Alternatives to Explicit State Space Search
Decoupled Search

Daniel Gnad & Álvaro Torralba

ICAPS'17 Tutorial

June 19, 2017

About us

Daniel Gnad

Dr. Álvaro Torralba

Saarland University, Saarbrücken, Germany

About you

Target audience:

Ideally, you are..

- .. familiar with Classical Planning Formalisms (FDR/SAS$^+$).
- .. familiar with Planning as Heuristic Search.
- .. aware of an important issue in Explicit State Space Search
  $\rightarrow$ State Space Explosion

About you

Target audience:

Ideally, you are..

- .. familiar with Classical Planning Formalisms (FDR/SAS$^+$).
- .. familiar with Planning as Heuristic Search.
- .. aware of an important issue in Explicit State Space Search
  $\rightarrow$ State Space Explosion

Don't hesitate to ask questions if something is unclear!

Agenda

Agenda

Classical Planning

Definition. *A planning task is a 4-tuple* $\Pi = (V, A, I, G)$ *where:*

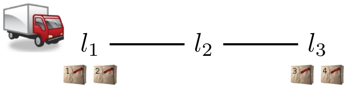
- V *is a set of state variables, each* $v \in V$ *with a finite domain* D_v.
- A *is a set of actions; each* $a \in A$ *is a triple* $(pre(a), \mathit{eff}(a), c(a))$, *of precondition and effect (partial assignments), and the action's cost* $c(a) \in \mathbb{R}^{0+}$.
- *Initial state* I *(complete assignment), goal* G *(partial assignment).*

Classical Planning

Definition. *A planning task is a 4-tuple* $\Pi = (V, A, I, G)$ *where:*

- V *is a set of state variables, each* $v \in V$ *with a finite domain* D_v.
- A *is a set of actions; each* $a \in A$ *is a triple* $(pre(a), eff(a), c(a))$, *of precondition and effect (partial assignments), and the action's cost* $c(a) \in \mathbb{R}^{0+}$.
- *Initial state* I *(complete assignment), goal* G *(partial assignment).*

Running Example:



- $V = \{t, p_1, p_2, p_3, p_4\}$ with $D_t = \{l_1, l_2, l_3\}$ and $D_{p_i} = \{t, l_1, l_2, l_3\}$.
- $A = \{load(p_i, x), unload(p_i, x), drive(x, x')\}$

Semantics – The State Space of a Planning Task

Definition. *Let $\Pi = (V, A, I, G)$ be an FDR planning task. The state space of Π is the labeled transition system $\Theta_\Pi = (S, L, c, T, I, S^G)$ where:*

- *The states S are the complete variable assignments.*
- *The labels $L = A$ are Π's actions; the cost function c is that of Π.*
- *The transitions are $T = \{s \xrightarrow{a} s' \mid pre(a) \subseteq s, s' = s[\![a]\!]\}$.*
  If $pre(a) \subseteq s$, then a is applicable in s and, for all $v \in V$, $s[\![a]\!][v] := \mathit{eff}(a)[v]$ if $\mathit{eff}(a)[v]$ is defined and $s[\![a]\!][v] := s[v]$ otherwise.
  If $pre(a) \not\subseteq s$, then $s[\![a]\!]$ is undefined.
- *The initial state I is identical to that of Π.*
- *The goal states $S^G = \{s \in S \mid G \subseteq s\}$ are those that satisfy Π's goal.*

Semantics – The State Space of a Planning Task

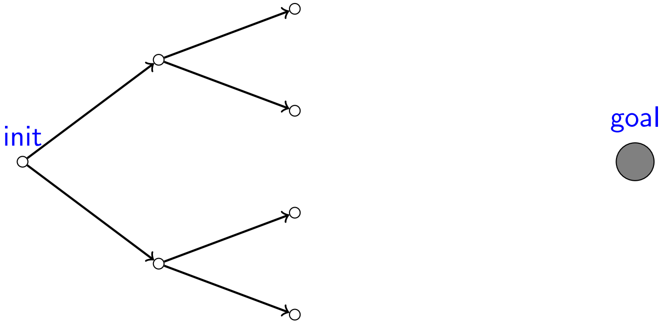
Definition. *Let $\Pi = (V, A, I, G)$ be an FDR planning task. The state space of Π is the labeled transition system $\Theta_\Pi = (S, L, c, T, I, S^G)$ where:*

- *The states S are the complete variable assignments.*
- *The labels $L = A$ are Π's actions; the cost function c is that of Π.*
- *The transitions are $T = \{s \xrightarrow{a} s' \mid pre(a) \subseteq s, s' = s[\![a]\!]\}$.*
  If $pre(a) \subseteq s$, then a is applicable in s and, for all $v \in V$, $s[\![a]\!][v] := \mathit{eff}(a)[v]$ if $\mathit{eff}(a)[v]$ is defined and $s[\![a]\!][v] := s[v]$ otherwise.
  If $pre(a) \not\subseteq s$, then $s[\![a]\!]$ is undefined.
- *The initial state I is identical to that of Π.*
- *The goal states $S^G = \{s \in S \mid G \subseteq s\}$ are those that satisfy Π's goal.*

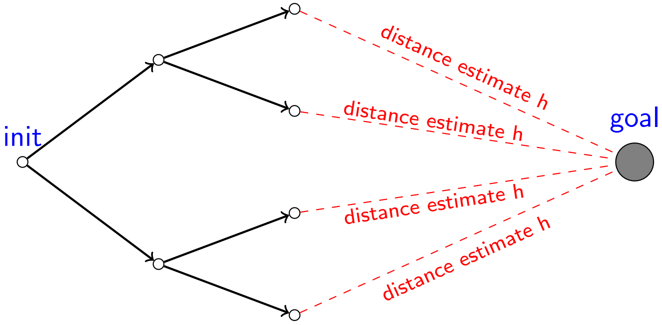
$\rightarrow$ Solution ("Plan"): Action sequence mapping I into $s \in S^G$.
Optimal plan: Minimum summed-up cost.

A successful approach: Heuristic Search

A successful approach: Heuristic Search



A successful approach: Heuristic Search

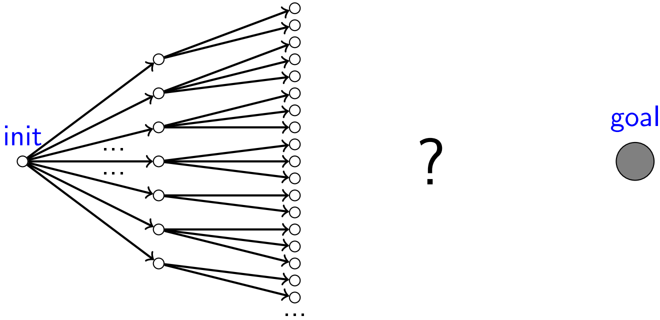


→ Forward state space search. Heuristic function h maps states s to an estimate $h(s)$ of goal distance.

Alternatives to State Space Search (not covered here)

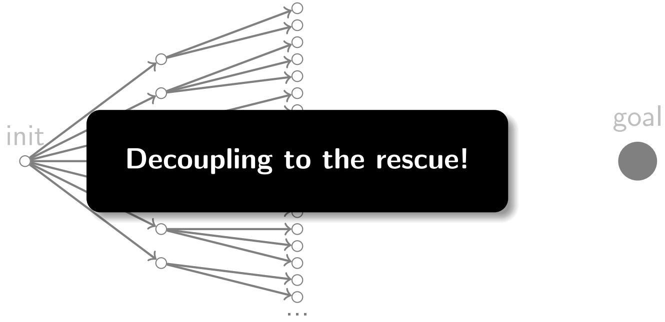
- **Planning as SAT**: Extensions use, e. g., heuristics, symmetry breaking.
  Kautz and Selman (1992, 1996); Ernst *et al.* (1997); Rintanen (1998, 2003, 2012)
- **Property Directed Reachability**
  Bradley (2011); Eén *et al.* (2011); Suda (2014)
- Planning via **Petri Net Unfolding**
  Godefroid and Wolper (1991); McMillan (1992); Esparza *et al.* (2002); Edelkamp *et al.* (2004); Hickmott *et al.* (2007); Bonet *et al.* (2008, 2014)
- **Partial-order Planning**
  Sacerdoti (1975); Kambhampati *et al.* (1995); Younes and Simmons (2003); Bercher *et al.* (2013)
- **Factored Planning** (details later)
  Knoblock (1994); Amir and Engelhardt (2003); Brafman and Domshlak (2006); Kelareva *et al.* (2007); Brafman and Domshlak (2008, 2013); Fabre *et al.* (2010)
- ...

State Space Explosion



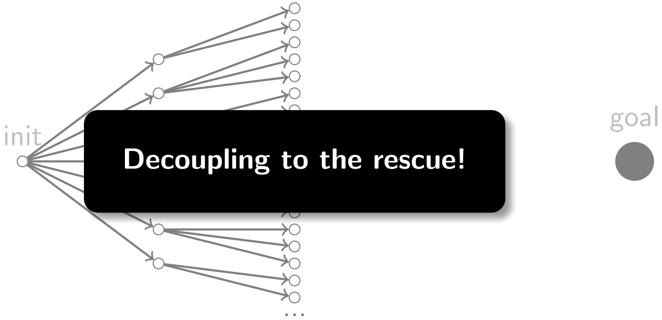
Huge branching factor $\rightarrow$ state space *explosion*.

State Space Explosion



Huge branching factor → state space *explosion*.

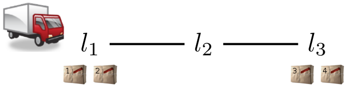
State Space Explosion



Huge branching factor $\rightarrow$ state space *explosion*.
Helmert and Röger (2008)

Agenda

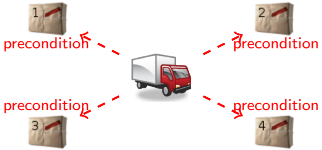
Star-Topology Decoupling



Running Example:

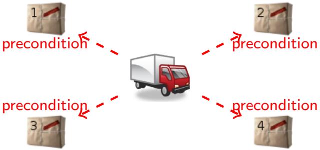
- $V = \{t, p_1, p_2, p_3, p_4\}$.
- $A = \{load(p_i, x), unload(p_i, x), drive(x, x')\}$, where:
  $pre(load(p_i, x)) = \{(t, x), (p_i, x)\}$ and $eff(load(i, x)) = \{(p_i, t)\}$,
  $pre(unload(p_i, x)) = \{(t, x), (p_i, t)\}$ and $eff(unload(i, x)) = \{(p_i, x)\}$.

Causal Graph: **Dependencies across (components of) state variables.**



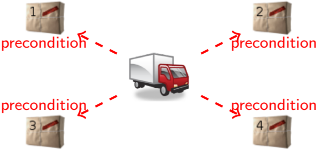
Star-Topology Decoupling

Causal Graph: **Dependencies across (components of) state variables.**



Star-Topology Decoupling

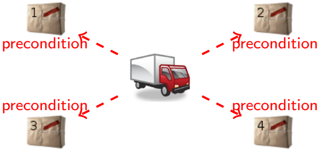
Causal Graph: **Dependencies across (components of) state variables.**



Decomposition: “Instantiate center to break the conditional dependencies”.

Star-Topology Decoupling

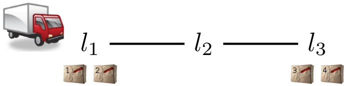
Causal Graph: **Dependencies across (components of) state variables.**



Decomposition: "Instantiate center to break the conditional dependencies".

Search over center actions; handle each leaf component separately.

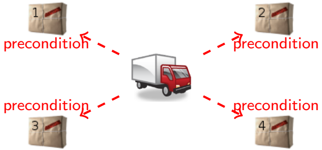
Star-Topology Decoupling



Running Example:

- $V = \{t, p_1, p_2, p_3, p_4\}$.
- $A = \{load(p_i, x), unload(p_i, x), drive(x, x')\}$, where:
  $pre(load(p_i, x)) = \{(t, x), (p_i, x)\}$ and $eff(load(i, x)) = \{(p_i, t)\}$,
  $pre(unload(p_i, x)) = \{(t, x), (p_i, t)\}$ and $eff(unload(i, x)) = \{(p_i, x)\}$.

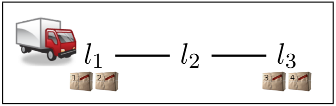
Causal Graph: **Dependencies across (components of) state variables.**



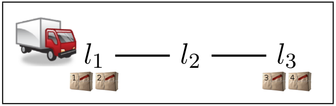
Decomposition: "Instantiate center to break the conditional dependencies".

Search over center actions; handle each leaf component separately.

"Conditional Independence"



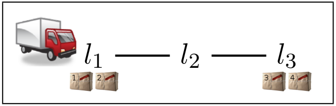
"Conditional Independence"



Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

“Conditional Independence”



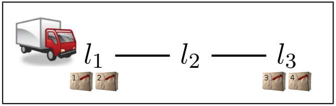
Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

"Conditional Independence"



Center path:

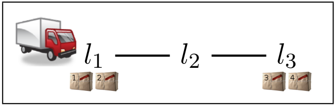
$$l_1 \xrightarrow{\text{drive}(l_1,l_2)} l_2 \xrightarrow{\text{drive}(l_2,l_3)} l_3 \xrightarrow{\text{drive}(l_3,l_2)} l_2 \xrightarrow{\text{drive}(l_2,l_1)} l_1$$

:

(a) $l_1 \xrightarrow{\text{load}(p_1,l_1)} t \xrightarrow{\text{unload}(p_1,l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1,l_1)} t \qquad\qquad \xrightarrow{\text{unload}(p_1,l_3)} l_3$

"Conditional Independence"



Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

:

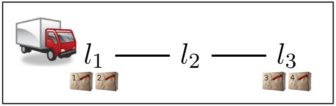
(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

:

(a) $l_3 \xrightarrow{\text{load}(p_3, l_3)} t \xrightarrow{\text{unload}(p_3, l_2)} l_2$

"Conditional Independence"



Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

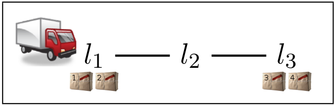
(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

:

(a) $l_3 \xrightarrow{\text{load}(p_3, l_3)} t \xrightarrow{\text{unload}(p_3, l_2)} l_2$

(b) $l_3 \xrightarrow{\text{load}(p_3, l_3)} t \qquad \xrightarrow{\text{unload}(p_3, l_1)} l_1$

“Conditional Independence”



Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

:

(a) $l_3 \xrightarrow{\text{load}(p_3, l_3)} t \xrightarrow{\text{unload}(p_3, l_2)} l_2$

(b) $l_3 \xrightarrow{\text{load}(p_3, l_3)} t \qquad \xrightarrow{\text{unload}(p_3, l_1)} l_1$

We can choose (a) or (b) for each of p_1 and p_3 independently

$\Longrightarrow$ **Maintain the compliant paths for each leaf separately.**

Exponential Reduction of the State Space

Domain	Success: Std	Success: POR	Success: Unfold.	Success: Decoupled	Representation Size (in Thousands): Std	Representation Size (in Thousands): POR	Representation Size (in Thousands): Decoupled
Reachable State Space. Right: Average over Instances Commonly Built							
Solvable Benchmarks: From the International Planning Competition (IPC)							
Depots	4	4	2	**5**	30,954.8	30,954.8	**3,970.0**
Driverlog	5	5	3	**10**	35,632.4	35,632.4	**127.2**
Elevators	21	17	3	**41**	22,652.1	22,651.1	**186.7**
Logistics	12	12	11	**27**	3,793.8	3,793.8	**8.2**
Miconic	50	45	30	**145**	52,728.9	52,673.1	**2.4**
NoMystery	11	11	7	**40**	29,459.3	25,581.5	**10.0**
Pathways	**4**	**4**	3	**4**	54,635.5	**1,229.0**	11,211.9
PSR	3	3	3	3	39.4	33.9	**11.1**
Rovers	5	**6**	4	5	98,051.6	6,534.4	**4,032.9**
Satellite	**5**	**5**	**5**	4	2,864.2	582.5	**352.7**
TPP	5	5	4	**11**	340,961.5	326,124.8	**.8**
Transport	28	23	11	**34**	4,958.6	4,958.5	**173.3**
Woodworking	11	20	**22**	16	438,638.5	**226.8**	9,688.9
Zenotravel	**7**	**7**	4	**7**	17,468.0	17,467.5	**99.4**
Unsolvable Benchmarks: Extended from Hoffmann and Nebel (2001)							
NoMystery	9	8	4	**40**	85,254.2	65,878.2	**3.8**
Rovers	**4**	**4**	0	**4**	697,778.9	302,608.9	**20,924.4**
$\sum$	186	181	116	**398**			

Exponential Reduction of the State Space

Domain	Std	POR	Unfold.	Decoupled	Std	POR	Decoupled
	Reachable State Space. Success				Right: Average over Instances Commonly Built Representation Size (in Thousands)		
Solvable Benchmarks: From the International Planning Competition (IPC)							
Depots	4	4	2	**5**	30,954.8	30,954.8	**3,970.0**
Driverlog	5	5	3	**10**	35,632.4	35,632.4	**127.2**
Elevators	21	17	3	**41**	22,652.1	22,651.1	**186.7**
Logistics	12	12	11	**27**	3,793.8	3,793.8	**8.2**
Miconic	50	45	30	**145**	52,728.9	52,673.1	**2.4**
NoMystery	11	11	7	**40**	29,459.3	25,581.5	**10.0**
Pathways	**4**	**4**	3	**4**	54,635.5	**1,229.0**	11,211.9
PSR	3	3	3	3	39.4	33.9	**11.1**
Rovers	5	**6**	4	5	98,051.6	6,534.4	**4,032.9**
Satellite	**5**	**5**	**5**	4	2,864.2	582.5	**352.7**
TPP	5	5	4	**11**	340,961.5	326,124.8	**.8**
Transport	28	23	11	**34**	4,958.6	4,958.5	**173.3**
Woodworking	11	20	**22**	16	438,638.5	**226.8**	9,688.9
Zenotravel	**7**	**7**	4	**7**	17,468.0	17,467.5	**99.4**
Unsolvable Benchmarks: Extended from Hoffmann and Nebel (2001)							
NoMystery	9	8	4	**40**	85,254.2	65,878.2	**3.8**
Rovers	**4**	**4**	0	**4**	697,778.9	302,608.9	**20,924.4**
$\sum$	186	181	116	**398**			

Decoupled Search can be viewed as a form of Petri Net Unfolding, exploiting the star topology to avoid the hardness of detecting the reachable markings.

Agenda

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Definition *$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F \in \mathcal{F}$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.*

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Definition $\mathcal{F}$ *is a star factoring if* $|\mathcal{F}| > 1$ *and there exists* $F^C \in \mathcal{F}$ *such that, for every action* a *where* $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$*, there exists* $F \in \mathcal{F}$ *with* $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ *and* $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.

Center interacts with leaves arbitrarily, **no direct leaf-leaf interaction**.

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Definition *$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F \in \mathcal{F}$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.*

Center interacts with leaves arbitrarily, **no direct leaf-leaf interaction**.

Notation conventions:

- $\mathcal{F} = \{F^C\} \cup \mathcal{F}^L$. (center + leaves)

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Definition $\mathcal{F}$ *is a star factoring if* $|\mathcal{F}| > 1$ *and there exists* $F^C \in \mathcal{F}$ *such that, for every action* a *where* $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, *there exists* $F \in \mathcal{F}$ *with* $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ *and* $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.

Center interacts with leaves arbitrarily, **no direct leaf-leaf interaction**.

Notation conventions:

- $\mathcal{F} = \{F^C\} \cup \mathcal{F}^L$. (center + leaves)
- Center Actions A^C: affect (have an effect on) a variable $v \in F^C$.
- Leaf Actions A^L: affect only one leaf $F^L \in \mathcal{F}^L$.

Star Factorings

Factoring $\mathcal{F}$:= A partitioning of V into non-empty subsets.

Definition *$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F \in \mathcal{F}$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.*

Center interacts with leaves arbitrarily, **no direct leaf-leaf interaction**.

Notation conventions:

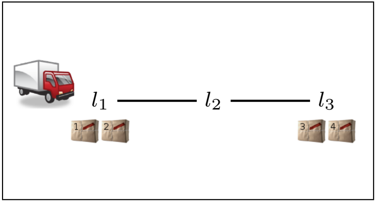
- $\mathcal{F} = \{F^C\} \cup \mathcal{F}^L$. (center + leaves)
- Center Actions A^C: affect (have an effect on) a variable $v \in F^C$.
- Leaf Actions A^L: affect only one leaf $F^L \in \mathcal{F}^L$.
- Center States s^C: complete assignment to F^C.
- Leaf States s^L: complete assignment to an $F^L \in \mathcal{F}^L$.

Decoupled States

Definition *Let* $\mathcal{F}$ *be a star factoring with center* F^C *and leaves* $\mathcal{F}^L$*. A decoupled state* $s^\mathcal{F}$ *is a triple* $(\pi^C(s^\mathcal{F}), \text{center}(s^\mathcal{F}), prices(s^\mathcal{F}))$ *where* $\pi^C(s^\mathcal{F})$ *is a center path,* $\text{center}(s^\mathcal{F})$ *is a center state, and* $prices(s^\mathcal{F})$ *is a pricing function,* $prices(s^\mathcal{F}) : S^L \mapsto \mathbb{R}^{0+} \cup \{\infty\}$*, mapping each leaf state to a non-negative price.*

Decoupled States

Definition *Let $\mathcal{F}$ be a star factoring with center F^C and leaves $\mathcal{F}^L$. A* *decoupled state* $s^{\mathcal{F}}$ *is a triple* $(\pi^C(s^{\mathcal{F}}), \text{center}(s^{\mathcal{F}}), prices(s^{\mathcal{F}}))$ *where* $\pi^C(s^{\mathcal{F}})$ *is a* *center path*, center$(s^{\mathcal{F}})$ *is a* *center state*, *and* $prices(s^{\mathcal{F}})$ *is a* *pricing function*, $prices(s^{\mathcal{F}}) : S^L \mapsto \mathbb{R}^{0+} \cup \{\infty\}$, *mapping each leaf state to a non-negative price.*



	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

The Compliant-Path Graph

Center path:

$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t$ $\xrightarrow{\text{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let $\pi^C = \langle a^C_1, \dots, a^C_n \rangle$ be a center path traversing center states $\langle s^C_0, \dots, s^C_n \rangle$. The π^C-compliant-path graph for a leaf $F^L \in \mathcal{F}^L$, denoted* $\mathrm{CompG}[\pi^C, F^L]$, *is the arc-labeled weighted directed graph whose vertices are $\{s^L_t \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$, and whose arcs are:*

(i) $s^L_t \xrightarrow{a^L} s'^L_t$ *with weight $c(a^L)$ whenever $s^L, s'^L \in S^L[F^L]$ and $a^L \in A^L[F^L] \setminus A^C$ are such that $s^C_t[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, and $s^L[\![a^L]\!] = s'^L$.*

(ii) $s^L_t \xrightarrow{0} s'^L_{t+1}$ *with weight 0 whenever $s^L, s'^L \in S^L[F^L]$ are such that $s^L[\mathcal{V}(pre(a^C_t)) \cap F^L] = pre(a^C_t)[F^L]$ and $s^L[\![a^C_t]\!] = s'^L$.*

Center path:

$l_1 \xrightarrow{\mathsf{drive}(l_1, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_3)} l_3 \xrightarrow{\mathsf{drive}(l_3, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t \xrightarrow{\mathsf{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t$ $\qquad \xrightarrow{\mathsf{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let* $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ *be a center path traversing center states* $\langle s_0^C, \dots, s_n^C \rangle$. *The* π^C*-compliant-path graph for a leaf* $F^L \in \mathcal{F}^L$, *denoted* $\mathsf{CompG}[\pi^C, F^L]$, *is the arc-labeled weighted directed graph whose vertices are* $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$, *and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} s'^L_t$ *with weight* $c(a^L)$ *whenever* $s^L, s'^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, *and* $s^L[\![a^L]\!] = s'^L$.

(ii) $s_t^L \xrightarrow{0} s'^L_{t+1}$ *with weight* 0 *whenever* $s^L, s'^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ *and* $s^L[\![a_t^C]\!] = s'^L$.

Center path:

$l_1 \xrightarrow{\mathsf{drive}(l_1,l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2,l_3)} l_3 \xrightarrow{\mathsf{drive}(l_3,l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2,l_1)} l_1$

:

(a) $l_1 \xrightarrow{\mathsf{load}(p_1,l_1)} t \xrightarrow{\mathsf{unload}(p_1,l_2)} l_2$

(b) $l_1 \xrightarrow{\mathsf{load}(p_1,l_1)} t$ $\xrightarrow{\mathsf{unload}(p_1,l_3)} l_3$

The Compliant-Path Graph

Definition *Let $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ be a center path traversing center states $\langle s_0^C, \dots, s_n^C \rangle$. The π^C-compliant-path graph for a leaf $F^L \in \mathcal{F}^L$, denoted* CompG$[\pi^C, F^L]$, *is the arc-labeled weighted directed graph whose vertices are $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$, and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} {s'}_t^L$ *with weight* $c(a^L)$ *whenever* $s^L, {s'}^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, *and* $s^L[\![a^L]\!] = {s'}^L$.

(ii) $s_t^L \xrightarrow{0} {s'}_{t+1}^L$ *with weight* 0 *whenever* $s^L, {s'}^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ *and* $s^L[\![a_t^C]\!] = {s'}^L$.

Center path:

$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t$ $\xrightarrow{\text{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let* $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ *be a center path traversing center states* $\langle s_0^C, \dots, s_n^C \rangle$*. The* π^C*-compliant-path graph for a leaf* $F^L \in \mathcal{F}^L$*, denoted* $\text{CompG}[\pi^C, F^L]$*, is the arc-labeled weighted directed graph whose vertices are* $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$*, and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} {s'}_t^L$ *with weight* $c(a^L)$ *whenever* $s^L, {s'}^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, *and* $s^L[\![a^L]\!] = {s'}^L$.

(ii) $s_t^L \xrightarrow{0} {s'}_{t+1}^L$ *with weight* 0 *whenever* $s^L, {s'}^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ *and* $s^L[\![a_t^C]\!] = {s'}^L$.

Center path:

$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ be a center path traversing center states $\langle s_0^C, \dots, s_n^C \rangle$. The π^C-compliant-path graph for a leaf $F^L \in \mathcal{F}^L$, denoted* CompG$[\pi^C, F^L]$, *is the arc-labeled weighted directed graph whose vertices are $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$, and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} {s'}_t^L$ *with weight* $c(a^L)$ *whenever* $s^L, {s'}^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, *and* $s^L[\![a^L]\!] = {s'}^L$.

(ii) $s_t^L \xrightarrow{0} {s'}_{t+1}^L$ *with weight* 0 *whenever* $s^L, {s'}^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ *and* $s^L[\![a_t^C]\!] = {s'}^L$.

Center path:

$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ be a center path traversing center states $\langle s_0^C, \dots, s_n^C \rangle$. The π^C-compliant-path graph for a leaf $F^L \in \mathcal{F}^L$, denoted* CompG$[\pi^C, F^L]$*, is the arc-labeled weighted directed graph whose vertices are $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$, and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} {s'}_t^L$ *with weight $c(a^L)$ whenever $s^L, {s'}^L \in S^L[F^L]$ and $a^L \in A^L[F^L] \setminus A^C$ are such that $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$, and $s^L[\![a^L]\!] = {s'}^L$.*

(ii) $s_t^L \xrightarrow{0} {s'}_{t+1}^L$ *with weight 0 whenever $s^L, {s'}^L \in S^L[F^L]$ are such that $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ and $s^L[\![a_t^C]\!] = {s'}^L$.*

Center path:

$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let* $\pi^C = \langle a_1^C, \dots, a_n^C \rangle$ *be a center path traversing center states* $\langle s_0^C, \dots, s_n^C \rangle$. *The* π^C*-compliant-path graph for a leaf* $F^L \in \mathcal{F}^L$*, denoted* $\mathsf{CompG}[\pi^C, F^L]$*, is the arc-labeled weighted directed graph whose vertices are* $\{s_t^L \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$*, and whose arcs are:*

(i) $s_t^L \xrightarrow{a^L} {s'}_t^L$ *with weight* $c(a^L)$ *whenever* $s^L, {s'}^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s_t^C[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$, $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$*, and* $s^L[\![a^L]\!] = {s'}^L$.

(ii) $s_t^L \xrightarrow{0} {s'}_{t+1}^L$ *with weight* 0 *whenever* $s^L, {s'}^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a_t^C)) \cap F^L] = pre(a_t^C)[F^L]$ *and* $s^L[\![a_t^C]\!] = {s'}^L$.

Center path:

$l_1 \xrightarrow{\mathsf{drive}(l_1, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_3)} l_3 \xrightarrow{\mathsf{drive}(l_3, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t \xrightarrow{\mathsf{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t \qquad \xrightarrow{\mathsf{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph

Definition *Let* $\pi^C = \langle a^C_1, \dots, a^C_n \rangle$ *be a center path traversing center states* $\langle s^C_0, \dots, s^C_n \rangle$*. The* π^C*-compliant-path graph for a leaf* $F^L \in \mathcal{F}^L$*, denoted* $\mathsf{CompG}[\pi^C, F^L]$*, is the arc-labeled weighted directed graph whose vertices are* $\{s^L_t \mid s^L \in S^L[F^L], 0 \leq t \leq n\}$*, and whose arcs are:*

(i) $s^L_t \xrightarrow{a^L} {s'}^L_t$ *with weight* $c(a^L)$ *whenever* $s^L, {s'}^L \in S^L[F^L]$ *and* $a^L \in A^L[F^L] \setminus A^C$ *are such that* $s^C_t[\mathcal{V}(pre(a^L)) \cap F^C] = pre(a^L)[F^C]$*,* $s^L[\mathcal{V}(pre(a^L)) \cap F^L] = pre(a^L)[F^L]$*, and* $s^L[\![a^L]\!] = {s'}^L$*.*

(ii) $s^L_t \xrightarrow{0} {s'}^L_{t+1}$ *with weight* 0 *whenever* $s^L, {s'}^L \in S^L[F^L]$ *are such that* $s^L[\mathcal{V}(pre(a^C_t)) \cap F^L] = pre(a^C_t)[F^L]$ *and* $s^L[\![a^C_t]\!] = {s'}^L$*.*

Center path:

$l_1 \xrightarrow{\mathsf{drive}(l_1, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_3)} l_3 \xrightarrow{\mathsf{drive}(l_3, l_2)} l_2 \xrightarrow{\mathsf{drive}(l_2, l_1)} l_1$

:

(a) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t \xrightarrow{\mathsf{unload}(p_1, l_2)} l_2$

(b) $l_1 \xrightarrow{\mathsf{load}(p_1, l_1)} t \qquad \xrightarrow{\mathsf{unload}(p_1, l_3)} l_3$

The Compliant-Path Graph – Example

Center path π^C:

l_1

CompG$[\pi^C, \{p_1\}]$:

$(p_1 = t)_0$

1 (un)load(p_1, l_1)

$\underline{(p_1 = l_1)_0}$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{array}{l} (p_1 = t)_0 \\ 1 \quad \text{(un)load}(p_1, l_1) \\ \underline{(p_1 = l_1)_0} \end{array}$$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2$$

$\text{CompG}[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, leftrightarrow, bend right, "1"', "\text{(un)load}(p_1, l_1)"] & (p_1 = t)_1 \arrow[dd, leftrightarrow, bend left, "1", "\text{(un)load}(p_1, l_2)"'] \\
\underline{(p_1 = l_1)_0} \arrow[r, "0"] & (p_1 = l_1)_1 \\
& (p_1 = l_2)_1
\end{tikzcd}$$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3$$

$\text{CompG}[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, leftrightarrow, bend right, "1\ \text{(un)load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[dd, leftrightarrow, bend left, "\text{(un)load}(p_1, l_2)\ 1"] \\
\underline{(p_1 = l_1)_0} \arrow[r, "0"] & (p_1 = l_1)_1 \\
 & (p_1 = l_2)_1
\end{tikzcd}$$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, leftrightarrow, "1\ (\text{un})\text{load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[r, "0"] \arrow[dd, leftrightarrow, "(\text{un})\text{load}(p_1, l_2)\ 1"] & (p_1 = t)_2 \arrow[ddd, leftrightarrow, "(\text{un})\text{load}(p_1, l_3)\ 1"] \\
\underline{(p_1 = l_1)_0} \arrow[r, "0"] & (p_1 = l_1)_1 \arrow[r, "0"] & (p_1 = l_1)_2 \\
& (p_1 = l_2)_1 \arrow[r, "0"] & (p_1 = l_2)_2 \\
& & (p_1 = l_3)_2
\end{tikzcd}$$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, leftrightarrow, bend right, "1\ \text{(un)load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[r, "0"] \arrow[dd, leftrightarrow, bend left, "\text{(un)load}(p_1, l_2)\ 1"] & (p_1 = t)_2 \arrow[ddd, leftrightarrow, bend left, "\text{(un)load}(p_1, l_3)\ 1"] \\
\underline{(p_1 = l_1)_0} \arrow[r, "0"] & (p_1 = l_1)_1 \arrow[r, "0"] & (p_1 = l_1)_2 \\
& (p_1 = l_2)_1 \arrow[r, "0"] & (p_1 = l_2)_2 \\
& & (p_1 = l_3)_2
\end{tikzcd}$$

The Compliant-Path Graph – Example

Center path π^C:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, bend right, leftrightarrow, "1\ (\text{un})\text{load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[r, "0"] \arrow[d, leftrightarrow, "(\text{un})\text{load}(p_1, l_2)\ 1"] & (p_1 = t)_2 \arrow[r, "0"] \arrow[d, leftrightarrow, "(\text{un})\text{load}(p_1, l_3)\ 1"] & (p_1 = t)_3 \\
\underline{(p_1 = l_1)_0} \arrow[r, "0"] & (p_1 = l_1)_1 \arrow[r, "0"] & (p_1 = l_1)_2 \arrow[r, "0"] & (p_1 = l_1)_3 \\
 & (p_1 = l_2)_1 \arrow[r, "0"] & (p_1 = l_2)_2 \arrow[r, "0"] & (p_1 = l_2)_3 \\
 & & (p_1 = l_3)_2 \arrow[r, "0"] & (p_1 = l_3)_3
\end{tikzcd}$$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

l_1

CompG$[\pi^C, \{p_1\}]$:

$(p_1 = t)_0$

1 (un)load(p_1, l_1)

$\underline{(p_1 = l_1)_0}$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$(p_1 = t)_0$

1 (un)load(p_1, l_1)

$\underline{(p_1 = l_1)_0}$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2$

CompG$[\pi^C, \{p_1\}]$:

$(p_1 = t)_0 \xrightarrow{0} (p_1 = t)_1$

1 (un)load(p_1, l_1)

$\underline{(p_1 = l_1)_0}$

(un)load(p_1, l_2) 1

$(p_1 = l_2)_1$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2 \xrightarrow{\text{drive}(l_2, l_3, p_1)} l_3$$

CompG$[\pi^C, \{p_1\}]$:

$$(p_1 = t)_0 \xrightarrow{0} (p_1 = t)_1$$

$(p_1 = t)_0 \leftrightarrow \underline{(p_1 = l_1)_0}$: 1 (un)load(p_1, l_1)

$(p_1 = t)_1 \leftrightarrow (p_1 = l_2)_1$: (un)load$(p_1, l_2)$ 1

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z = t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2 \xrightarrow{\text{drive}(l_2, l_3, p_1)} l_3$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, bend right, "1\ \text{(un)load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[r, "0"] \arrow[d, bend left, "\text{(un)load}(p_1, l_2)\ 1"] & (p_1 = t)_2 \arrow[d, bend left, "\text{(un)load}(p_1, l_3)\ 1"] \\
\underline{(p_1 = l_1)_0} \arrow[u, bend left] & (p_1 = l_2)_1 \arrow[u, bend right] & (p_1 = l_3)_2 \arrow[u, bend right]
\end{tikzcd}$$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2 \xrightarrow{\text{drive}(l_2, l_3, p_1)} l_3 \xrightarrow{\text{drive}(l_3, l_2, p_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$$\begin{tikzcd}
(p_1 = t)_0 \arrow[r, "0"] \arrow[d, leftrightarrow, "1\ \text{(un)load}(p_1, l_1)"'] & (p_1 = t)_1 \arrow[r, "0"] \arrow[d, leftrightarrow, "\text{(un)load}(p_1, l_2)\ 1"] & (p_1 = t)_2 \arrow[dd, leftrightarrow, "\text{(un)load}(p_1, l_3)\ 1"] \\
\underline{(p_1 = l_1)_0} & (p_1 = l_2)_1 & \\
& & (p_1 = l_3)_2
\end{tikzcd}$$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2 \xrightarrow{\text{drive}(l_2, l_3, p_1)} l_3 \xrightarrow{\text{drive}(l_3, l_2, p_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:

$$(p_1 = t)_0 \xrightarrow{0} (p_1 = t)_1 \xrightarrow{0} (p_1 = t)_2 \xrightarrow{0} (p_1 = t)_3$$

$(p_1 = t)_0 \leftrightarrow (p_1 = l_1)_0$: 1 (un)load$(p_1, l_1)$

$\underline{(p_1 = l_1)_0}$

$(p_1 = t)_1 \leftrightarrow (p_1 = l_2)_1$: (un)load$(p_1, l_2)$ 1

$(p_1 = t)_2 \leftrightarrow (p_1 = l_3)_2$: (un)load$(p_1, l_3)$ 1

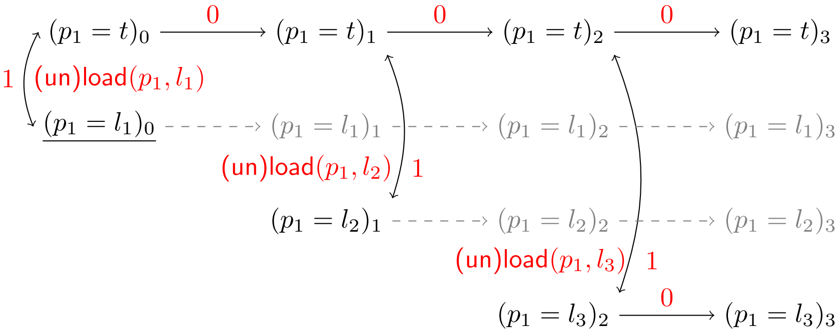
$$(p_1 = l_3)_2 \xrightarrow{0} (p_1 = l_3)_3$$

The Compliant-Path Graph – No-Empty Example

Center path π^C: $pre(drive(l_x, l_y, \mathbf{p_z})) = \{t = l_x, \mathbf{p_z} = \mathbf{t}\}$

$$l_1 \xrightarrow{\text{drive}(l_1, l_2, p_1)} l_2 \xrightarrow{\text{drive}(l_2, l_3, p_1)} l_3 \xrightarrow{\text{drive}(l_3, l_2, p_2)} l_2$$

CompG$[\pi^C, \{p_1\}]$:



Decoupled State Space

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) $S^{\mathcal{F}}$ *is the set of all decoupled states.*

Decoupled State Space

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) $S^{\mathcal{F}}$ *is the set of all decoupled states.*

(ii) $I^{\mathcal{F}}$ *is the decoupled initial state, where* center$(I^{\mathcal{F}}) := I[F^C]$, $\pi^C(I^{\mathcal{F}}) := \langle\rangle$, *and, for every leaf* $F^L \in \mathcal{F}^L$ *and leaf state* $s^L \in S^L[F^L]$, $prices(I^{\mathcal{F}})[s^L]$ *is the cost of a cheapest path from* $I[F^L]_0$ *to* s^L_0 *in* CompG$[\langle\rangle, F^L]$.

Decoupled State Space

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) $S^{\mathcal{F}}$ *is the set of all decoupled states.*

(ii) $I^{\mathcal{F}}$ *is the decoupled initial state, where* $\mathrm{center}(I^{\mathcal{F}}) := I[F^C]$, $\pi^C(I^{\mathcal{F}}) := \langle\rangle$, *and, for every leaf* $F^L \in \mathcal{F}^L$ *and leaf state* $s^L \in S^L[F^L]$, $prices(I^{\mathcal{F}})[s^L]$ *is the cost of a cheapest path from* $I[F^L]_0$ *to* s^L_0 *in* $\mathrm{CompG}[\langle\rangle, F^L]$.

(iii) $S^{\mathcal{F}}_G$ *are the decoupled goal states* s_G, *where* $G[F^C] \subseteq \mathrm{center}(s_G)$ *and, for every* $F^L \in \mathcal{F}^L$, *there exists a leaf goal state* $s^L \in S^L[F^L]$ *s.t.* $G[F^L] \subseteq s^L$ *and* $prices(s_G)[s^L] < \infty$.

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta_\Pi^{\mathcal{F}} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S_G^{\mathcal{F}})$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

(ii) $T^{\mathcal{F}}$ *contains a transition* $(s^{\mathcal{F}} \xrightarrow{a^C} t^{\mathcal{F}}) \in T^{\mathcal{F}}$ *whenever* $a^C \in A^C$ *and* $s^{\mathcal{F}}, t^{\mathcal{F}}$ *are such that:*

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

(ii) $T^{\mathcal{F}}$ *contains a transition* $(s^{\mathcal{F}} \xrightarrow{a^C} t^{\mathcal{F}}) \in T^{\mathcal{F}}$ *whenever* $a^C \in A^C$ *and* $s^{\mathcal{F}}, t^{\mathcal{F}}$ *are such that:*

1. $\pi^C(s^{\mathcal{F}}) \circ \langle a^C \rangle = \pi^C(t^{\mathcal{F}})$;

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta_\Pi^\mathcal{F} = (S^\mathcal{F}, A^C, c|_{A^C}, T^\mathcal{F}, I^\mathcal{F}, S_G^\mathcal{F})$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

(ii) $T^\mathcal{F}$ *contains a transition* $(s^\mathcal{F} \xrightarrow{a^C} t^\mathcal{F}) \in T^\mathcal{F}$ *whenever* $a^C \in A^C$ *and* $s^\mathcal{F}, t^\mathcal{F}$ *are such that:*

1. $\pi^C(s^\mathcal{F}) \circ \langle a^C \rangle = \pi^C(t^\mathcal{F})$;
2. $\mathrm{center}(s^\mathcal{F})[\mathcal{V}(pre(a^C)) \cap F^C] = pre(a^C)[F^C]$;

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta_\Pi^\mathcal{F} = (S^\mathcal{F}, A^C, c|_{A^C}, T^\mathcal{F}, I^\mathcal{F}, S_G^\mathcal{F})$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

(ii) $T^\mathcal{F}$ *contains a transition* $(s^\mathcal{F} \xrightarrow{a^C} t^\mathcal{F}) \in T^\mathcal{F}$ *whenever* $a^C \in A^C$ *and* $s^\mathcal{F}, t^\mathcal{F}$ *are such that:*

1. $\pi^C(s^\mathcal{F}) \circ \langle a^C \rangle = \pi^C(t^\mathcal{F})$;
2. $\mathsf{center}(s^\mathcal{F})[\mathcal{V}(pre(a^C)) \cap F^C] = pre(a^C)[F^C]$;
3. $\mathsf{center}(s^\mathcal{F})[\![a^C]\!] = \mathsf{center}(t^\mathcal{F})$;

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) *The transition labels are the center actions* A^C, *the cost function is that of* Π, *restricted to* A^C.

(ii) $T^{\mathcal{F}}$ *contains a transition* $(s^{\mathcal{F}} \xrightarrow{a^C} t^{\mathcal{F}}) \in T^{\mathcal{F}}$ *whenever* $a^C \in A^C$ *and* $s^{\mathcal{F}}, t^{\mathcal{F}}$ *are such that:*

1. $\pi^C(s^{\mathcal{F}}) \circ \langle a^C \rangle = \pi^C(t^{\mathcal{F}})$;
2. $\text{center}(s^{\mathcal{F}})[\mathcal{V}(pre(a^C)) \cap F^C] = pre(a^C)[F^C]$;
3. $\text{center}(s^{\mathcal{F}})[\![a^C]\!] = \text{center}(t^{\mathcal{F}})$;
4. *for every* $F^L \in \mathcal{F}^L$ *where* $\mathcal{V}(pre(a^C)) \cap F^L \neq \emptyset$, *there exists* $s^L \in S^L[F^L]$ *s.t.* $s^L[\mathcal{V}(pre(a^C)) \cap F^L] = pre(a^C)[F^L]$ *and* $prices(s)[s^L] < \infty$; *and*

Decoupled State Space – *Cont.*

Definition *The decoupled state space is a labeled transition system* $\Theta^{\mathcal{F}}_{\Pi} = (S^{\mathcal{F}}, A^C, c|_{A^C}, T^{\mathcal{F}}, I^{\mathcal{F}}, S^{\mathcal{F}}_G)$ *as follows:*

(i) *The transition labels are the center actions* A^C*, the cost function is that of* Π*, restricted to* A^C*.*

(ii) $T^{\mathcal{F}}$ *contains a transition* $(s^{\mathcal{F}} \xrightarrow{a^C} t^{\mathcal{F}}) \in T^{\mathcal{F}}$ *whenever* $a^C \in A^C$ *and* $s^{\mathcal{F}}, t^{\mathcal{F}}$ *are such that:*

1. $\pi^C(s^{\mathcal{F}}) \circ \langle a^C \rangle = \pi^C(t^{\mathcal{F}})$;
2. center$(s^{\mathcal{F}})[\mathcal{V}(pre(a^C)) \cap F^C] = pre(a^C)[F^C]$;
3. center$(s^{\mathcal{F}})[\![a^C]\!] =$ center$(t^{\mathcal{F}})$;
4. *for every* $F^L \in \mathcal{F}^L$ *where* $\mathcal{V}(pre(a^C)) \cap F^L \neq \emptyset$*, there exists* $s^L \in S^L[F^L]$ *s.t.* $s^L[\mathcal{V}(pre(a^C)) \cap F^L] = pre(a^C)[F^L]$ *and* $prices(s)[s^L] < \infty$*; and*
5. *for every leaf* $F^L \in \mathcal{F}^L$ *and leaf state* $s^L \in S^L[F^L]$*,* $prices(t^{\mathcal{F}})[s^L]$ *is the cost of a cheapest path from* $I[F^L]_0$ *to* s^L_n *in* CompG$[\pi^C(t^{\mathcal{F}}), F^L]$*, where* $n := |\pi^C(t^{\mathcal{F}})|$*.*

Decoupled Search – Example

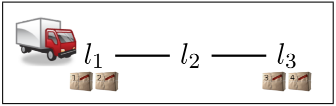
	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0



Decoupled Search – Example

	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

$\xrightarrow{\text{drive}(l_1, l_2)}$

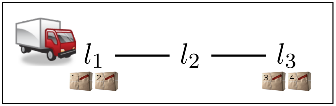
	l_1	t	l_2	l_3
p_1	0	1	**2**	∞

	l_1	t	l_2	l_3
p_2	0	1	**2**	∞

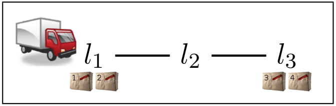
$t = l_2$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0



Decoupled Search – Example



	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

drive(l_1, l_2)

	l_1	t	l_2	l_3
p_1	0	1	**2**	∞

	l_1	t	l_2	l_3
p_2	0	1	**2**	∞

$t = l_2$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

drive(l_2, l_3)

	l_1	t	l_2	l_3
p_1	0	1	2	**2**

	l_1	t	l_2	l_3
p_2	0	1	2	**2**

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	**1**	∞	0

	l_1	t	l_2	l_3
p_4	∞	**1**	∞	0

Decoupled Search – No-Empty Example

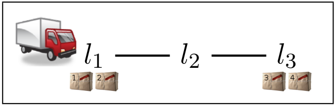
	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0



Decoupled Search – No-Empty Example

	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

$\xrightarrow{\text{drive}(l_1, l_2, p_1)}$

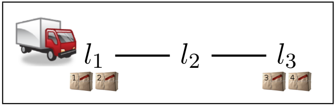
	l_1	t	l_2	l_3
p_1	∞	1	**2**	∞

	l_1	t	l_2	l_3
p_2	0	1	**2**	∞

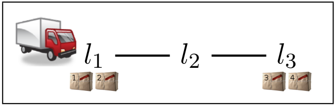
$t = l_2$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0



Decoupled Search – No-Empty Example



	l_1	t	l_2	l_3
p_1	0	1	∞	∞

	l_1	t	l_2	l_3
p_2	0	1	∞	∞

$t = l_1$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

drive(l_1, l_2, p_1)

	l_1	t	l_2	l_3
p_1	∞	1	**2**	∞

	l_1	t	l_2	l_3
p_2	0	1	**2**	∞

$t = l_2$

	l_1	t	l_2	l_3
p_3	∞	∞	∞	0

	l_1	t	l_2	l_3
p_4	∞	∞	∞	0

drive(l_2, l_3, p_2)

	l_1	t	l_2	l_3
p_1	∞	1	2	**2**

	l_1	t	l_2	l_3
p_2	∞	1	∞	**2**

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	**1**	∞	0

	l_1	t	l_2	l_3
p_4	∞	**1**	∞	0

Hypercubes

Definition *A state* p *in* Π *is a member state of a decoupled state* $s^{\mathcal{F}}$*, if* $p[F^C] = \text{center}(s^{\mathcal{F}})$ *and, for all leaves* $F^L \in \mathcal{F}^L$*,* $prices(s^{\mathcal{F}})[p[F^L]] < \infty$*. We say that* p *has cost* $cost_{s^{\mathcal{F}}}(p)$ *in* $s^{\mathcal{F}}$*, where* $cost_{s^{\mathcal{F}}}(p) := \sum_{F^L \in \mathcal{F}^L} prices(s^{\mathcal{F}})[p[F^L]]$*. The hypercube of* $s^{\mathcal{F}}$*, denoted* $[s^{\mathcal{F}}]$*, is the set of all member states of* $s^{\mathcal{F}}$*.*

Hypercubes

Definition *A state p in Π is a member state of a decoupled state $s^{\mathcal{F}}$, if $p[F^C] = \text{center}(s^{\mathcal{F}})$ and, for all leaves $F^L \in \mathcal{F}^L$, $prices(s^{\mathcal{F}})[p[F^L]] < \infty$. We say that p has cost $cost_{s^{\mathcal{F}}}(p)$ in $s^{\mathcal{F}}$, where $cost_{s^{\mathcal{F}}}(p) := \sum_{F^L \in \mathcal{F}^L} prices(s^{\mathcal{F}})[p[F^L]]$. The hypercube of $s^{\mathcal{F}}$, denoted $[s^{\mathcal{F}}]$, is the set of all member states of $s^{\mathcal{F}}$.*

	l_1	t	l_2	l_3
p_1	∞	1	2	2

	l_1	t	l_2	l_3
p_2	∞	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled State $s^{\mathcal{F}}$

$tl_3, p_1t, p_2l_3, p_3l_3, p_4l_3$ 3

$tl_3, p_1l_3, p_2l_3, p_3t, p_4t$ 6

$tl_3, p_1t, p_2t, p_3t, p_4l_3$ 3

$tl_3, p_1l_3, p_2t, p_3t, p_4l_3$ 4

$tl_3, p_1t, p_2l_3, p_3t, p_4t$ 5

...

Hypercube $[s^{\mathcal{F}}]$ (24 member states!)

Hypercubes

Definition *A state p in Π is a* member state *of a decoupled state $s^{\mathcal{F}}$, if* $p[F^C] = \text{center}(s^{\mathcal{F}})$ *and, for all leaves $F^L \in \mathcal{F}^L$,* $prices(s^{\mathcal{F}})[p[F^L]] < \infty$. *We say that p has* cost $cost_{s^{\mathcal{F}}}(p)$ *in $s^{\mathcal{F}}$, where* $cost_{s^{\mathcal{F}}}(p) := \sum_{F^L \in \mathcal{F}^L} prices(s^{\mathcal{F}})[p[F^L]]$. *The* hypercube *of $s^{\mathcal{F}}$, denoted* $[s^{\mathcal{F}}]$, *is the set of all member states of $s^{\mathcal{F}}$.*

	l_1	t	l_2	l_3
p_1	∞	1	2	2

	l_1	t	l_2	l_3
p_2	∞	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled State $s^{\mathcal{F}}$

$tl_3, p_1t, p_2l_3, p_3l_3, p_4l_3$ 3

$tl_3, p_1l_3, p_2l_3, p_3t, p_4t$ 6

$tl_3, p_1t, p_2t, p_3t, p_4l_3$ 3

$tl_3, p_1l_3, p_2t, p_3t, p_4l_3$ 4

$tl_3, p_1t, p_2l_3, p_3t, p_4t$ 5

...

Hypercube $[s^{\mathcal{F}}]$ (24 member states!)

Hypercube dimensions = Leaves; Axis values = Leaf States.

Hypercubes II

$[s^{\mathcal{F}}]$ **contains all states reachable via a path** π **that contains** $\pi^{C}(s^{\mathcal{F}})$**.**

Hypercubes II

$[s^{\mathcal{F}}]$ **contains all states reachable via a path** π **that contains** $\pi^{C}(s^{\mathcal{F}})$**.**

Center path:

$$l_1 \xrightarrow{\text{drive}(l_1, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_3)} l_3 \xrightarrow{\text{drive}(l_3, l_2)} l_2 \xrightarrow{\text{drive}(l_2, l_1)} l_1$$

:

$$l_1 \xrightarrow{\text{load}(p_1, l_1)} t \xrightarrow{\text{unload}(p_1, l_2)} l_2$$

$$\ldots$$

$$l_1 \xrightarrow{\text{load}(p_1, l_1)} t \qquad \xrightarrow{\text{unload}(p_1, l_3)} l_3$$

:

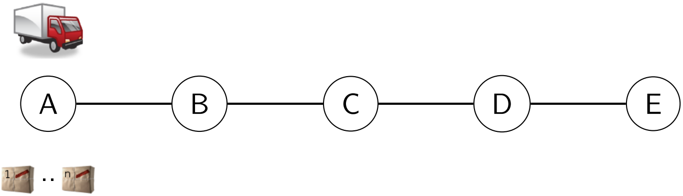
$$l_3 \xrightarrow{\text{load}(p_3, l_3)} t \xrightarrow{\text{unload}(p_3, l_2)} l_2$$

$$\ldots$$

$$l_3 \xrightarrow{\text{load}(p_3, l_3)} t \qquad \xrightarrow{\text{unload}(p_3, l_1)} l_1$$

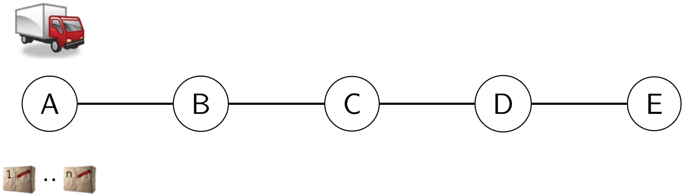
State Space Size Reduction

Illustrative Example:



State Space Size Reduction

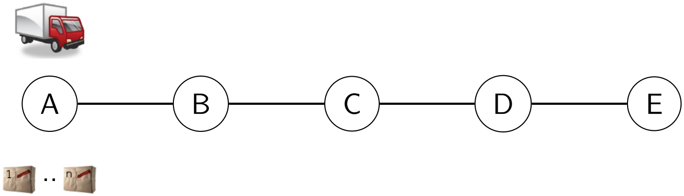
Illustrative Example:



- E.g. loading the packages at A:
  2^n **reachable standard states but only a single decoupled state!**

State Space Size Reduction

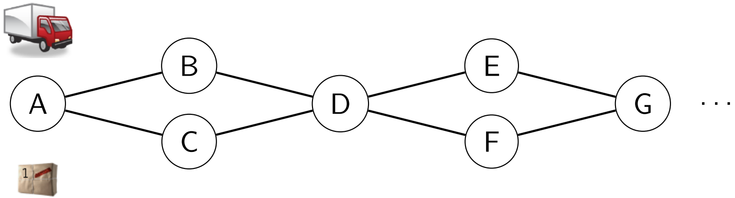
Illustrative Example:



- E.g. loading the packages at A:
  2^n **reachable standard states but only a single decoupled state!**
- Standard state space size here is $5 * 6^n$; decoupled is 15.

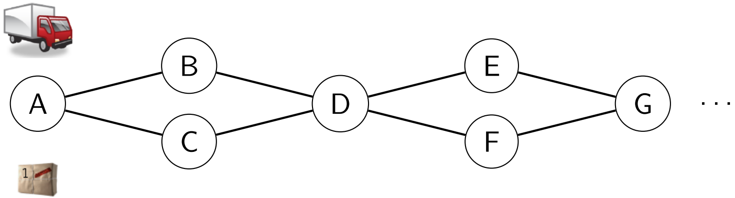
Exponential Blow-Up

Illustrative Example:



Exponential Blow-Up

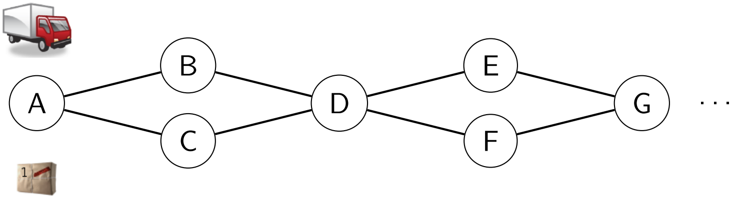
Illustrative Example:



- Drive to D via B or C:
  Standard state space $\rightarrow$ identical state
  Decoupled state space $\rightarrow$ different states! (pricing function)

Exponential Blow-Up

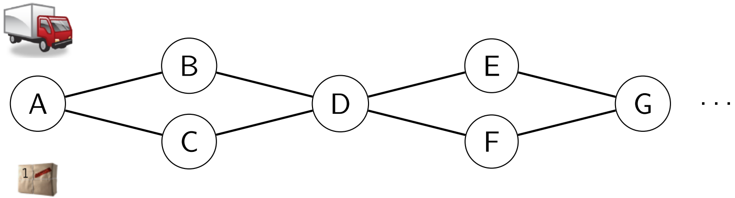
Illustrative Example:



- Drive to D via B or C:
  Standard state space $\rightarrow$ identical state
  Decoupled state space $\rightarrow$ different states! (pricing function)
- Decoupled state "remembers" the taken center path.
- $\Theta_{\Pi}^{\mathcal{F}}$ **exponentially larger** than Θ_{Π}!

Exponential Blow-Up

Illustrative Example:



- Drive to D via B or C:
  Standard state space → identical state
  Decoupled state space → different states! (pricing function)
- Decoupled state "remembers" the taken center path.
- $\Theta_\Pi^\mathcal{F}$ **exponentially larger** than Θ_Π!
- Multiple packages: both effects occur.
  Length of map vs. # of packages.

Agenda

Planning Heuristics

Definition *A heuristic h is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$. Its value $h(s)$ for a state s is referred to as the state's heuristic value, or h value.*

Planning Heuristics

Definition *A heuristic h is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$. Its value $h(s)$ for a state s is referred to as the state's heuristic value, or h value.*

Definition *For a state $s \in S$, the perfect heuristic value h^* of s is the cost of an optimal plan for s, or ∞ if there exists no plan for s.*

$\rightarrow$ Heuristic functions h estimate the remaining cost h^*.

Decoupled Heuristic Functions

Definition. *A decoupled heuristic is a function* h *from decoupled states* $S^{\mathcal{F}}$ *into* $\mathbb{R}_0^+ \cup \infty$.

Decoupled Heuristic Functions

Definition. *A decoupled heuristic is a function* h *from decoupled states* $S^{\mathcal{F}}$ *into* $\mathbb{R}_0^+ \cup \infty$*.*

The center-perfect heuristic h^{C*} *is that where* $h^{C*}(s^{\mathcal{F}})$ *is the cost of a cheapest center path reaching the goal from* $s^{\mathcal{F}}$*.*

Decoupled Heuristic Functions

Definition. *A decoupled heuristic is a function h from decoupled states $S^{\mathcal{F}}$ into $\mathbb{R}_0^+ \cup \infty$.*

The center-perfect heuristic h^{C*} is that where $h^{C*}(s^{\mathcal{F}})$ is the cost of a cheapest center path reaching the goal from $s^{\mathcal{F}}$.

The star-perfect heuristic h^{S*} is that where $h^{S*}(s^{\mathcal{F}})$ is the minimum over path-cost + goal-price for all center paths reaching the goal from $s^{\mathcal{F}}$.

Decoupled Heuristic Functions

Definition. *A decoupled heuristic is a function h from decoupled states $S^{\mathcal{F}}$ into $\mathbb{R}_0^+ \cup \infty$.*

The center-perfect heuristic h^{C*} is that where $h^{C*}(s^{\mathcal{F}})$ is the cost of a cheapest center path reaching the goal from $s^{\mathcal{F}}$.

The star-perfect heuristic h^{S*} is that where $h^{S*}(s^{\mathcal{F}})$ is the minimum over path-cost + goal-price for all center paths reaching the goal from $s^{\mathcal{F}}$.

We say that h is center-admissible if $h \leq h^{C*}$, and star-admissible if $h \leq h^{S*}$.

Decoupled Heuristic Functions

Definition. *A decoupled heuristic is a function h from decoupled states $S^{\mathcal{F}}$ into $\mathbb{R}_0^+ \cup \infty$.*

The center-perfect heuristic h^{C*} is that where $h^{C*}(s^{\mathcal{F}})$ is the cost of a cheapest center path reaching the goal from $s^{\mathcal{F}}$.

The star-perfect heuristic h^{S*} is that where $h^{S*}(s^{\mathcal{F}})$ is the minimum over path-cost + goal-price for all center paths reaching the goal from $s^{\mathcal{F}}$.

We say that h is center-admissible if $h \leq h^{C*}$, and star-admissible if $h \leq h^{S*}$.

Center heuristics h^C estimate h^{C*}, and star heuristics h^S estimate h^{S*}.

$\rightarrow$ But how to compute such heuristics?

Heuristic Compilation

Connect to standard classical planning heuristics!

- **Center heuristics:** Set leaf action costs to 0,

Heuristic Compilation

Connect to standard classical planning heuristics!

- **Center heuristics:** Set leaf action costs to 0, include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost 0.

Heuristic Compilation

Connect to standard classical planning heuristics!

- **Center heuristics:** Set leaf action costs to 0, include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost 0.
- **Star heuristics:** Include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost $prices(s^{\mathcal{F}})[s^L]$.

Heuristic Compilation

Connect to standard classical planning heuristics!

- **Center heuristics:** Set leaf action costs to 0, include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost 0.
- **Star heuristics:** Include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost $prices(s^{\mathcal{F}})[s^L]$.

A^L_{aux} changes for every decoupled state $s^{\mathcal{F}}$!

	l_1	t	l_2	l_3
p_1	0	1	∞	2

	l_1	t	l_2	l_3
p_2	0	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	2	0

	l_1	t	l_2	l_3
p_4	∞	1	2	0

- Can use standard heuristics via this compilation.

Heuristic Compilation

Connect to standard classical planning heuristics!

- **Center heuristics:** Set leaf action costs to 0, include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost 0.
- **Star heuristics:** Include **auxiliary actions** A^L_{aux} allowing to achieve each leaf state s^L reached in $s^{\mathcal{F}}$ at cost $prices(s^{\mathcal{F}})[s^L]$.

A^L_{aux} changes for every decoupled state $s^{\mathcal{F}}$!

	l_1	t	l_2	l_3
p_1	0	1	∞	2

	l_1	t	l_2	l_3
p_2	0	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	2	0

	l_1	t	l_2	l_3
p_4	∞	1	2	0

- Can use standard heuristics via this compilation.
- Heuristic admissible? $\rightarrow$ We can guarantee optimality!

How to guarantee Optimality?

Search Space Reformulation:

- Introduce a new goal state G'.

How to guarantee Optimality?

Search Space Reformulation:

- Introduce a new goal state G'.
- Give all decoupled goal states $s^{\mathcal{F}}$ a transition $s^{\mathcal{F}} \xrightarrow{a} G'$ with $c(a) =$ goal price at $s^{\mathcal{F}}$.

Goal price $= 2 + 0 + 1 = 3$
$=$ price of cheapest member goal state in $s^{\mathcal{F}}$.

	l_1	t	l_2	l_3
p_1	0	1	∞	2

	l_1	t	l_2	l_3
p_2	0	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	2	0

	l_1	t	l_2	l_3
p_4	∞	1	2	0

How to guarantee Optimality?

Search Space Reformulation:

- Introduce a new goal state G'.
- Give all decoupled goal states $s^{\mathcal{F}}$ a transition $s^{\mathcal{F}} \xrightarrow{a} G'$ with $c(a) =$ goal price at $s^{\mathcal{F}}$.
- Extend h^S by $h^S(G') := 0$.

Goal price $= 2 + 0 + 1 = 3$
$=$ price of cheapest member goal state in $s^{\mathcal{F}}$.

	l_1	t	l_2	l_3
p_1	0	1	∞	2

	l_1	t	l_2	l_3
p_2	0	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	2	0

	l_1	t	l_2	l_3
p_4	∞	1	2	0

How to guarantee Optimality?

Search Space Reformulation:

- Introduce a new goal state G'.
- Give all decoupled goal states $s^{\mathcal{F}}$ a transition $s^{\mathcal{F}} \xrightarrow{a} G'$ with $c(a) =$ goal price at $s^{\mathcal{F}}$.
- Extend h^S by $h^S(G') := 0$.

Goal price $= 2 + 0 + 1 = 3$
$=$ price of cheapest member goal state in $s^{\mathcal{F}}$.

	l_1	t	l_2	l_3
p_1	0	1	∞	2

	l_1	t	l_2	l_3
p_2	0	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	2	0

	l_1	t	l_2	l_3
p_4	∞	1	2	0

Any standard (complete/optimal) search on the modified graph yields a (complete/optimal) decoupled search algorithm.

$$\mathbf{A}^* \rightarrow \mathbf{DA}^*$$

Hypercubes

$tl_3, p_1t, p_2l_3, p_3l_3, p_4l_3$ $h = 5$

$tl_3, p_1l_3, p_2l_3, p_3t, p_4t$ $h = 7$

$tl_3, p_1t, p_2t, p_3t, p_4l_3$ $h = 4$

$tl_3, p_1l_3, p_2t, p_3t, p_4l_3$ $h = 2$

$tl_3, p_1t, p_2l_3, p_3t, p_4t$ $h = 5$

$\dots$

Estimate the minimum heuristic value of all member states.

$\rightarrow$ Captured by A^L_{aux}.

	l_1	t	l_2	l_3
p_1	∞	1	2	2

	l_1	t	l_2	l_3
p_2	∞	1	∞	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Agenda

Why no Duplicate Checking?

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$$t = l_3$$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled states are complex!

Why no Duplicate Checking?

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled states are complex!

$\rightarrow$ Identical decoupled states are only generated very rarely.

Why no Duplicate Checking?

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled states are complex!

$\rightarrow$ Identical decoupled states are only generated very rarely.

Instead:

- Dominance relation over decoupled states.
- Prune newly generated states if dominated by already seen ones.

Why no Duplicate Checking?

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Decoupled states are complex!

$\rightarrow$ Identical decoupled states are only generated very rarely.

Instead:

- Dominance relation over decoupled states.
- Prune newly generated states if dominated by already seen ones.

$\rightarrow$ **Guarantees finiteness of decoupled state space!**

Dominance over Decoupled States

When is a decoupled state **"better"** than another one?

Dominance over Decoupled States

When is a decoupled state **"better"** than another one?

New generated state:

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Previously seen state:

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	**2**	0

	l_1	t	l_2	l_3
p_4	∞	1	**2**	0

Dominance over Decoupled States

When is a decoupled state **"better"** than another one?

New generated state:

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	∞	0

	l_1	t	l_2	l_3
p_4	∞	1	∞	0

Previously seen state:

	l_1	t	l_2	l_3
p_1	0	1	2	2

	l_1	t	l_2	l_3
p_2	0	1	2	2

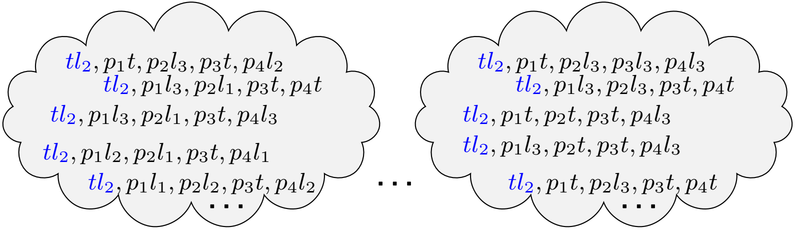
$t = l_3$

	l_1	t	l_2	l_3
p_3	∞	1	**2**	0

	l_1	t	l_2	l_3
p_4	∞	1	**2**	0

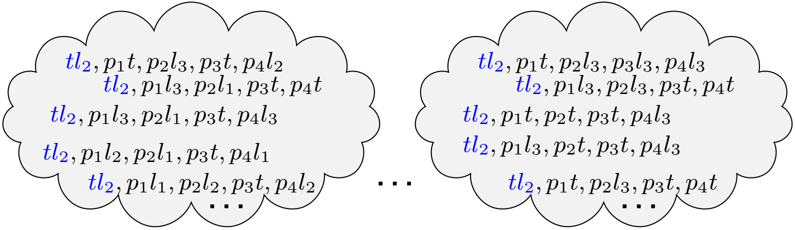
Definition *A decoupled state $s^{\mathcal{F}}$ dominates another state $t^{\mathcal{F}}$, if the center state is the same and* **for all leaf states** s^L*:* $prices(s^{\mathcal{F}})[s^L] \leq prices(t^{\mathcal{F}})[s^L]$.

Hypercube Pruning



Given new state $s^{\mathcal{F}}$, seen states $s_1^{\mathcal{F}}, \dots, s_n^{\mathcal{F}}$ s.t. center$(s_i^{\mathcal{F}})$ = center$(s^{\mathcal{F}})$.

Hypercube Pruning

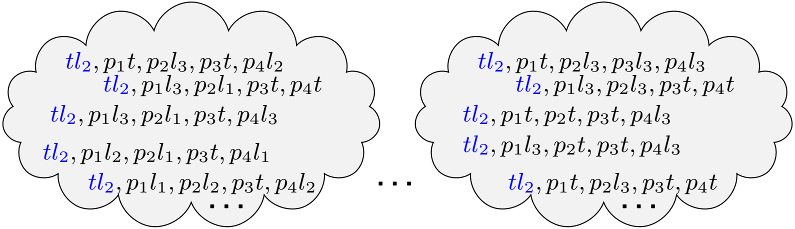


Given new state $s^{\mathcal{F}}$, seen states $s_1^{\mathcal{F}}, \dots, s_n^{\mathcal{F}}$ s.t. center$(s_i^{\mathcal{F}}) =$ center$(s^{\mathcal{F}})$.

Prune $s^{\mathcal{F}}$ if

$$[s^{\mathcal{F}}] \setminus [s_1^{\mathcal{F}}] \cup [s_2^{\mathcal{F}}] \cup \dots \cup [s_n^{\mathcal{F}}] = \emptyset.$$

Hypercube Pruning



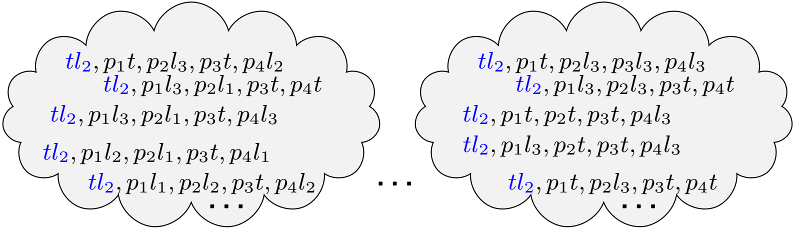
Given new state $s^\mathcal{F}$, seen states $s_1^\mathcal{F}, \dots, s_n^\mathcal{F}$ s.t. center$(s_i^\mathcal{F})$ = center$(s^\mathcal{F})$.

Prune $s^\mathcal{F}$ if

$$[s^\mathcal{F}] \setminus [s_1^\mathcal{F}] \cup [s_2^\mathcal{F}] \cup \cdots \cup [s_n^\mathcal{F}] = \emptyset.$$

$\rightarrow$ Size guarantee

Hypercube Pruning



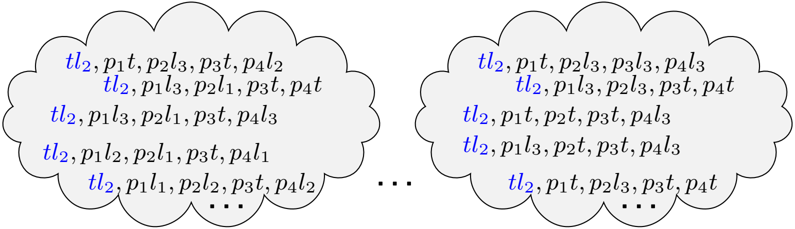
Given new state $s^{\mathcal{F}}$, seen states $s_1^{\mathcal{F}}, \dots, s_n^{\mathcal{F}}$ s.t. center$(s_i^{\mathcal{F}}) =$ center$(s^{\mathcal{F}})$.

Prune $s^{\mathcal{F}}$ if

$$[s^{\mathcal{F}}] \setminus [s_1^{\mathcal{F}}] \cup [s_2^{\mathcal{F}}] \cup \cdots \cup [s_n^{\mathcal{F}}] = \emptyset.$$

$\rightarrow$ Size guarantee, **but** this is an **NP**-hard problem!

Hypercube Pruning



Given new state $s^{\mathcal{F}}$, seen states $s_1^{\mathcal{F}}, \dots, s_n^{\mathcal{F}}$ s.t. center$(s_i^{\mathcal{F}}) =$ center$(s^{\mathcal{F}})$.

Prune $s^{\mathcal{F}}$ if

$$[s^{\mathcal{F}}] \setminus [s_1^{\mathcal{F}}] \cup [s_2^{\mathcal{F}}] \cup \dots \cup [s_n^{\mathcal{F}}] = \emptyset.$$

$\rightarrow$ Size guarantee, **but** this is an **NP**-hard problem!
Cube elimination! Hoffmann and Kupferschmid (2005).

Hypercube Pruning

$tl_2, p_1t, p_2l_3, p_3t, p_4l_2$
$tl_2, p_1l_3, p_2l_1, p_3t, p_4t$
$tl_2, p_1l_3, p_2l_1, p_3t, p_4l_3$
$tl_2, p_1l_2, p_2l_1, p_3t, p_4l_1$
$tl_2, p_1l_1, p_2l_2, p_3t, p_4l_2$
$\dots$

$\dots$

$tl_2, p_1t, p_2l_3, p_3l_3, p_4l_3$
$tl_2, p_1l_3, p_2l_3, p_3t, p_4t$
$tl_2, p_1t, p_2t, p_3t, p_4l_3$
$tl_2, p_1l_3, p_2t, p_3t, p_4l_3$
$tl_2, p_1t, p_2l_3, p_3t, p_4t$
$\dots$

Given new state $s^{\mathcal{F}}$, seen states $s_1^{\mathcal{F}}, \dots, s_n^{\mathcal{F}}$ s.t. center$(s_i^{\mathcal{F}})$ = center$(s^{\mathcal{F}})$.

Prune $s^{\mathcal{F}}$ if

$$[s^{\mathcal{F}}] \setminus [s_1^{\mathcal{F}}] \cup [s_2^{\mathcal{F}}] \cup \dots \cup [s_n^{\mathcal{F}}] = \emptyset.$$

$\rightarrow$ Size guarantee, **but** this is an **NP**-hard problem!
Cube elimination! Hoffmann and Kupferschmid (2005).

Extension to optimality? $\rightarrow$ Need to take prices into account.

Agenda

Variable Dependencies

Definition *The causal graph of* Π *is the directed graph* $CG(\Pi)$ *with vertices* V *and an arc* $u \rightarrow v$ *if* $u \neq v$ *and there exists an action* $a \in A$ *so that either (i) there exists* $a \in A$ *so that* $pre(a)[u]$ *and* $\mathit{eff}(a)[v]$ *are both defined, or (ii) there exists* $a \in A$ *so that* $\mathit{eff}(a)[u]$ *and* $\mathit{eff}(a)[v]$ *are both defined.*

Causal graphs capture variable dependencies.

Variable Dependencies

Definition *The causal graph of Π is the directed graph $CG(\Pi)$ with vertices V and an arc $u \rightarrow v$ if $u \neq v$ and there exists an action $a \in A$ so that either (i) there exists $a \in A$ so that $pre(a)[u]$ and $eff(a)[v]$ are both defined, or (ii) there exists $a \in A$ so that $eff(a)[u]$ and $eff(a)[v]$ are both defined.*

Causal graphs capture variable dependencies.

Definition *The interaction graph of Π given $\mathcal{F}$ is the directed graph $\mathsf{IG}_{\Pi}(\mathcal{F})$, with vertices $\mathcal{F}$, and an arc $F \rightarrow F'$ if $F \neq F'$, and there exist $v \in F$ and $v' \in F'$, s.t. $v \rightarrow v'$ is an arc in $CG(\Pi)$.*

The interaction graph is the quotient of $CG(\Pi)$ over $\mathcal{F}$.

Factored Planning – Related Work

Factoring := partitioning of state variables.

Factored Planning – Related Work

Factoring := partitioning of state variables.

Traditional Factoring: (e. g. Sacerdoti (1974); Knoblock (1994); Lansky and Getoor (1995); Amir and Engelhardt (2003); Fabre *et al.* (2010); Brafman and Domshlak (2013))

- Design factors motivated by abstraction hierarchies or agents.
- Cater for arbitrary cross-factor interactions.

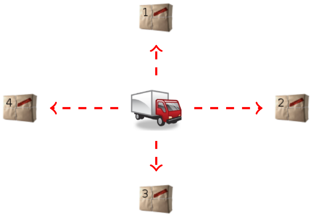
Factored Planning – Related Work

Factoring := partitioning of state variables.

Traditional Factoring: (e. g. Sacerdoti (1974); Knoblock (1994); Lansky and Getoor (1995); Amir and Engelhardt (2003); Fabre *et al.* (2010); Brafman and Domshlak (2013))

- Design factors motivated by abstraction hierarchies or agents.
- Cater for arbitrary cross-factor interactions.

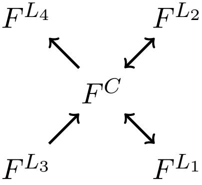
Star Factoring: Force the factoring to induce a star profile!



$\rightarrow$ That is, choose factoring $\mathcal{F}$ so that the interaction graph has a center incident to all arcs.

Factoring Types

Recap:
$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F^L \in \mathcal{F}^L$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.



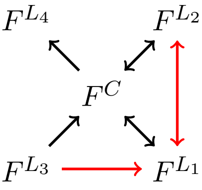
Factoring Types

Recap:
$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F^L \in \mathcal{F}^L$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.

$$\begin{tikzcd}
F^{L_4} & & F^{L_2} \arrow[dd, leftrightarrow] \\
& F^C \arrow[ul] \arrow[ur, leftrightarrow] \arrow[dr, leftrightarrow] & \\
F^{L_3} \arrow[ur] \arrow[rr] & & F^{L_1}
\end{tikzcd}$$

Factoring Types

Recap:
$\mathcal{F}$ is a star factoring if $|\mathcal{F}| > 1$ and there exists $F^C \in \mathcal{F}$ such that, for every action a where $\mathcal{V}(\mathit{eff}(a)) \cap F^C = \emptyset$, there exists $F^L \in \mathcal{F}^L$ with $\mathcal{V}(\mathit{eff}(a)) \subseteq F$ and $\mathcal{V}(\mathit{pre}(a)) \subseteq F \cup F^C$.



Strict-Star Factoring:
Arbitrary center-leaf interaction; no dependencies between leaves directly.

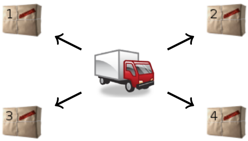
Definition *A factoring $\mathcal{F}$ is a strict-star factoring, if there exists a center factor $F^C \in \mathcal{F}$ such that the arcs in $\mathsf{IG}_\Pi(\mathcal{F})$ are in* $\{F^C \rightarrow F^L \mid F^L \in \mathcal{F} \setminus \{F^C\}\} \cup \{F^C \leftarrow F^L \mid F^L \in \mathcal{F} \setminus \{F^C\}\}$.

How to automatically find a factoring?

Based on the SCCs of the causal graph:

Fork / Inverted-Fork Factorings:
Set leaves to causal graph leaf / root components.

Xshape Factoring:
First do fork factoring, then inverted-fork on resulting center.
Need to take care of leaf-leaf dependencies!

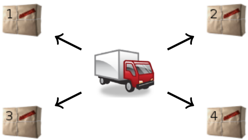


How to automatically find a factoring?

Based on the SCCs of the causal graph:

Fork / Inverted-Fork Factorings:
Set leaves to causal graph leaf / root components.

Xshape Factoring:
First do fork factoring, then inverted-fork on resulting center.
Need to take care of leaf-leaf dependencies!

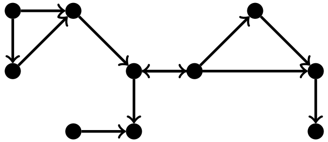


Important criteria:

- **Number of leaves factors** → exponential gain.
- **Leaf flexibility** → *frozen* leaves.
- Leaf size (domain size product) → runtime/memory overhead.
- Structure of leaf state space.

Maximizing the Number of Leaves

Maximum number of leaves in a strict-star factoring?
$\rightarrow$ size of a **maximum independent set (MIS)** of the causal graph.

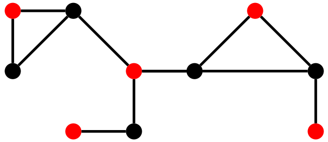


Maximizing the Number of Leaves

Maximum number of leaves in a strict-star factoring?
→ size of a **maximum independent set (MIS)** of the causal graph.

MIS Strategy:

1. Seed factoring = MIS of causal graph (one MIS variables per leaf).

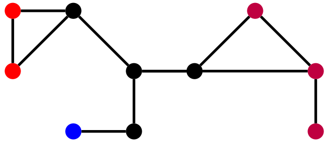


Maximizing the Number of Leaves

Maximum number of leaves in a strict-star factoring?
→ size of a **maximum independent set (MIS)** of the causal graph.

MIS Strategy:

1. Seed factoring = MIS of causal graph (one MIS variables per leaf).
2. Apply post-process to **increase flexibility**.

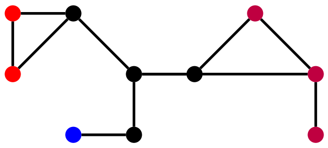


Maximizing the Number of Leaves

Maximum number of leaves in a strict-star factoring?
$\rightarrow$ size of a **maximum independent set (MIS)** of the causal graph.

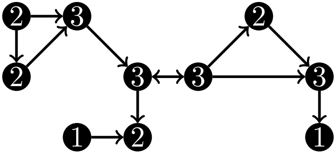
MIS Strategy:

1. Seed factoring = MIS of causal graph (one MIS variables per leaf).
2. Apply post-process to **increase flexibility**.
3. **Abstain** if less than 2 leaves.



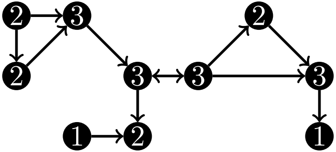
A Greedy Approach – # Incident Arcs (IA)

What if computing an MIS is infeasible?



A Greedy Approach – # Incident Arcs (IA)

What if computing an MIS is infeasible?

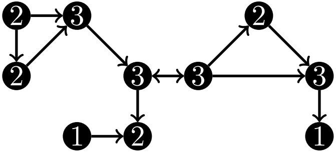


IA Strategy:

1. Move variables most densely connected in causal graph to center.

A Greedy Approach – # Incident Arcs (IA)

What if computing an MIS is infeasible?

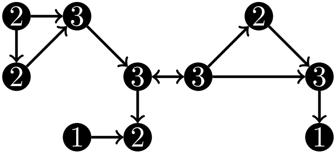


IA Strategy:

1. Move variables most densely connected in causal graph to center.
2. Leaves = weakly connected components in $CG[V \setminus F^C]$.

A Greedy Approach – # Incident Arcs (IA)

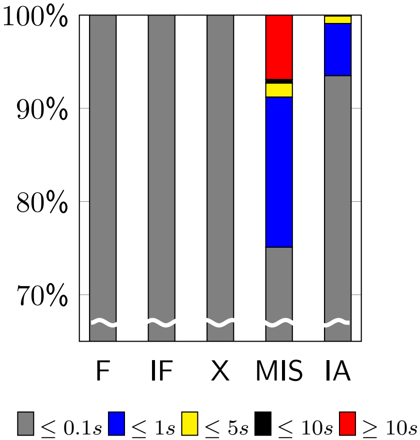
What if computing an MIS is infeasible?



IA Strategy:

1. Move variables most densely connected in causal graph to center.
2. Leaves = weakly connected components in $CG[V \setminus F^C]$.
3. Select factoring with maximum number of **mobile** leaves.
4. **Abstain** if less than 2 leaves.

Factoring Time



Agenda

The interesting stuff . . . after the break

Short Break (5 min)

The interesting stuff . . . after the break

Short Break (5 min)

Implementation in Fast Downward

The interesting stuff . . . after the break

Short Break (5 min)

Implementation in Fast Downward

Hands-On – Implement your own heuristic in decoupled search

Instructions:

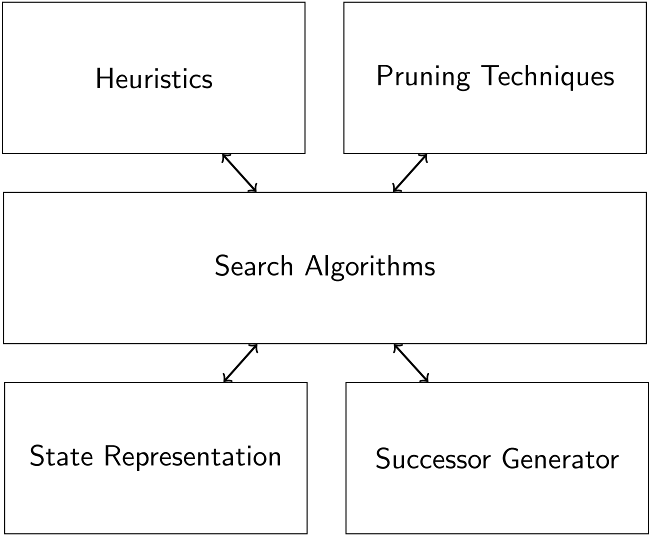
hg clone `https://bitbucket.org/dagnad/decoupled-fd`

hg up icaps-tutorial

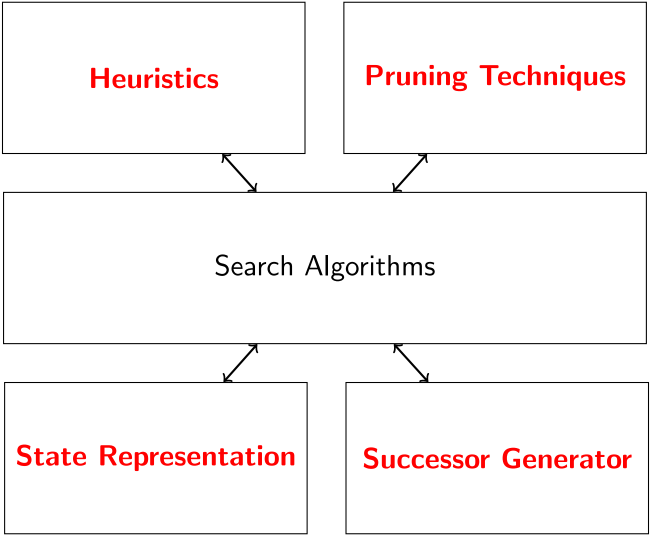
The relevant files are `../src/search/icaps_heuristic.*`

The URL is linked on `http://fai.uni-saarland.de/software.html` ("Decoupled Fast Downward").

Decoupled Search within Fast Downward (Helmert (2006))



Decoupled Search within Fast Downward (Helmert (2006))



Representing Decoupled States

Decoupled State Registry: In fact, many state registries!

Representing Decoupled States

Decoupled State Registry: In fact, many state registries!

- Center state registry,
- One registry for every leaf factor $\rightarrow$ store every leaf state only once.

Representing Decoupled States

Decoupled State Registry: In fact, many state registries!

- Center state registry,
- One registry for every leaf factor $\rightarrow$ store every leaf state only once.

Decoupled State: a pair of

- Center State + **auxiliary variable** to distinguish decoupled states with the same center,
- Instance of "CompliantPathGraph" (many sub-classes).

$\rightarrow$ Search Algorithms use (augmented) center state only.

$\rightarrow$ Accessing pricing function via a vector indexed by aug. center state ID.

Representing Decoupled States II

CompliantPathGraphs:
Many different variants for **optimal/satisficing** planning, more elaborate **dominance pruning** methods, **symbolic leaves**, ...

Representing Decoupled States II

CompliantPathGraphs:
Many different variants for **optimal/satisficing** planning, more elaborate **dominance pruning** methods, **symbolic leaves**, ...

Pricing Function (optimal planning):

- A `vector<int>` for each leaf factor.
- vector indexed by leaf state IDs.

Representing Decoupled States II

CompliantPathGraphs:
Many different variants for **optimal/satisficing** planning, more elaborate **dominance pruning** methods, **symbolic leaves**, ...

Pricing Function (optimal planning):

- A `vector<int>` for each leaf factor.
- vector indexed by leaf state IDs.

Update of Pricing Function:

- Run uniform-cost search from currently reached leaf states.
- Cache entire leaf state spaces to be more efficient.

Interface to Heuristics/Pruning/Successor Generator

```
CPGStorage::get_cpg(State)
```

→ Global access to the pricing function of a decoupled state.

Interface to Heuristics/Pruning/Successor Generator

`CPGStorage::get_cpg(State)`

→ Global access to the pricing function of a decoupled state.

Access price/reachability of a leaf state by its ID + factor.

Functions:

- `has_leaf_state(id, factor)`,
- `get_cost_of_state(id, factor)`,
- `get_number_state(factor)`,
- `goal_reachable(factor)`.

Wanna Take a Look at the Code?

Wanna Take a Look at the Code?

It's a ~~trap~~ mess!

Hands-On Decoupled Search in Fast Downward

`CPGStorage::get_cpg(State)`

→ Global access to the pricing function of a decoupled state.

Functions:

- `has_leaf_state(id, factor)`,
- `get_cost_of_state(id, factor)`,
- `get_number_state(factor)`,
- `goal_reachable(factor)`.

Hands-On Decoupled Search in Fast Downward

`CPGStorage::get_cpg(State)`

→ Global access to the pricing function of a decoupled state.

Functions:

- `has_leaf_state(id, factor)`,
- `get_cost_of_state(id, factor)`,
- `get_number_state(factor)`,
- `goal_reachable(factor)`.

Hands-on:

Want to implement a simple heuristic function?

A mixture between goal-counting and PDB heuristic!

Hands-On Decoupled Search in Fast Downward

`CPGStorage::get_cpg(State)`

→ Global access to the pricing function of a decoupled state.

Functions:

- `has_leaf_state(id, factor)`,
- `get_cost_of_state(id, factor)`,
- `get_number_state(factor)`,
- `goal_reachable(factor)`.

Hands-on:

Want to implement a simple heuristic function?

A mixture between goal-counting and PDB heuristic!

`g_min_goal_cost`: precomputed minimum goal cost with "patterns" $\mathcal{F}^L$.

Build: `./build_all`

Execute: `./fast-downward.py [task-file] --decoupling "fork" --search "astar(icaps)"`

Agenda

What's Done

Basic Framework is Established! Gnad and Hoffmann (2015); Gnad *et al.* (2015)

What's Done

Basic Framework is Established! Gnad and Hoffmann (2015); Gnad *et al.* (2015)

+ lots of extensions:

- **Factoring methods**, target-profile factoring. Gnad *et al.* (2017a)
  → Tomorrow at HSDIP!
- **Partial-order Reduction**. Gnad *et al.* (2016)
- **Symmetry Breaking**. Gnad *et al.* (2017c)
  → Friday morning!
- **Dominance Pruning**. Torralba *et al.* (2016)
- Combination with **symbolic search**, BDDs to represent leaf state spaces. Gnad *et al.* (2017b)

Alternative Planning Formulations

It works for Classical Planning, but what about . . .

- **Numerical Planning?**
- **Probabilistic Planning?**
- **Temporal Planning?**
- **Generalized Planning?**

Beyond Delete-Relaxation

What we have:

$$h^{\mathsf{max}}, h^{\mathsf{add}}, h^{\mathsf{FF}}, h^{\mathsf{LM\text{-}cut}}$$

How to connect to **other types of heuristics**?

- **Abstraction Heuristics**
  - Patter Database Heuristics (PDB)
  - Merge-And-Shrink
  - Cartesian Abstractions
- **Linear-Programming-based Heuristics**
  - Operator Counting
  - Potential Heuristics
- **Landmark Heuristics**
- ...

Beyond Classical Planning

Why scramble for star topologies in IPC benchmarks when the world is full of applications that have a star topology by definition?

- **Multiple agents** (leaves) that interact on a set of shared variables (center).
- **Model Checking!** E.g., client-server architectures; parallel processors with central memory (weak memory models).
- Decoupled Search vs. **Petri Net Unfolding**.

Take Home Message

Explicit Search is the most prominent approach to tackle classical planning problems.

- Full generality (A^*, GBFS, hill-climbing, ...).
- Powerful heuristics.
- Various search enhancements available (e. g., successor pruning)
- Works well across all (IPC) domains.

Take Home Message

Explicit Search is the most prominent approach to tackle classical planning problems.

- Full generality (A^*, GBFS, hill-climbing, ...).
- Powerful heuristics.
- Various search enhancements available (e. g., successor pruning)
- Works well across all (IPC) domains.
- **Has its limitations!**

Take Home Message

Explicit Search is the most prominent approach to tackle classical planning problems.

- Full generality (A^*, GBFS, hill-climbing, ...).
- Powerful heuristics.
- Various search enhancements available (e. g., successor pruning)
- Works well across all (IPC) domains.
- **Has its limitations!**

Alternative State Representations like Decoupled Search offer:

- More specialized way to exploit, e.g., conditional independence.
- Potentially **exponential gain** over explicit search.

Take Home Message

Explicit Search is the most prominent approach to tackle classical planning problems.

- Full generality (A^*, GBFS, hill-climbing, …).
- Powerful heuristics.
- Various search enhancements available (e. g., successor pruning)
- Works well across all (IPC) domains.
- **Has its limitations!**

Alternative State Representations like Decoupled Search offer:

- More specialized way to exploit, e.g., conditional independence.
- Potentially **exponential gain** over explicit search.
- Easy to Use:
    - Implemented in Fast Downward.
    - Many heuristics + pruning methods applicable.
    - Factoring process is fast!

References I

Eyal Amir and Barbara Engelhardt. Factored planning. In G. Gottlob, editor, *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*, pages 929–935, Acapulco, Mexico, August 2003. Morgan Kaufmann.

Pascal Bercher, Thomas Geier, and Susanne Biundo. Using state-based planning heuristics for partial-order causal-link planning. In Ingo J. Timm and Matthias Thimm, editors, *Proceedings of the 36th Annual German Conference on Artificial Intelligence (KI'11)*, volume 8077 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013.

Blai Bonet, Patrik Haslum, Sarah L. Hickmott, and Sylvie Thiébaux. Directed unfolding of petri nets. *Transactions on Petri Nets and Other Models of Concurrency*, 1:172–198, 2008.

Blai Bonet, Patrik Haslum, Victor Khomenko, Sylvie Thiébaux, and Walter Vogler. Recent advances in unfolding technique. *Theoretical Computer Science*, 551:84–101, 2014.

Aaron R. Bradley. Sat-based model checking without unrolling. In *Proceedings of the 12th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'11)*, pages 70–87, 2011.

References II

Ronen Brafman and Carmel Domshlak. Factored planning: How, when, and when not. In Yolanda Gil and Raymond J. Mooney, editors, *Proceedings of the 21st National Conference of the American Association for Artificial Intelligence (AAAI'06)*, pages 809–814, Boston, Massachusetts, USA, July 2006. AAAI Press.

Ronen I. Brafman and Carmel Domshlak. From one to many: Planning for loosely coupled multi-agent systems. In Jussi Rintanen, Bernhard Nebel, J. Christopher Beck, and Eric Hansen, editors, *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS'08)*, pages 28–35. AAAI Press, 2008.

Ronen Brafman and Carmel Domshlak. On the complexity of planning for agent teams and its implications for single agent planning. *Artificial Intelligence*, 198:52–71, 2013.

Stefan Edelkamp, Stefan Leue, and Alberto Lluch-Lafuente. Partial-order reduction and trail improvement in directed model checking. *International Journal on Software Tools for Technology Transfer*, 6(4):277–301, 2004.

Niklas Eén, Alan Mishchenko, and Robert K. Brayton. Efficient implementation of property directed reachability. In *International Conference on Formal Methods in Computer-Aided Design, FMCAD '11, Austin, TX, USA*, pages 125–134, 2011.

References III

Michael D. Ernst, Todd D. Millstein, and Daniel S. Weld. Automatic SAT-compilation of planning problems. In M. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 1169–1177, Nagoya, Japan, August 1997. Morgan Kaufmann.

Javier Esparza, Stefan Römer, and Walter Vogler. An improvement of mcmillan's unfolding algorithm. *Formal Methods in System Design*, 20(3):285–310, 2002.

Eric Fabre, Loïg Jezequel, Patrik Haslum, and Sylvie Thiébaux. Cost-optimal factored planning: Promises and pitfalls. In Ronen I. Brafman, Hector Geffner, Jörg Hoffmann, and Henry A. Kautz, editors, *Proceedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS'10)*, pages 65–72. AAAI Press, 2010.

Daniel Gnad and Jörg Hoffmann. Beating LM-cut with h^{max} (sometimes): Fork-decoupled state space search. In Ronen Brafman, Carmel Domshlak, Patrik Haslum, and Shlomo Zilberstein, editors, *Proceedings of the 25th International Conference on Automated Planning and Scheduling (ICAPS'15)*, pages 88–96. AAAI Press, 2015.

References IV

Daniel Gnad, Jörg Hoffmann, and Carmel Domshlak. From fork decoupling to star-topology decoupling. In Levi Lelis and Roni Stern, editors, *Proceedings of the 8th Annual Symposium on Combinatorial Search (SOCS'15)*, pages 53–61. AAAI Press, 2015.

Daniel Gnad, Martin Wehrle, and Jörg Hoffmann. Decoupled strong stubborn sets. In Subbarao Kambhampati, editor, *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*, pages 3110–3116. AAAI Press/IJCAI, 2016.

Daniel Gnad, Valerie Poser, and Jörg Hoffmann. Beyond forks: Finding and ranking star factorings for decoupled search. In Carles Sierra, editor, *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*. AAAI Press/IJCAI, 2017.

Daniel Gnad, Álvaro Torralba, and Jörg Hoffmann. Symbolic leaf representation in decoupled search. In Alex Fukunaga and Akihiro Kishimoto, editors, *Proceedings of the 10th Annual Symposium on Combinatorial Search (SOCS'17)*. AAAI Press, 2017.

References V

Daniel Gnad, Álvaro Torralba, Alexander Shleyfman, and Jörg Hoffmann. Symmetry breaking in star-topology decoupled search. In *Proceedings of the 27th International Conference on Automated Planning and Scheduling (ICAPS'17)*. AAAI Press, 2017.

Patrice Godefroid and Pierre Wolper. Using partial orders for the efficient verification of deadlock freedom and safety properties. In *Proceedings of the 3rd International Workshop on Computer Aided Verification (CAV'91)*, pages 332–342, 1991.

Malte Helmert and Gabriele Röger. How good is almost perfect? In Dieter Fox and Carla Gomes, editors, *Proceedings of the 23rd National Conference of the American Association for Artificial Intelligence (AAAI'08)*, pages 944–949, Chicago, Illinois, USA, July 2008. AAAI Press.

Malte Helmert. The Fast Downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

Sarah L. Hickmott, Jussi Rintanen, Sylvie Thiébaux, and Langford B. White. Planning via petri net unfolding. In Manuela Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 1904–1911, Hyderabad, India, January 2007. Morgan Kaufmann.

References VI

Jörg Hoffmann and Sebastian Kupferschmid. A covering problem for hypercubes. In Leslie Pack Kaelbling, editor, *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 1523–1524, Edinburgh, UK, August 2005. Morgan Kaufmann.

Jörg Hoffmann and Bernhard Nebel. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:253–302, 2001.

Subbarao Kambhampati, Craig A. Knoblock, and Qiang Yang. Planning as refinement search: a unified framework for evaluating design tradeoffs in partial-order planning. *Artificial Intelligence*, 76(1–2):167–238, July 1995.

Henry A. Kautz and Bart Selman. Planning as satisfiability. In B. Neumann, editor, *Proceedings of the 10th European Conference on Artificial Intelligence (ECAI'92)*, pages 359–363, Vienna, Austria, August 1992. Wiley.

Henry A. Kautz and Bart Selman. Pushing the envelope: Planning, propositional logic, and stochastic search. In William J. Clancey and Daniel Weld, editors, *Proceedings of the 13th National Conference of the American Association for Artificial Intelligence (AAAI'96)*, pages 1194–1201, Portland, OR, July 1996. MIT Press.

References VII

Elena Kelareva, Olivier Buffet, Jinbo Huang, and Sylvie Thiébaux. Factored planning using decomposition trees. In Manuela Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 1942–1947, Hyderabad, India, January 2007. Morgan Kaufmann.

Craig Knoblock. Automatically generating abstractions for planning. *Artificial Intelligence*, 68(2):243–302, 1994.

Amy L. Lansky and Lise Getoor. Scope and abstraction: Two criteria for localized planning. In S. Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1612–1619, Montreal, Canada, August 1995. Morgan Kaufmann.

Kenneth L. McMillan. Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits. In Gregor von Bochmann and David K. Probst, editors, *Proceedings of the 4th International Workshop on Computer Aided Verification (CAV'92)*, Lecture Notes in Computer Science, pages 164–177. Springer, 1992.

References VIII

Jussi T. Rintanen. A planning algorithm not based on directional search. In A. Cohn, L. Schubert, and S. Shapiro, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 6th International Conference (KR-98)*, pages 953–960, Trento, Italy, May 1998. Morgan Kaufmann.

Jussi Rintanen. Symmetry reduction for SAT representations of transition systems. In Enrico Giunchiglia, Nicola Muscettola, and Dana Nau, editors, *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS'03)*, pages 32–41, Trento, Italy, 2003. Morgan Kaufmann.

Jussi Rintanen. Planning as satisfiability: Heuristics. *Artificial Intelligence*, 193:45–86, 2012.

Earl D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artificial Intelligence*, 5:115135, 1974.

Earl D. Sacerdoti. The nonlinear nature of plans. In *Proceedings of the 4th International Joint Conference on Artificial Intelligence (IJCAI'75)*, pages 206–214, Tiblisi, USSR, September 1975. William Kaufmann.

Martin Suda. Property directed reachability for automated planning. *Journal of Artificial Intelligence Research*, 50:265–319, 2014.

References IX

Álvaro Torralba, Daniel Gnad, Patrick Dubbert, and Jörg Hoffmann. On state-dominance criteria in fork-decoupled search. In Subbarao Kambhampati, editor, *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI'16)*. AAAI Press/IJCAI, 2016.

Håkan L. S. Younes and Reid G. Simmons. VHPOP: versatile heuristic partial order planner. *Journal of Artificial Intelligence Research*, 20:405–430, 2003.