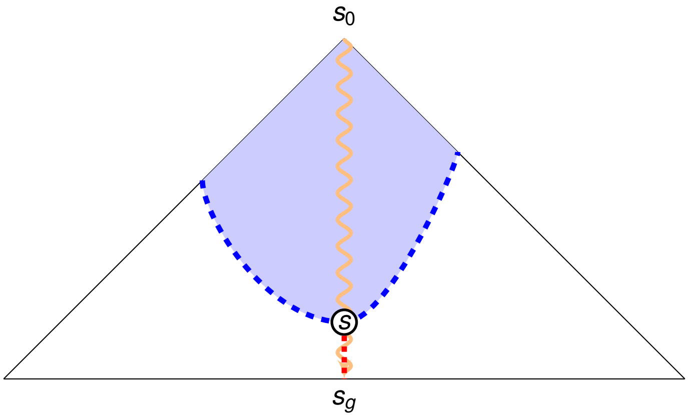
Optimality and Admissibility

- We know that A^* search with an admissible heuristic guarantees an optimal solution
- Is this a necessary condition?

Optimality and Admissibility

- We know that A^* search with an admissible heuristic guarantees an optimal solution
- Is this a necessary condition? No

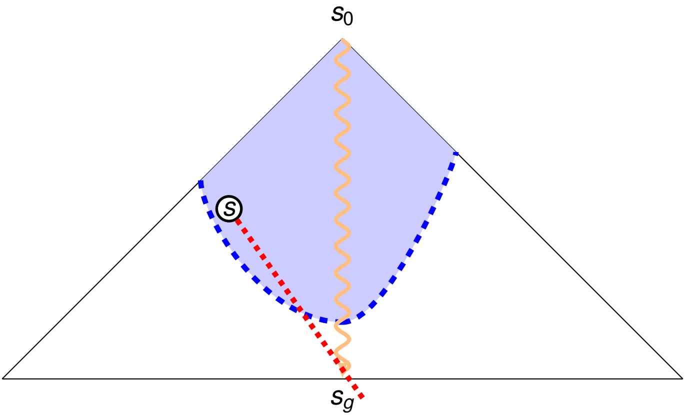
Global Admissibility



Globally Admissible

A heuristic is globally admissible iff there exists some optimal solution ρ such that for any state s along ρ: $h(s) \leq h^*(s)$

Global Admissibility



Globally Admissible

A heuristic is globally admissible iff there exists some optimal solution ρ such that for any state s along ρ: $h(s) \leq h^*(s)$

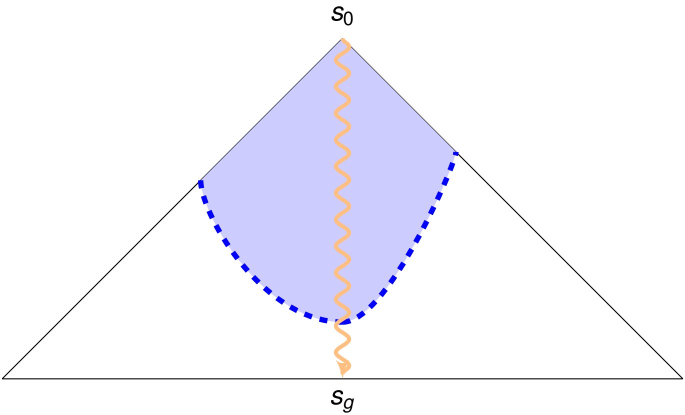
Global Admissibility

- As noted by Dechter & Pearl (1985), using A^* with a globally admissible heuristic guarantees finding an optimal solution
- Examples of globally admissible heuristics
  - Symmetry-based pruning (Pochter et al, 2011; Coles & Smith 2008; Rintanen 2003; Fox & Long, 2002)
  - Partial order reduction (Chen & Yao, 2009; Haslum, 2000)
- Can be seen as assigning ∞ to pruned states
- But heuristic estimates can be path-dependent

Global Admissibility

- As noted by Dechter & Pearl (1985), using A^* with a globally admissible heuristic guarantees finding an optimal solution
- Examples of globally admissible heuristics
  - Symmetry-based pruning (Pochter et al, 2011; Coles & Smith 2008; Rintanen 2003; Fox & Long, 2002)
  - Partial order reduction (Chen & Yao, 2009; Haslum, 2000)
- Can be seen as assigning ∞ to pruned states
- But heuristic estimates can be path-dependent

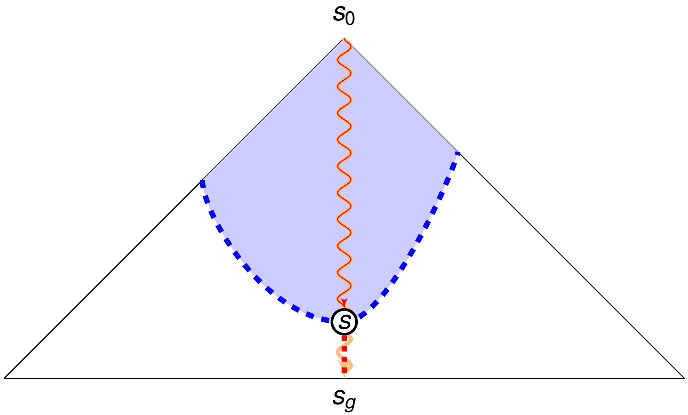
Path Dependent Admissibility



$\{\rho\}$-Admissible

A heuristic is $\{\rho\}$-admissible iff ρ is an optimal solution, and for any prefix π of ρ leading to state s: $h(\pi) \leq h^*(s)$

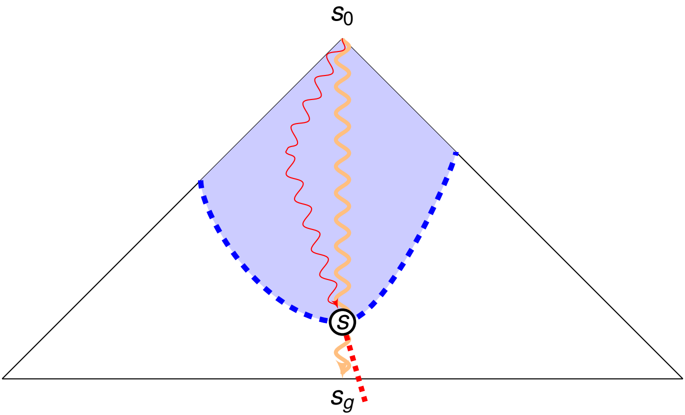
Path Dependent Admissibility



$\{\rho\}$-Admissible

A heuristic is $\{\rho\}$-admissible iff ρ is an optimal solution, and for any prefix π of ρ leading to state s: $h(\pi) \leq h^*(s)$

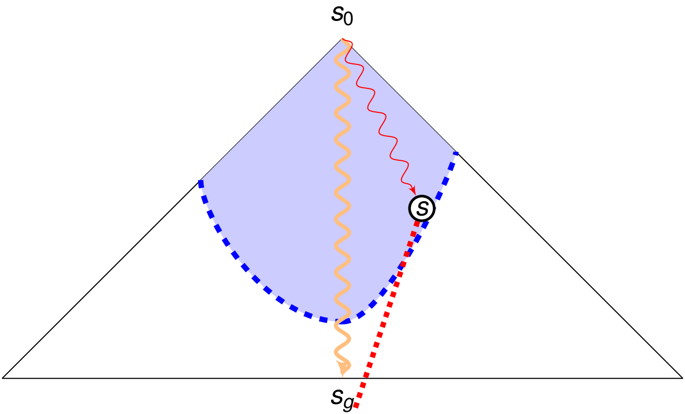
Path Dependent Admissibility



$\{\rho\}$-Admissible

A heuristic is $\{\rho\}$-admissible iff ρ is an optimal solution, and for any prefix π of ρ leading to state s: $h(\pi) \leq h^*(s)$

Path Dependent Admissibility



$\{\rho\}$-Admissible

A heuristic is $\{\rho\}$-admissible iff ρ is an optimal solution, and for any prefix π of ρ leading to state s: $h(\pi) \leq h^*(s)$

Path-admissible Heuristics

- Can be generalized to χ-admissibility for a set of solutions χ
- If χ is the set of all optimal solutions, we call h path admissible
- If χ contains at least one optimal solutions, we call h globally path admissible

Search with Path-admissible Heuristics

- Using a (globally) path admissible heuristic with A^* does not guarantee an optimal solution will be found
- However, tree based search algorithms can guarantee an optimal solution is found with a (globally) path admissible heuristic
- It is also possible to do some duplicate detection — details later

Outline

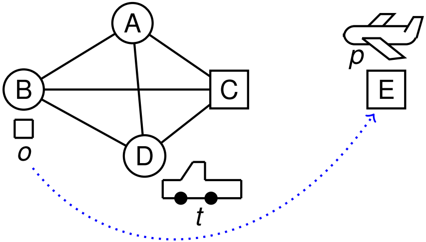
STRIPS

- A STRIPS planning problem with action costs is a 5-tuple $\Pi = \langle P, s_0, G, A, C \rangle$
  - P is a set of boolean propositions
  - $s_0 \subseteq P$ is the initial state
  - $G \subseteq P$ is the goal
  - A is a set of actions.
  - Each action is a triple $a = \langle \text{pre}(a), \text{add}(a), \text{del}(a) \rangle$
  - $C : A \rightarrow \mathbb{R}^{0+}$ assigns a cost to each action
- Applying action sequence $\rho = \langle a_0, a_1, \ldots, a_n \rangle$ at state s leads to $s[[\rho]]$
- The cost of action sequence ρ is $\sum_{i=0}^{n} C(a_i)$

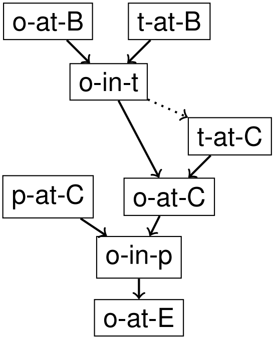
Landmarks

- A landmark is a formula that must be true at some point in every plan (Hoffmann, Porteous & Sebastia 2004)
- Landmarks can be (partially) ordered according to the order in which they must be achieved
- Some landmarks and orderings can be discovered automatically

Example Planning Problem - Logistics



(Slide due to Silvia Richter)



Partial landmarks graph

Admissible Landmark-based Heuristics

- Any landmarks that were not achieved yet, must be achieved later (note: path-dependent)
- Can use action cost-partitioning to get an admissible estimate (Karpas & Domshlak, 2009)
- Idea: the cost of a set of landmarks is no greater than the cost of any single action that achieves them
- Given that, the sum of costs of landmarks that still need to be achieved is an admissible heuristic

Outline

Intended Effects

Chicken logic

Why did the chicken cross the road?

Intended Effects

Chicken logic

Why did the chicken cross the road?

To get to the other side

Intended Effects

Chicken logic

Why did the chicken cross the road?

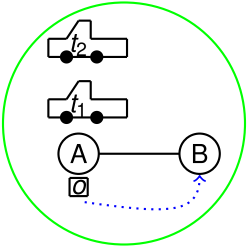
To get to the other side

Observation

Every along action an optimal plan is there for a reason

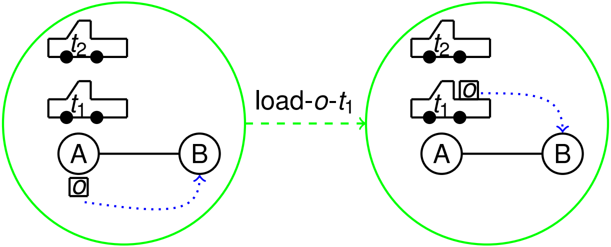
- Achieve a precondition for another action
- Achieve a goal

Intended Effects — Example



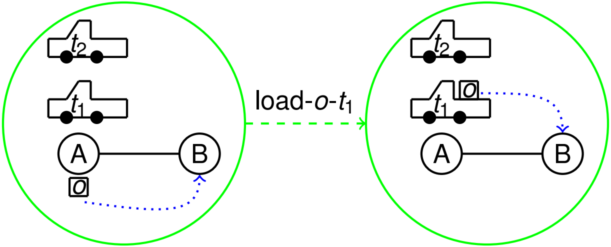
- There must be a reason for applying load-o-t_1
- load-o-t_1 achieves o-in-t_1
- Any continuation of this path to an optimal plan must use some action which requires o-in-t_1

Intended Effects — Example



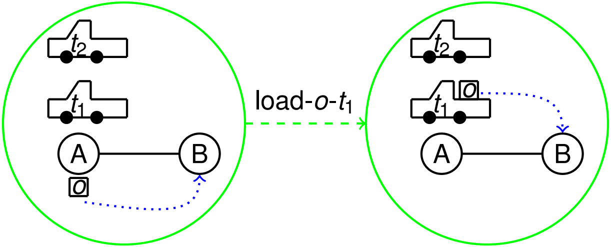
- There must be a reason for applying load-o-t_1
- load-o-t_1 achieves o-in-t_1
- Any continuation of this path to an optimal plan must use some action which requires o-in-t_1

Intended Effects — Example



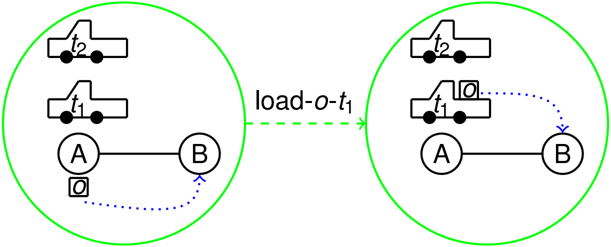
- There must be a reason for applying load-o-t_1
- load-o-t_1 achieves o-in-t_1
- Any continuation of this path to an optimal plan must use some action which requires o-in-t_1

Intended Effects — Example



- There must be a reason for applying load-o-t_1
- load-o-t_1 achieves o-in-t_1
- Any continuation of this path to an optimal plan must use some action which requires o-in-t_1

Intended Effects — Example



- There must be a reason for applying load-o-t_1
- load-o-t_1 achieves o-in-t_1
- Any continuation of this path to an optimal plan must use some action which requires o-in-t_1

Intended Effects — Intuition

- We formalize chicken logic using the notion of Intended Effects
- A set of propositions $X \subseteq s_0 [[\pi]]$ is an intended effect of path π, if we can use X to continue π into an optimal plan
- Using X refers to the presence of causal links in the optimal plan

Causal Link

Let $\pi = \langle a_0, a_1, \ldots a_n \rangle$ be some path. The triple $\langle a_i, p, a_j \rangle$ forms a *causal link* in π if a_i is the actual provider of precondition p for a_j.

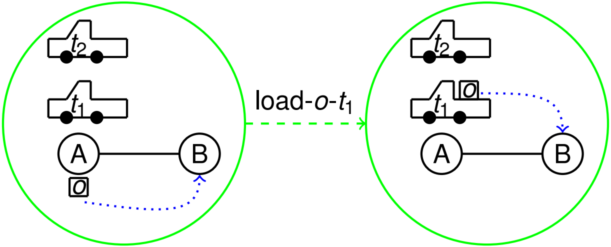
Intended Effects — Formal Definition

Intended Effects

Let OPT be a set of optimal plans for planning task Π. Given a path $\pi = \langle a_0, a_1, \ldots a_n \rangle$ a set of propositions $X \subseteq s_0[[\pi]]$ is an OPT-intended effect of π iff there exists a path π' such that $\pi \cdot \pi' \in \text{OPT}$ and π' consumes exactly X ($p \in X$ iff there is a causal link $\langle a_i, p, a_j \rangle$ in $\pi \cdot \pi'$, with $a_i \in \pi$ and $a_j \in \pi'$).

- $\text{IE}(\pi|\text{OPT})$ — the set of all OPT-intended effect of π
- $\text{IE}(\pi) = \text{IE}(\pi|\text{OPT})$ when OPT is the set of all optimal plans

Intended Effects — Set Example



The Intended Effects of $\pi = \langle \text{load-}o\text{-}t_1 \rangle$ are $\{\{o\text{-in-}t_1\}\}$

Intended Effects — It's Logical

- Working directly with the set of subsets $\mathrm{IE}(\pi|\mathrm{OPT})$ is difficult
- We can interpret $\mathrm{IE}(\pi|\mathrm{OPT})$ as a boolean formula ϕ

$$X \in \mathrm{IE}(\pi|\mathrm{OPT}) \iff X \models \phi$$

- We can also interpret any path π' from $s_0[[\pi]]$ as a boolean valuation over propositions P

$$p = \mathrm{TRUE} \iff \text{there is a causal link } \langle a_i, p, a_j \rangle \text{ with } a_i \in \pi \text{ and } a_j \in \pi'$$

- Thus we can check if path $\pi' \models \phi$

Intended Effects — It's Logical

- Working directly with the set of subsets $\mathrm{IE}(\pi|\mathrm{OPT})$ is difficult
- We can interpret $\mathrm{IE}(\pi|\mathrm{OPT})$ as a boolean formula ϕ

$$X \in \mathrm{IE}(\pi|\mathrm{OPT}) \iff X \models \phi$$

- We can also interpret any path π' from s_0 $[[\pi]]$ as a boolean valuation over propositions P

$$p = \mathrm{TRUE} \iff \text{there is a causal link } \langle a_i, p, a_j \rangle \text{ with } a_i \in \pi \text{ and } a_j \in \pi'$$

- Thus we can check if path $\pi' \models \phi$

Intended Effects — It's Logical

- Working directly with the set of subsets $\text{IE}(\pi|\text{OPT})$ is difficult
- We can interpret $\text{IE}(\pi|\text{OPT})$ as a boolean formula ϕ

$$X \in \text{IE}(\pi|\text{OPT}) \iff X \models \phi$$

- We can also interpret any path π' from s_0 $[[\pi]]$ as a boolean valuation over propositions P

$$p = \text{TRUE} \iff \text{there is a causal link } \langle a_i, p, a_j \rangle \text{ with } a_i \in \pi \text{ and } a_j \in \pi'$$

- Thus we can check if path $\pi' \models \phi$

Intended Effects — It's Logical

- Working directly with the set of subsets $\text{IE}(\pi|\text{OPT})$ is difficult
- We can interpret $\text{IE}(\pi|\text{OPT})$ as a boolean formula ϕ

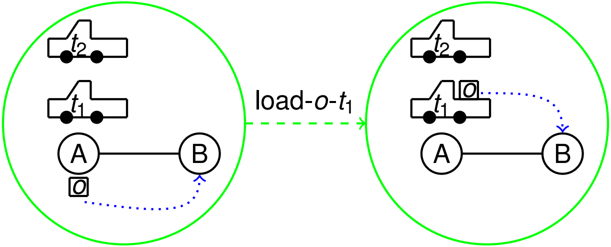
$$X \in \text{IE}(\pi|\text{OPT}) \iff X \models \phi$$

- We can also interpret any path π' from s_0 $[[\pi]]$ as a boolean valuation over propositions P

$$p = \text{TRUE} \iff \text{there is a causal link } \langle a_i, p, a_j \rangle \text{ with } a_i \in \pi \text{ and } a_j \in \pi'$$

- Thus we can check if path $\pi' \models \phi$

Intended Effects — Formula Example



The Intended Effects of $\pi = \langle$load-o-$t_1\rangle$ are described by the formula $\phi =$ o-in-t_1

Intended Effects — What Are They Good For?

We can use a logical formula describing $\mathrm{IE}(\pi|\textsc{opt})$ to derive constraints about what must happen in any continuation of π to a plan in $\textsc{opt}$.

Theorem 1

Let $\textsc{opt}$ be a set of optimal plans for a planning task Π, π be a path, and ϕ be a propositional logic formula describing $\mathrm{IE}(\pi|\textsc{opt})$. Then, for any $s_0[[\pi]]$-plan π', $\pi \cdot \pi' \in \textsc{opt}$ implies $\pi' \models \phi$.

Intended Effects — The Bad News

It's P-SPACE Hard to find the intended effects of path π.

Theorem 2

Let INTENDED be the following decision problem: Given a planning task Π, a path π, and a set of propositions $X \subseteq P$, is $X \in \text{IE}(\pi)$? Deciding INTENDED is P-SPACE Complete.

Approximate Intended Effects — The Good News

We can use supersets of $\mathrm{IE}(\pi|\textsc{opt})$ to derive constraints about any continuation of π.

Theorem 3

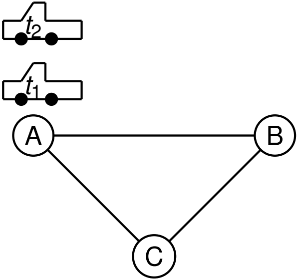
Let OPT be a set of optimal plans for a planning task Π, π be a path, $\mathrm{PIE}(\pi|\textsc{opt}) \supseteq \mathrm{IE}(\pi|\textsc{opt})$ be a set of possible OPT-intended effects of π, and ϕ be a logical formula describing $\mathrm{PIE}(\pi|\textsc{opt})$. Then, for any path π' from $s_0[[\pi]]$, $\pi \cdot \pi' \in \textsc{opt}$ implies $\pi' \models \phi$.

Finding Approximate Intended Effects — Shortcuts

- Intuition: X can not be an intended effect of π if there is a cheaper way to achieve X

- Assume we have some library $\mathscr{L}$ of "shortcut" paths
- $X \subseteq s_0[[\pi]]$ can not be an intended effect of π if there exists some $\pi' \in \mathscr{L}$ such that:
  1. $C(\pi') < C(\pi)$
  2. $X \subseteq s_0[[\pi']]$

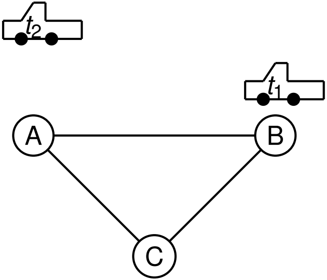
Shortcuts Example



Causal Structure

$\pi = \langle \qquad \rangle$

Shortcuts Example

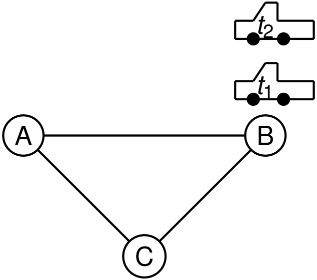


Causal Structure

drive-t_1-A-B

$\pi = \langle$ drive-t_1-A-B $\rangle$

Shortcuts Example

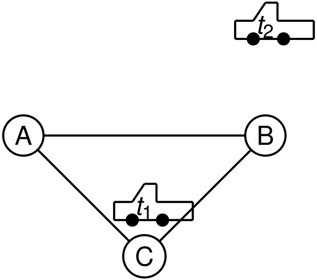


Causal Structure

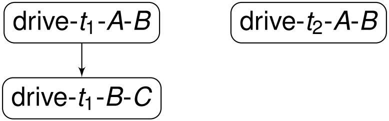
drive-t_1-A-B drive-t_2-A-B

$\pi = \langle$ drive-t_1-A-B ,drive-t_2-A-B $\rangle$

Shortcuts Example

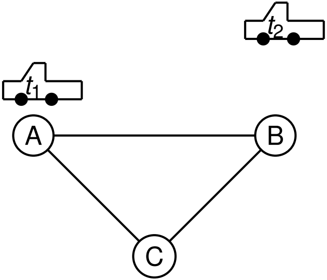


Causal Structure

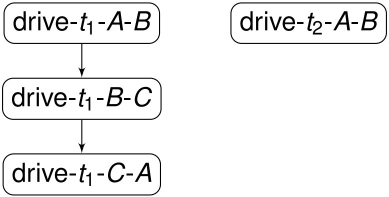


$\pi = \langle$ drive-t_1-A-B ,drive-t_2-A-B ,drive-t_1-B-C $\rangle$

Shortcuts Example

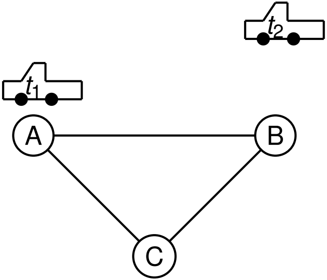


Causal Structure

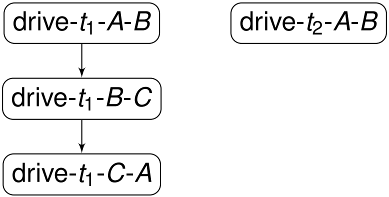


$$\pi = \langle \text{ drive-}t_1\text{-}A\text{-}B\text{ ,drive-}t_2\text{-}A\text{-}B\text{ ,drive-}t_1\text{-}B\text{-}C\text{ ,drive-}t_1\text{-}C\text{-}A \rangle$$

Shortcuts Example



Causal Structure



$\pi = \langle$ drive-t_1-A-B ,drive-t_2-A-B ,drive-t_1-B-C ,drive-t_1-C-A $\rangle$

$\pi' = \langle$drive-t_2-A-$B\rangle$

Shortcuts in Logic Form

- For $X \subseteq s_0[[\pi]]$ to be an intended effect of π, it must achieve something that no shortcut does
- Expressed as a CNF formula:

$$\phi_{\mathscr{L}}(\pi) = \bigwedge_{\pi' \in \mathscr{L} : C(\pi') < C(\pi)} \bigvee_{p \in s_0[[\pi]] \setminus s_0[[\pi']]} p$$

- Each clause of this formula stands for an existential optimal disjunctive action landmark: There must exist some action in some optimal continuation that consumes one of its propositions

Finding Shortcuts

- Where does the shortcut library $\mathscr{L}$ come from?
- It does not need to be static — it can be dynamically generated for each path
- We use the causal structure of the current path — a graph whose nodes are actions, with an edge from a_i to a_j if there is a causal link where a_i provides some proposition for a_j
- We attempt to remove parts of the causal structure, to obtain a “shortcut”

Shortcuts as Landmarks

- The formula $\phi_{\mathscr{L}}(\pi)$ describes $\exists$-opt landmarks — landmarks which occur in some optimal plan
- We can incorporate those landmarks with “regular” landmarks, and derive a heuristic using the cost partitioning method
- The resulting heuristic is path admissible
- To guarantee optimality, we modify A^* to reevaluate $h(s)$ every time a cheaper path to s is found

$\{\rho\}$-path Admissibility

We also have another variant of the heuristic — $\phi_{\mathscr{L}}(\pi|\{\rho\})$

- $\{\rho\}$-admissible
- ρ is the lexicographically lowest optimal plan
- Requires more modifications to A^*

Outline

Coverage

coverage	$\phi_{\mathscr{L}}(\pi)$	$\phi_{\mathscr{L}}(\pi \mid \{\rho\})$	h_{LA}	**LM-**A^*
airport (50)	**28**	27	**28**	28
depot (22)	**5**	**5**	4	4
driverlog (20)	**9**	**9**	7	7
elevators (30)	**7**	0	**7**	7
freecell (80)	**51**	49	**51**	51
mprime (35)	**19**	17	15	15
mystery (30)	**15**	**15**	12	12
parcprinter (30)	**12**	**12**	11	11
pipesworld-tankage (50)	**10**	8	**10**	9
satellite (36)	**6**	4	4	4
sokoban (30)	**15**	0	**15**	15
trucks-strips (30)	**7**	**7**	6	6
SUM	**547**	514	531	530

Only interesting domains are shown

Expansions

expansions	$\phi_{\mathscr{L}}(\pi)$	$\phi_{\mathscr{L}}(\pi \vert \{\rho\})$	h_{LA}
airport (27)	**211052**	420947	211647
blocks (21)	**1064433**	1160581	1070441
depot (4)	**290141**	388822	401696
driverlog (7)	**170534**	224226	363541
freecell (49)	**403030**	556692	**403030**
grid (2)	**227288**	231599	467078
gripper (5)	**458498**	594875	**458498**
logistics00 (20)	**816589**	1487932	862443
logistics98 (3)	**13227**	22014	45654
miconic (141)	**135213**	183319	**135213**
mprime (15)	**35308**	42093	313576
mystery (14)	**37698**	48785	290133
openstacks (12)	**1579931**	1756117	**1579931**
parcprinter (11)	**101178**	146959	158090
pathways (4)	**32287**	58912	173593
pegsol (26)	**3948303**	4364821	**3948303**
pipesworld-notankage (15)	**1248036**	1775363	1377390
pipesworld-tankage (8)	**24080**	36830	28761
psr-small (48)	**358647**	373242	698003
rovers (5)	**98118**	343152	231380
satellite (4)	**5906**	8817	10623
scanalyzer (13)	**22251**	27893	23213
storage (13)	**313259**	359482	475049
tpp (5)	**4227**	7355	12355
transport (9)	**915027**	1062859	929285
trucks-strips (6)	**230699**	314618	1261745
woodworking (11)	**92195**	163589	152975
zenotravel (8)	**66600**	86782	186334
SUM	**12903755**	16248676	16269980

Thank You