Contrastive Analysis: Heuristic Search Beyond Heuristics

Álvaro Torralba

AALBORG UNIVERSITY

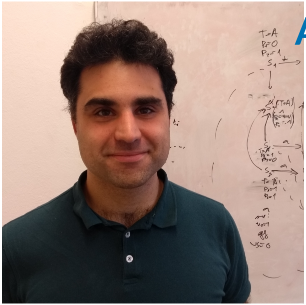
AAAI'23

About Me

Associate Professor

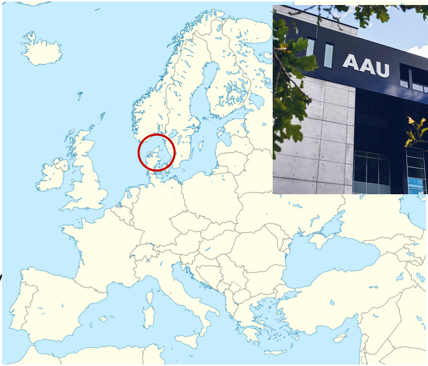
DEIS

Aalborg University (Denmark)



Álvaro Torralba

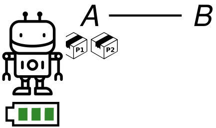
https://homes.cs.aau.dk/alto/



Outline of this Talk

AI Planning in a Nutshell

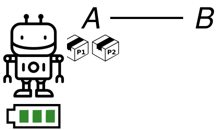
Deliver both packages to location B without depleting the battery:



- $V = \{r, p_1, p_2, b\}$; $D_{p_i} = \{A, B, R\}$, $D_r = \{A, B\}$, $D_b = \{0, 1, 2, 3\}$.
- $A = \{grab(p_i, x), drop(p_i, x), move(x, x')\}$: $pre_{grab(p_i,x)} = \{p_i = x, r = x\}$ and $eff_{grab(p_i,x)} = \{p_i = r\}$.
- $I = \{r = A, p_1 = A, p_2 = A, b = 3\}$
- $G = \{p_1 = B, p_2 = B\}$.
- *cost*: All actions cost 1.
- Some actions consume battery.

AI Planning in a Nutshell

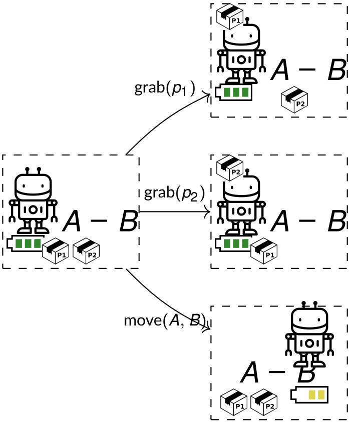
Deliver both packages to location B without depleting the battery:



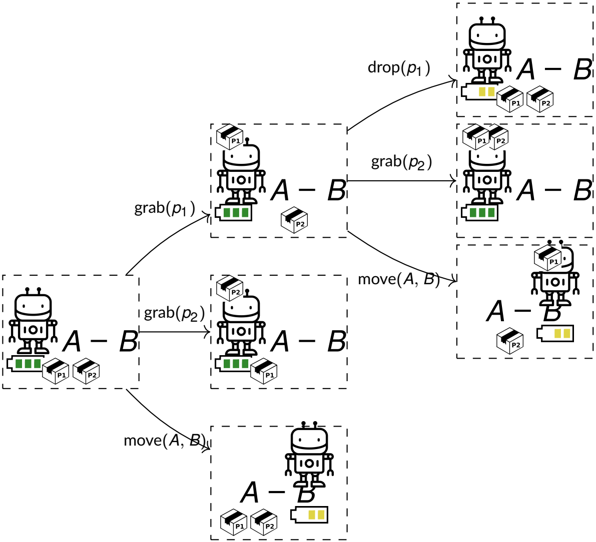
- $V = \{r, p_1, p_2, b\}$; $D_{p_i} = \{A, B, R\}$, $D_r = \{A, B\}$, $D_b = \{0, 1, 2, 3\}$.
- $A = \{grab(p_i, x), drop(p_i, x), move(x, x')\}$: $pre_{grab(p_i,x)} = \{p_i = x, r = x\}$ and $eff_{grab(p_i,x)} = \{p_i = r\}$.
- $I = \{r = A, p_1 = A, p_2 = A, b = 3\}$
- $G = \{p_1 = B, p_2 = B\}$.
- *cost*: All actions cost 1.
- Some actions consume battery.

Plan of minimum cost:
$grab(p_1, A), grab(p_2, A), move(A, B), drop(p_2, B), drop(p_1, B)$

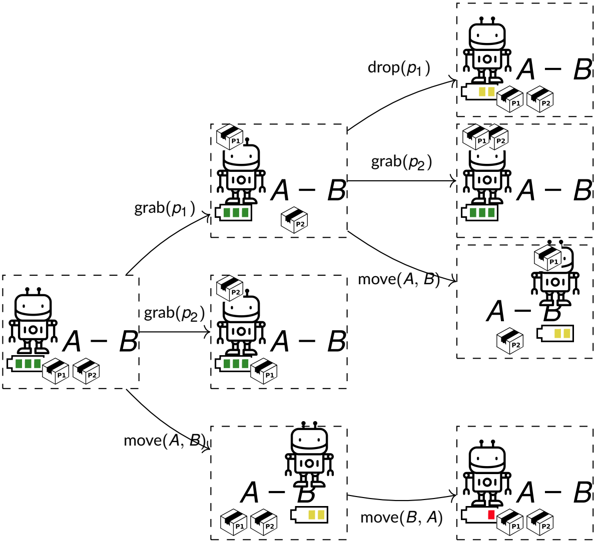
State-Space Search



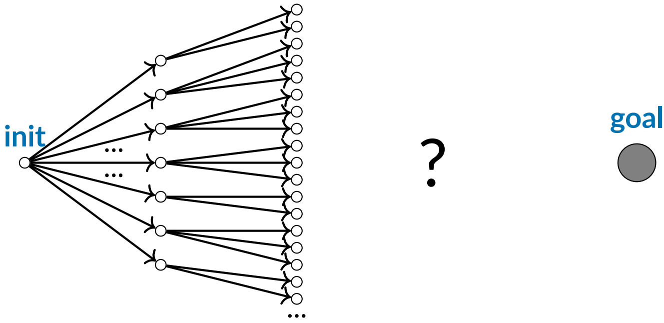
State-Space Search



State-Space Search



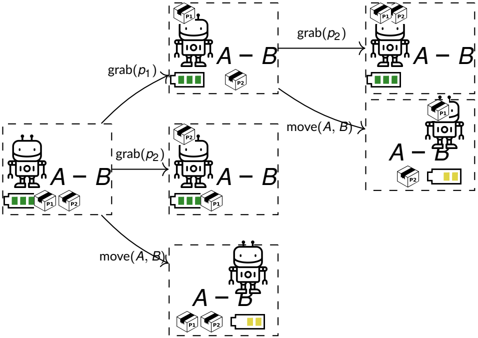
State-Space Explosion



Huge branching factor + exponential in depth

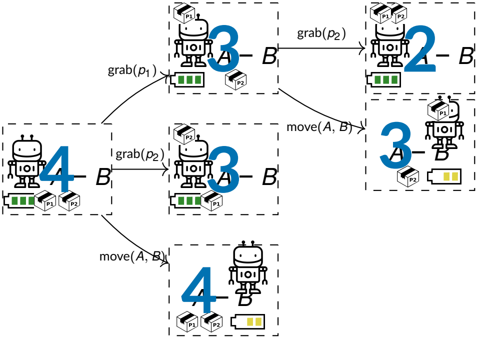
→ **state space *explosion***.

State-Space Search



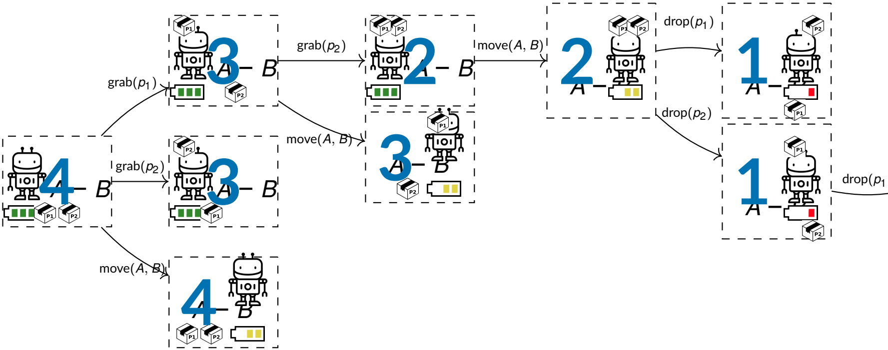
- Guide search with an evaluation function, $h : S \mapsto \mathbb{R}$
- Estimation of the real goal-distance h^*
  $\rightarrow$ derived by inference and/or learning techniques

State-Space Search



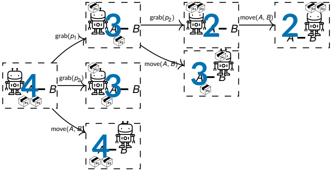
- Guide search with an evaluation function, $h : S \mapsto \mathbb{R}$
- Estimation of the real goal-distance h^*
  $\rightarrow$ derived by inference and/or learning techniques

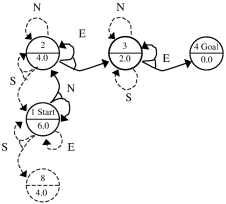
State-Space Search



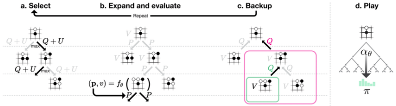
- Guide search with an evaluation function, $h : S \mapsto \mathbb{R}$
- Estimation of the real goal-distance h^*
  $\rightarrow$ derived by inference and/or learning techniques

Search + Heuristic: A common pattern





A^* (Shortest path/Classical Planning)   LAO* (solving MDPs)[1]

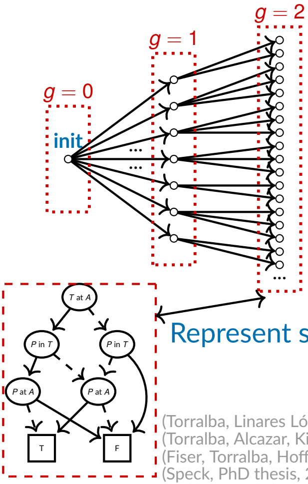


Monte Carlo Tree Search (online decision making, e.g. AlphaGo)[1]

[1] Images taken from (Hansen and Zilberstein, 2001) and (Silver et al. 2016).

Reasoning with Sets of States (Topic for Another Day)

Symbolic Search:

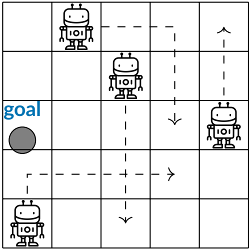


?

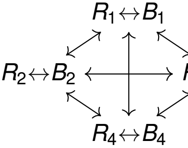
Represent sets as BDDs

(Torralba, Linares López, Borrajo IJCAI'16)
(Torralba, Alcazar, Kissmann, Edelkamp, AIJ'17)
(Fiser, Torralba, Hoffmann, AAAI'22)
(Speck, PhD thesis, 2022)

Decoupled Search:



Causal Graph:



(Gnad, Hoffmann, AIJ'18)
(Gnad, Torralba, Fiser, ICAPS'22)
(Gnad, PhD thesis, 2021)

What's a Heuristic Anyway?

Heuristics (and Search) for Domain-independent Planning

What's a Heuristic Anyway?

Heuristics (and Search) for Domain-independent Planning

The way to guide the search

What's a Heuristic Anyway?

Heuristics (and Search) for Domain-independent Planning

~~The way to guide the search~~
A way to guide the search

What's a Heuristic Anyway?

Heuristics (and Search) for Domain-independent Planning

~~The way to guide the search~~
A way to guide the search

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.
$h(s)$ estimates goal distance $h^*(s)$

What's a Heuristic Anyway?

Heuristics (and Search) for Domain-independent Planning

~~The way to guide the search~~
A way to guide the search

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.
$h(s)$ estimates goal distance $h^*(s)$

Properties of Heuristics:

- Admissible: $h(s) \leq h^*(s)$
- Consistent: $h(s) - h(t) \leq c(s,t)$

What's Great About Heuristics

- Very effective method for planning!

What's Great About Heuristics

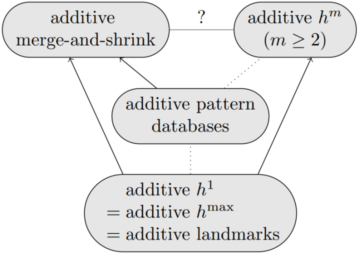
- Very effective method for planning!
- Provide a Theoretical Framework to Analyze Search-based Planning Algorithms

What's Great About Heuristics

- Very effective method for planning!
- Provide a Theoretical Framework to Analyze Search-based Planning Algorithms
- Disantangle search algorithm from the source of information

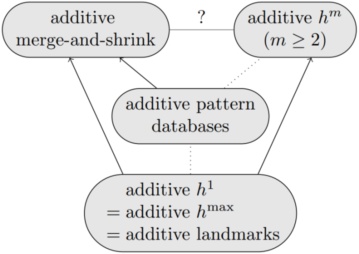
Disantangle search algorithm and heuristics

- define heuristics, analyze their properties(?), and compare them(?) without considering how search algorithms will use them:



Disantangle search algorithm and heuristics

- define heuristics, analyze their properties(?), and compare them(?) without considering how search algorithms will use them:



- We can compare search algorithms without considering what heuristic is being used

  $\rightarrow A^*$ is optimally efficient!

DXBB Algorithms

UDXBB: Unidirectional, Deterministic, Expansion-based, Black Box

Access to the state space Θ only via node expansions

Additionally the algorithm is given an admissible heuristic function h

h2 A10

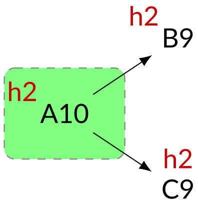
→ The algorithm does not have access to the task description. The heuristic is its only **source of information** to guide the search

DXBB Algorithms

***UDXBB*: Unidirectional, Deterministic, Expansion-based, Black Box**

Access to the state space Θ only via node expansions

Additionally the algorithm is given an admissible heuristic function h



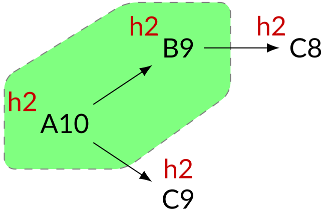
$\rightarrow$ The algorithm does not have access to the task description. The heuristic is its only **source of information** to guide the search

DXBB Algorithms

UDXBB: Unidirectional, Deterministic, Expansion-based, Black Box

Access to the state space Θ only via node expansions

Additionally the algorithm is given an admissible heuristic function *h*



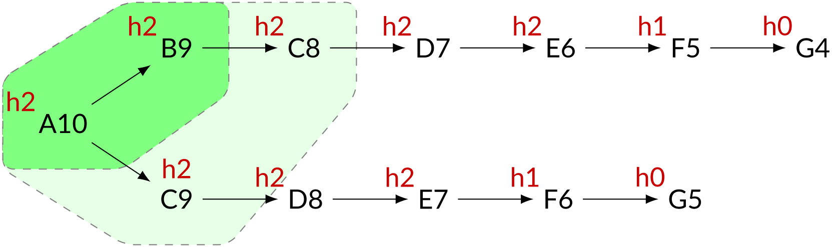
→ The algorithm does not have access to the task description. The heuristic is its only **source of information** to guide the search

DXBB Algorithms

***UDXBB*: Unidirectional, Deterministic, Expansion-based, Black Box**

Access to the state space Θ only via node expansions

Additionally the algorithm is given an admissible heuristic function h



$\rightarrow$ The algorithm does not have access to the task description. The heuristic is its only **source of information** to guide the search

A^* is Optimally Efficient (Dechter and Pearl, 1985)

A^*: Expand nodes based on f-value: $f(n_s) = g(n_s) + h(s)$

A^* is Optimally Efficient (Dechter and Pearl, 1985)

A^*: Expand nodes based on f-value: $f(n_s) = g(n_s) + h(s)$

*Generalized BF Search Strategies and the Optimality of A** 531

Class of Algorithms

Domain of Problem Instances		Admissible if $h \leq h^*$ A_{ad}	Globally Compatible with A^* A_{gc}	Best-First A_{bf}
	Admissible I_{AD}	A^{**} is 3-optimal No 2-optimal exists	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists
	Admissible and non pathological $I_{\bar{AD}}$	A^* is 2-optimal No 1-optimal exists	A^* is 0-optimal	A^* is 0-optimal
	Consistent I_{CON}	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists	A^* is 1-optimal No 0-optimal exists
	Consistent nonpathological $I_{\bar{CON}}$	A^* is 0-optimal	A^* is 0-optimal	A^* is 0-optimal

A^* is Optimally Efficient (Dechter and Pearl, 1985)

A^* is 1-optimal on consistent instances

Let N be the set of states expanded by any admissible *UDXBB* algorithm, then **there exists a tie-breaking** of A^* that expands a **subset** of N.

A^* is Optimally Efficient (Dechter and Pearl, 1985)

A^* is 1-optimal on consistent instances

Let N be the set of states expanded by any admissible *UDXBB* algorithm, then **there exists a tie-breaking** of A^* that expands a **subset** of N.

Consistent Heuristic: $h(s) - h(t) \leq c(s, t)$

A^* is Optimally Efficient (Dechter and Pearl, 1985)

A^* is 1-optimal on consistent instances

Let N be the set of states expanded by any admissible *UDXBB* algorithm, then **there exists a tie-breaking** of A^* that expands a **subset** of N.

Consistent Heuristic: $h(s) - h(t) \leq c(s, t)$

1. Nodes are expanded with their optimal g-value (no re-expansions)

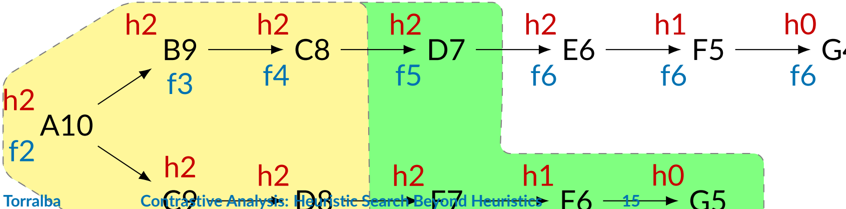
A* is Optimally Efficient (Dechter and Pearl, 1985)

A* is 1-optimal on consistent instances

Let N be the set of states expanded by any admissible *UDXBB* algorithm, then **there exists a tie-breaking** of A* that expands a **subset** of N.

Consistent Heuristic: $h(s) - h(t) \leq c(s, t)$

1. Nodes are expanded with their optimal g-value (no re-expansions)
2. Must-expand nodes: $f(n) < f^*$



Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

$$S \mapsto 2^A$$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}^+_0 \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

$$S \mapsto 2^A$$

- Fact Landmarks $S \mapsto 2^F$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

$$S \mapsto 2^A$$

- Fact Landmarks $S \mapsto 2^F$
- Disjunctive-Action Landmarks $S \mapsto 2^{2^A}$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

$$S \mapsto 2^A$$

- Fact Landmarks $S \mapsto 2^F$
- Disjunctive-Action Landmarks $S \mapsto 2^{2^A}$
- Heuristics with Uncertainty $S \mapsto PMF$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

$$S \mapsto 2^A$$

- Fact Landmarks $S \mapsto 2^F$
- Disjunctive-Action Landmarks $S \mapsto 2^{2^A}$
- Heuristics with Uncertainty $S \mapsto PMF$
- Operator-counting constraints, Operator-mutexes, ...

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

You cannot reduce everything to a number!

- Stubborn sets: identify a sub-set of actions such that at least one of them starts an optimal plan

$$S \mapsto 2^A$$

- Preferred operators: identify a sub-set of actions such that hopefully one of them starts an (optimal?) plan

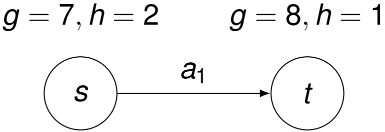
$$S \mapsto 2^A$$

- Fact Landmarks $S \mapsto 2^F$
- Disjunctive-Action Landmarks $S \mapsto 2^{2^A}$
- Heuristics with Uncertainty $S \mapsto PMF$
- Operator-counting constraints, Operator-mutexes, …

$\rightarrow$Many of these are reduced to heuristics, is this optimally efficient?

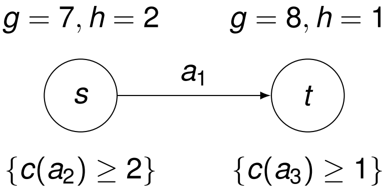
Are Operator Counting Constraints Optimally Efficient?

Part of a search tree on a task with optimal solution cost of 10:



Are Operator Counting Constraints Optimally Efficient?

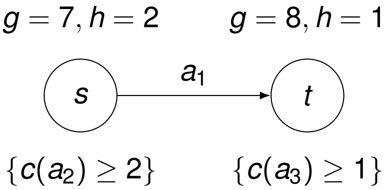
Part of a search tree on a task with optimal solution cost of 10:



A^* must expand t

Are Operator Counting Constraints Optimally Efficient?

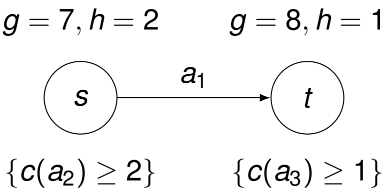
Part of a search tree on a task with optimal solution cost of 10:



A^* must expand *t*, but an optimally efficient algorithm does not have to

Are Operator Counting Constraints Optimally Efficient?

Part of a search tree on a task with optimal solution cost of 10:



A^* must expand t, but an optimally efficient algorithm does not have to

$\rightarrow$Well, perhaps this is just under inconsistent operator counting constraints ...

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

Limitations of Heuristics

Definition: A heuristic is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

States are evaluated independently of each other

Limitations of Heuristics

Definition: A heuristic is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

States are evaluated independently of each other

- Symmetry Detection: $S \times S \mapsto \{0, 1\}$
- Dominance Analysis: $S \times S \mapsto \{0, 1\}$
- Quantitative Dominance Analysis: $S \times S \mapsto \mathbb{R} \cup \{-\infty\}$
- Novelty Pruning: $S \times 2^S \mapsto \{0, 1\}$
- Novelty Heuristics: $S \times 2^S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$

Limitations of Heuristics

Definition: A **heuristic** is a function $h : S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$.

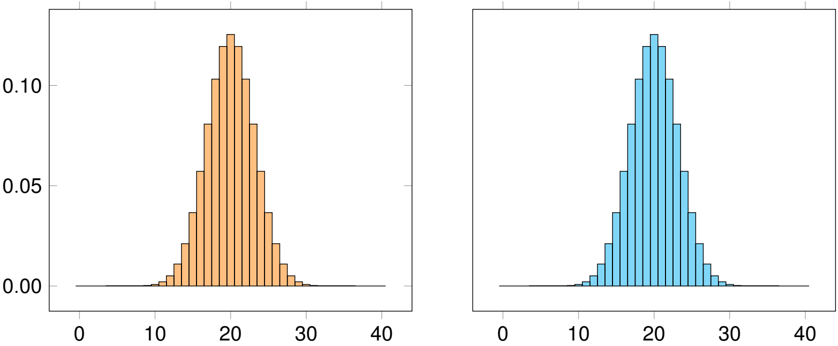
States are evaluated independently of each other

- Symmetry Detection: $S \times S \mapsto \{0, 1\}$
- Dominance Analysis: $S \times S \mapsto \{0, 1\}$
- Quantitative Dominance Analysis: $S \times S \mapsto \mathbb{R} \cup \{-\infty\}$
- Novelty Pruning: $S \times 2^S \mapsto \{0, 1\}$
- Novelty Heuristics: $S \times 2^S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$

- Contrastive Analysis: $S \times S \mapsto ?$

Heuristics with Uncertainty

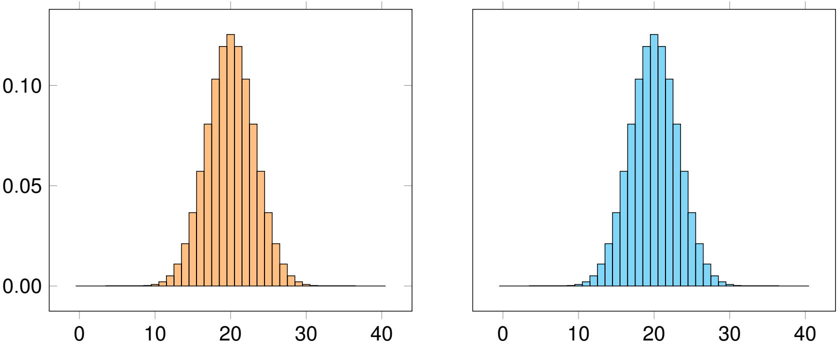
Consider heuristic functions that return a probability distribution (?):



Multi-Valued Pattern Databases (?)

Heuristics with Uncertainty

Consider heuristic functions that return a probability distribution (?):

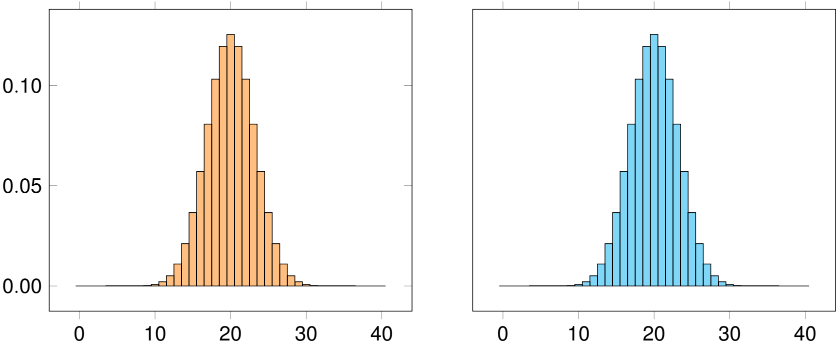


Multi-Valued Pattern Databases (?)

- Heuristics evaluate states assuming independence!
  →rarely the case for states close in the search tree

Heuristics with Uncertainty

Consider heuristic functions that return a probability distribution (?):



Multi-Valued Pattern Databases (?)

- Heuristics evaluate states assuming independence!
  →rarely the case for states close in the search tree
- We cannot express $h(blue) \leq h(orange)$

In the Rest of This Talk

Assumption of most search algorithms: Evaluate states independently of each other

In the Rest of This Talk

Assumption of most search algorithms: Evaluate states independently of each other

Break this assumption by **directly comparing states**:
→ **New source of information**

In the Rest of This Talk

Assumption of most search algorithms: Evaluate states independently of each other

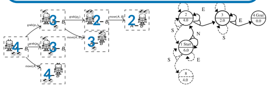
Break this assumption by **directly comparing states**:
$\rightarrow$ **New source of information**

Inference

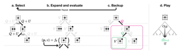
Find information

- Automatic: on any domain!
- Polynomial time
- Reliable (safe to use)

Exploitation



A* (Shortest path/Classical Planning) LAO* (solving MDPs)[1]



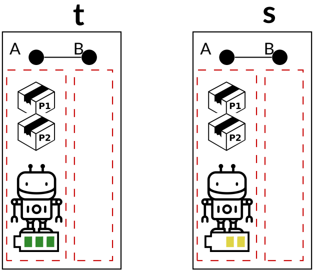
Monte Carlo Tree Search (online decision making, e.g. AlphaGo)[1]

Use information

- Re-design state-space search
- Theory: Optimally efficient algorithms!
- Practice: Balance inference/search effort

Dominance (Torralba, Hoffmann, IJCAI'15)

Is *t* at least as good as *s*?

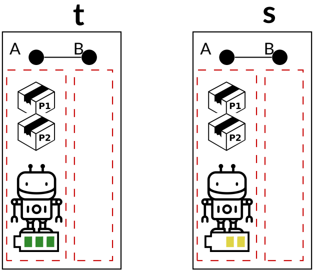


t is at least as good as s

Source of Information: $S \times S \mapsto \{0, 1\}$

Dominance (Torralba, Hoffmann, IJCAI'15)

Is t at least as good as s?



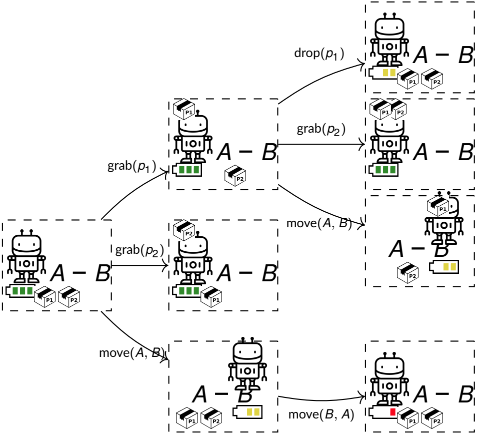
t is at least as good as s

Source of Information: $S \times S \mapsto \{0, 1\}$

Definition (Dominance Relation). *binary relation $\preceq$ on S such that $s \preceq t$ (t **dominates** s) only if t **is at least as good as** s, i.e., $h^*(s) \geq h^*(t)$.*

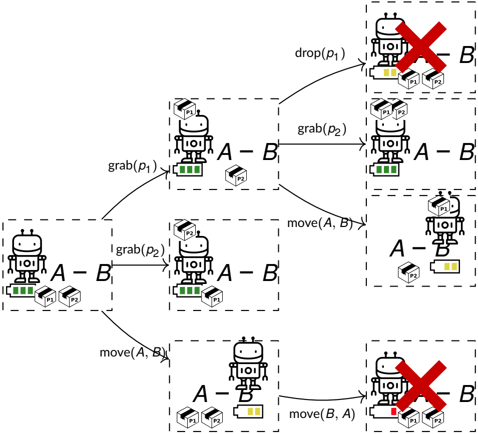
Using a Dominance Relation: Dominance Pruning

Prune a search node n_s if there exists another n_t that dominates it: $g(n_t) \leq g(n_s)$ and $s \preceq t$



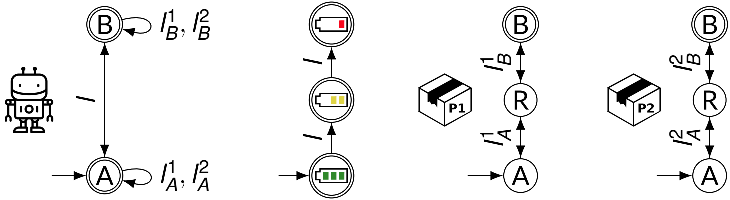
Using a Dominance Relation: Dominance Pruning

Prune a search node n_s if there exists another n_t that dominates it: $g(n_t) \leq g(n_s)$ and $s \preceq t$



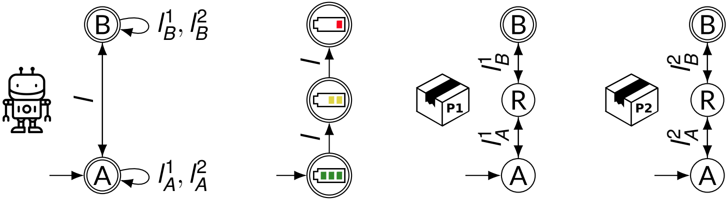
Inference of Dominance (?)

1. Consider a partition of the problem



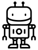
Inference of Dominance (?)

1. Consider a partition of the problem



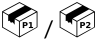
2. Compute coarsest **label-dominance relation** such that:

$$s \preceq_i t \implies \left(s \in S_i^G \vee \neg t \in S_i^G \text{ and } \forall_{s \xrightarrow{l} s'} \exists_{t \xrightarrow{l'} t'} s' \preceq_i t' \wedge l \preceq_i^L l\right)$$



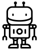
	at A	at B
at A	⊤	⊥
at B	⊥	⊤

			
	⊤	⊤	⊤
	⊥	⊤	⊤
	⊥	⊥	⊤

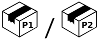


	at A	in R	at B
at A	⊤	⊤	⊤
in R	⊥	⊤	⊤
at B	⊥	⊥	⊤

Combining the Partitions



	at A	at B
at A	$\top$	$\bot$
at B	$\bot$	$\top$



	at A	in R	at B				
at A	$\top$	$\top$	$\top$		$\top$	$\top$	$\top$
in R	$\bot$	$\top$	$\top$		$\bot$	$\top$	$\top$
at B	$\bot$	$\bot$	$\top$		$\bot$	$\bot$	$\top$

A state dominates another iff it dominates in every aspect:

$$s \preceq t \text{ iff } s_i \preceq_i t_i \text{ for all } i.$$

For example:

Combining the Partitions

	at A	at B
at A	$\top$	$\bot$
at B	$\bot$	$\top$

/

	at A	in R	at B				
at A	$\top$	$\top$	$\top$		$\top$	$\top$	$\top$
in R	$\bot$	$\top$	$\top$		$\bot$	$\top$	$\top$
at B	$\bot$	$\bot$	$\top$		$\bot$	$\bot$	$\top$

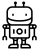
A state dominates another iff it dominates in every aspect:

$$s \preceq t \text{ iff } s_i \preceq_i t_i \text{ for all } i.$$

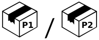
For example:

- A A A $\preceq$ A A B
  because $A \preceq_1 A$, $A \preceq_2 A$, $A \preceq_3 B$, and $\preceq_4$.

Combining the Partitions



	at A	at B
at A	$\top$	$\bot$
at B	$\bot$	$\top$



	at A	in R	at B
at A	$\top$	$\top$	$\top$
in R	$\bot$	$\top$	$\top$
at B	$\bot$	$\bot$	$\top$

	$\top$	$\top$	$\top$
	$\bot$	$\top$	$\top$
	$\bot$	$\bot$	$\top$

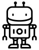
A state dominates another iff it dominates in every aspect:

$$s \preceq t \text{ iff } s_i \preceq_i t_i \text{ for all } i.$$

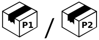
For example:

- A A A $\preceq$ A A B
  because $A \preceq_1 A$, $A \preceq_2 A$, $A \preceq_3 B$, and $\preceq_4$.
- A A A $\not\preceq$ B A B
  because $A \not\preceq_1 B$.

From Dominance Pruning to Dominance Analysis



	at A	at B
at A	⊤	⊥
at B	⊥	⊤

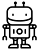


	at A	in R	at B
at A	⊤	⊤	⊤
in R	⊥	⊤	⊤
at B	⊥	⊥	⊤

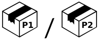
			
	⊤	⊤	⊤
	⊥	⊤	⊤
	⊥	⊥	⊤

This information is extremely interesting. **Useful for a lot of things beyond pruning search**!!

From Dominance Pruning to Dominance Analysis



	at A	at B
at A	$\top$	$\bot$
at B	$\bot$	$\top$



	at A	in R	at B
at A	$\top$	$\top$	$\top$
in R	$\bot$	$\top$	$\top$
at B	$\bot$	$\bot$	$\top$

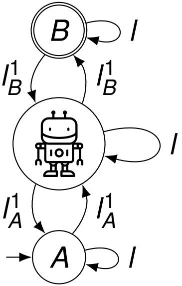
			
	$\top$	$\top$	$\top$
	$\bot$	$\top$	$\top$
	$\bot$	$\bot$	$\top$

This information is extremely interesting. **Useful for a lot of things beyond pruning search**!!

- **Dominance:** Comparing things better than others
- **Analysis:** Uses inference and/or learning to compare states according to “estimated” goal distance. $\rightarrow$In contrast to, e.g., dominance in multi-objective and/or decoupled search where states dominate each others in term of g-value

Identifying Irrelevant Actions (?)

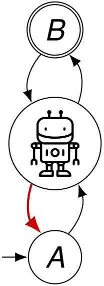
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

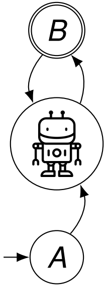
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

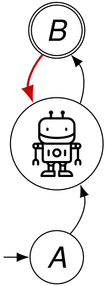
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

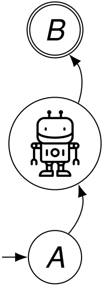
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

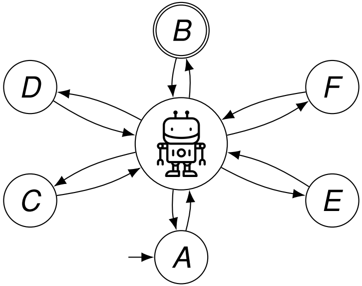
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

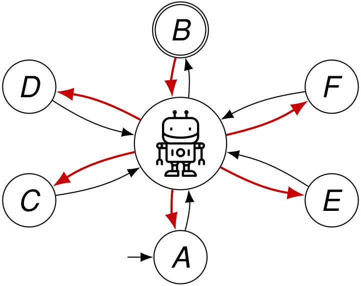
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

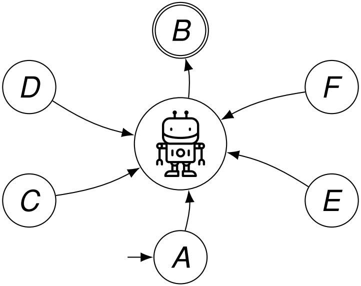
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

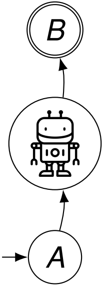
Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Identifying Irrelevant Actions (?)

Irrelevant Actions: those that can be removed while preserving an optimal plan



: $A \preceq T \preceq B$

Analyze the Optimal Efficiency

- A^*: canonical choice for solving shortest path problems
- A^* is **optimally efficient** in node expansions (?)

- **Dominance pruning** methods $\rightarrow$ new source of information!

Analyze the Optimal Efficiency

- A^*: canonical choice for solving shortest path problems
- A^* is **optimally efficient** in node expansions (?)

- **Dominance pruning** methods $\rightarrow$ new source of information!

- A^* with dominance pruning (A^*_{pr}):
  - Expand nodes based on f-value: $f(n_s) = g(n_s) + h(s)$
  - Prune any node that can be pruned
- Is this a good choice?

Results of Optimal Efficiency Analysis

- A^*_{pr} is #-optimally efficient on consistent instances over $UDXBB_{pr}$ algorithms

Results of Optimal Efficiency Analysis

- A^*_{pr} is #-optimally efficient on consistent instances over $UDXBB_{pr}$ algorithms
- Consistent instances:
  1. Consistent heuristic
  2. Dominance relation is a transitive cost-simulation relation
  3. Heuristic and dominance relation are consistent with each other

Results of Optimal Efficiency Analysis

- A^*_{pr} is #-optimally efficient on consistent instances over $UDXBB_{pr}$ algorithms
- Consistent instances:
  1. Consistent heuristic
  2. Dominance relation is a transitive cost-simulation relation
  3. Heuristic and dominance relation are consistent with each other
- **No access to $\preceq$**: can only use dominance for pruning nodes that are worse in g and h value

Open Question: What is the best way that any algorithm can leverage dominance relations?

Metamorphic Testing

- Detect mistakes of a policy by comparing behaviour on dominated/dominating states

J. Eisenhut, A. Torralba, M. Christakis, and J. Hoffmann,
Automatic Metamorphic Test Oracles for Action-Policy Testing
Tuesday, July, 11, 16:00-17:00

Quantitative Dominance (?)

By how much t dominates s?

Source of Information: $S \times S \mapsto \{0, 1\}$

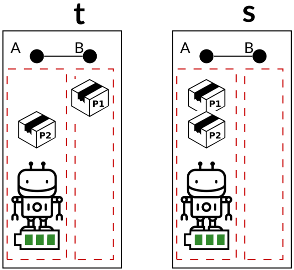
Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$

Quantitative Dominance (?)

By how much t dominates s?

Source of Information: $S \times S \mapsto \{0, 1\}$

Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$

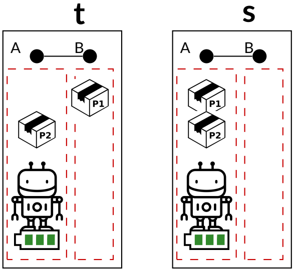


Quantitative Dominance (?)

By how much t dominates s?

Source of Information: $S \times S \mapsto \{0, 1\}$

Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$



$D(s, t) = 2$:

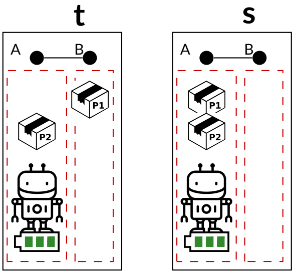
t is strictly closer to the goal than s by 2 actions (grab and drop p_1).

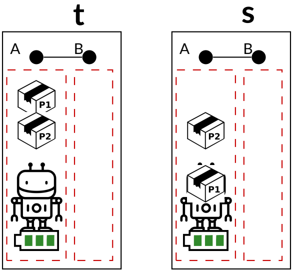
Quantitative Dominance (?)

By how much t dominates s?

Source of Information: $S \times S \mapsto \{0, 1\}$

Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$





$D(s, t) = 2$:

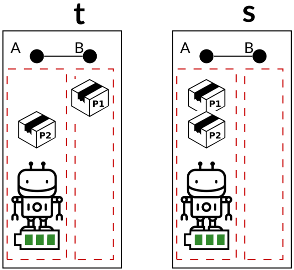
t is strictly closer to the goal than s by 2 actions (grab and drop p_1).

Quantitative Dominance (?)

By how much t dominates s?

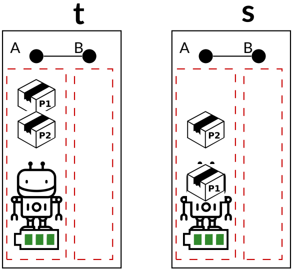
Source of Information: $S \times S \mapsto \{0, 1\}$

Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$



$D(s, t) = 2$:

t is strictly closer to the goal than s by 2 actions (grab and drop p_1).



$D(s, t) = -1$

t is not farther than 1 action to the goal than s (grab p_1).

Quantitative Dominance

Dominance Function: $D(s, t) \leq h^*(s) - h^*(t)$

Quantitative Dominance

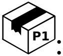
Dominance Function: $D(s,t) \leq h^*(s) - h^*(t)$

$$D(s,t) = \begin{cases} C & t \text{ is strictly closer to the goal than s (by at least } C\text{)} \\ 0 & t \text{ is at least as close as } s \\ -C & \text{t is at most C units of cost farther than } s \\ -\infty & \text{we know nothing} \end{cases}$$

Quantitative Dominance

Dominance Function: $D(s,t) \leq h^*(s) - h^*(t)$

$$D(s,t) = \begin{cases} C & t \text{ is strictly closer to the goal than s (by at least } C) \\ 0 & t \text{ is at least as close as } s \\ -C & \text{t is at most C units of cost farther than } s \\ -\infty & \text{we know nothing} \end{cases}$$



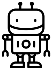
: $D_P(A,T) = D_P(T,B) = +1$

Quantitative Dominance

Dominance Function: $D(s,t) \leq h^*(s) - h^*(t)$

$$D(s,t) = \begin{cases} C & t \text{ is strictly closer to the goal than s (by at least } C) \\ 0 & t \text{ is at least as close as } s \\ -C & \text{t is at most C units of cost farther than } s \\ -\infty & \text{we know nothing} \end{cases}$$

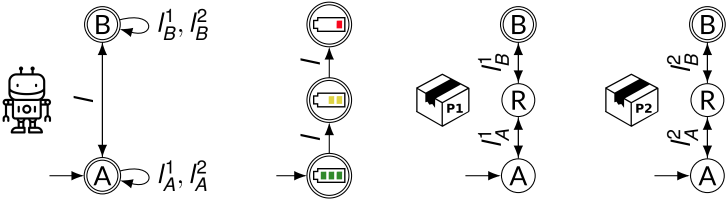
: $D_P(A,T) = D_P(T,B) = +1$



: $D_T(A,B) = D_T(B,A) = -1$

Inference of Quantitative Dominance

1. Consider a partition of the problem



2. Compute maximum fix point **label-dominance function** such that:

$$D_i(s,t) \leq \min_{s \xrightarrow{\ell} s'} \max_{u \xrightarrow{\ell'} u'} D_i(s',u') - h^{\tau}(t,u) + c(\ell) - c(\ell') + \sum_{j \neq i} \mathcal{D}_j^L(\ell,\ell')$$

	at A	at B
at A	0	$-\infty$
at B	$-\infty$	0

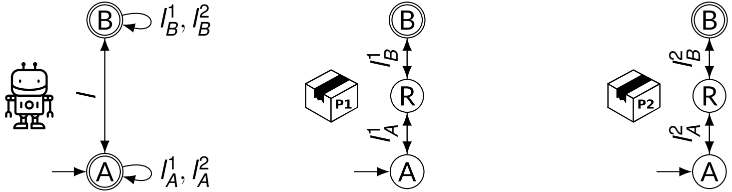
			
	0	0	0
	$-\infty$	0	0
	$-\infty$	$-\infty$	0

 / 

	at A	in R	at B
at A	0	1	2
in R	$-\infty$	0	1
at B	$-\infty$	$-\infty$	0

Inference of Quantitative Dominance

1. Consider a partition of the problem



2. Compute maximum fix point **label-dominance function** such that:

$$D_i(s,t) \leq \min_{s \xrightarrow{\ell} s'} \max_{u \xrightarrow{\ell'} u'} D_i(s',u') - h^{\tau}(t,u) + c(\ell) - c(\ell') + \sum_{j \neq i} \mathcal{D}_j^L(\ell,\ell')$$

	at A	at B
at A	0	−1
at B	−1	0

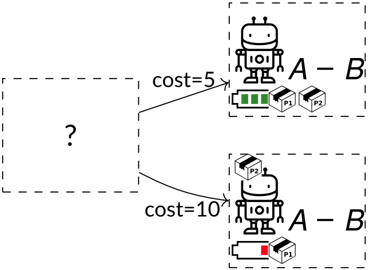
	at A	in R	at B
at A	0	1	2
in R	−3	0	1
at B	−5	−3	0

Quantitative Dominance Pruning

Prune n_s if there exists n_t s.t.

Qualitative $g(n_t) \leq g(n_s)$ and $s \preceq t$

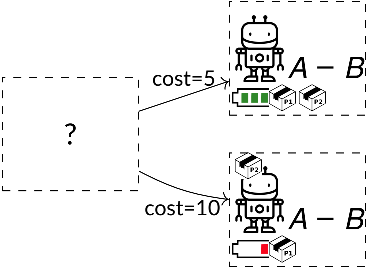
Quantitative



Quantitative Dominance Pruning

Prune n_s if there exists n_t s.t.

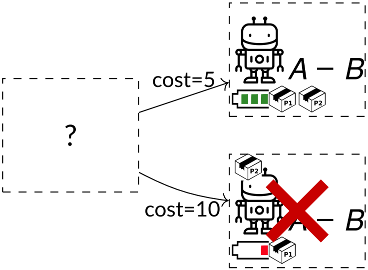
Qualitative	$g(n_t) \leq g(n_s)$ and $s \preceq t$
Quantitative	$D(s,t) + g(n_s) - g(n_t) \geq 0$ if $D(s,t) \geq 0$



Quantitative Dominance Pruning

Prune n_s if there exists n_t s.t.

Qualitative	$g(n_t) \leq g(n_s)$ and $s \preceq t$
Quantitative	$D(s,t) + g(n_s) - g(n_t) \geq 0$ if $D(s,t) \geq 0$
	$D(s,t) + g(n_s) - g(n_t) > 0$ if $D(s,t) < 0$

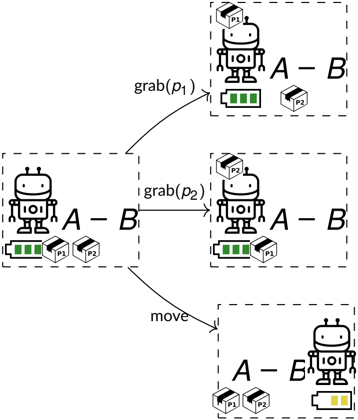


Action Selection Pruning

If $s \xrightarrow{a} s'$ and $D(s, s') \geq c(a)$ then ***a* starts an optimal plan** from s.

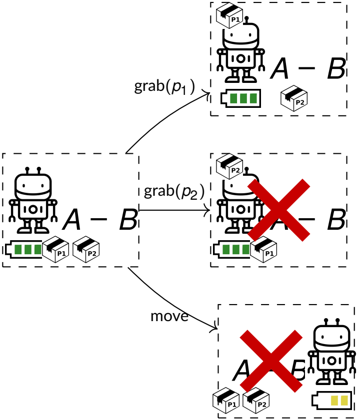
Action Selection Pruning

If $s \xrightarrow{a} s'$ and $D(s, s') \geq c(a)$ then *a* **starts an optimal plan** from s.



Action Selection Pruning

If $s \xrightarrow{a} s'$ and $D(s, s') \geq c(a)$ then *a* **starts an optimal plan** from s.

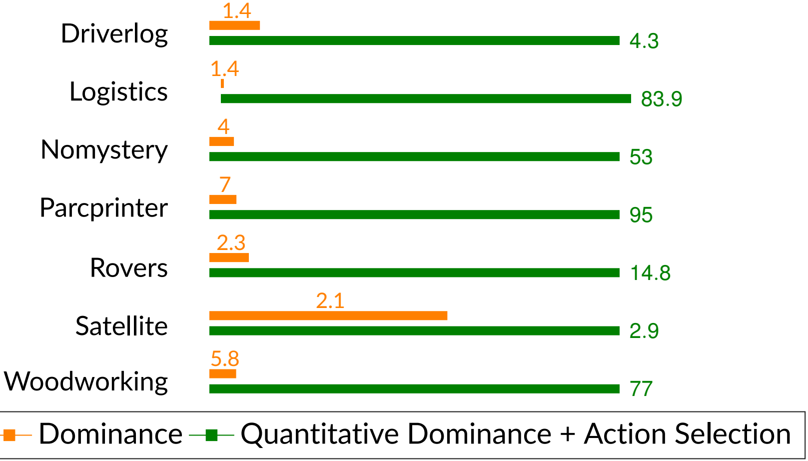


- Prune every other successor
- Reduce branching factor to 1!

→Branch only over move actions!

Dramatic Pruning Power!

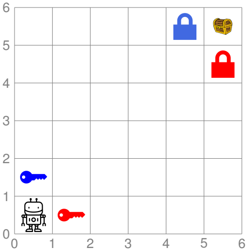
Factor of search space reduction to find optimal plan over A^* search with LM-cut (10 = one order of magnitude) in selected domains:

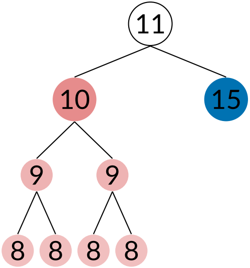


Novelty Pruning

Novelty (??): Compare each state against previously seen states to prioritize most novel states:

1. States that have a new fact that no other state had.
2. States that have a pair of facts that no other state had.
3. . . .





→ Extremely successful at diversifying search (e.g. also in Atari games (?))!

Novelty

The **novelty of s** $N(s)$ is defined to be the **size of the smallest fact set it produces for the first time**.

Novelty

The **novelty of s** $N(s)$ is defined to be the **size of the smallest fact set it produces for the first time**.

IW(K): Breadth first search, pruning all s with $N(s) > k$

- Polynomial time
- No guidance towards the goal
- Good for exploration/achieving single goal facts

Novelty

The **novelty of s** $N(s)$ is defined to be the **size of the smallest fact set it produces for the first time**.

IW(K): Breadth first search, pruning all s with $N(s) > k$

- Polynomial time
- No guidance towards the goal
- Good for exploration/achieving single goal facts

Novelty Heuristics:

- Combine the definition of novelty with heuristics
- **State of the art** in satisficing planning

Novelty

The **novelty of s** $N(s)$ is defined to be the **size of the smallest fact set it produces for the first time**.

IW(K): Breadth first search, pruning all s with $N(s) > k$

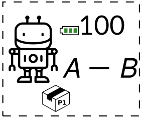
- Polynomial time
- No guidance towards the goal
- Good for exploration/achieving single goal facts

Novelty Heuristics:

- Combine the definition of novelty with heuristics
- **State of the art** in satisficing planning

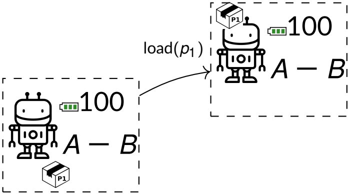
But, why is novelty so good?

Example IW(1)



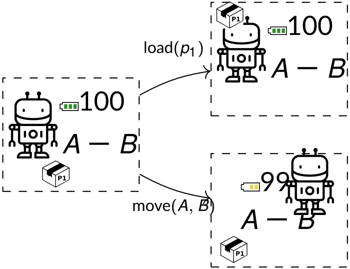
A	B	A	B	R	100	99	98	97
x		x			x			

Example IW(1)



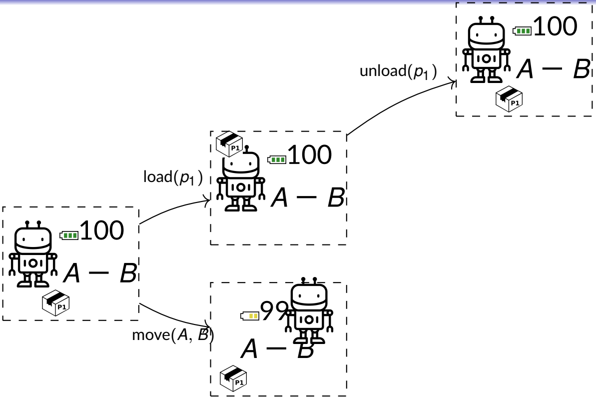
A	B	A	B	R	100	99	98	97
x		x		x	x			

Example IW(1)



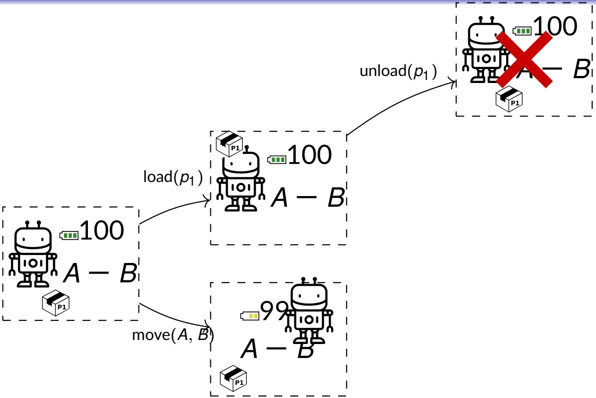
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x		

Example IW(1)



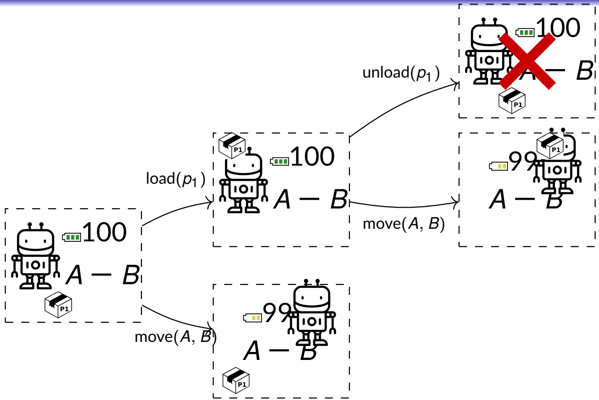
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x		

Example IW(1)



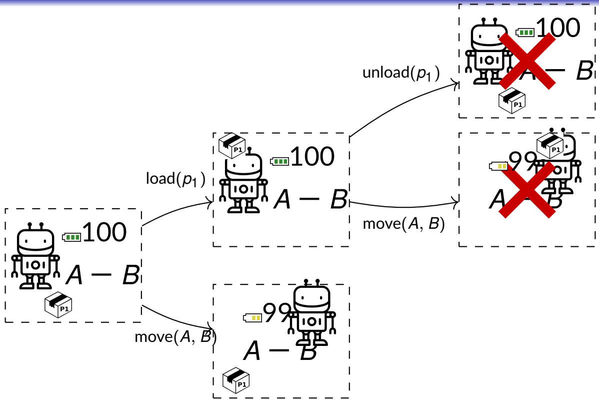
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x		

Example IW(1)



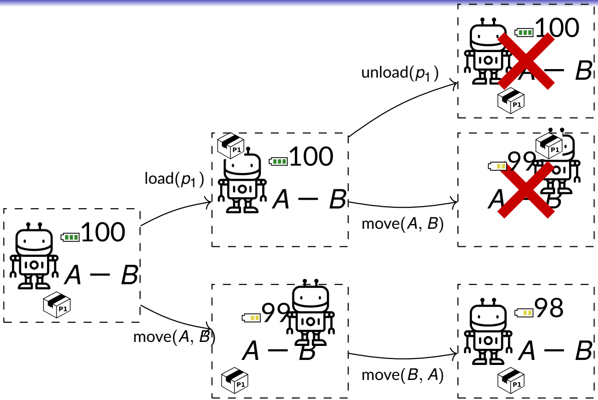
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x		

Example IW(1)



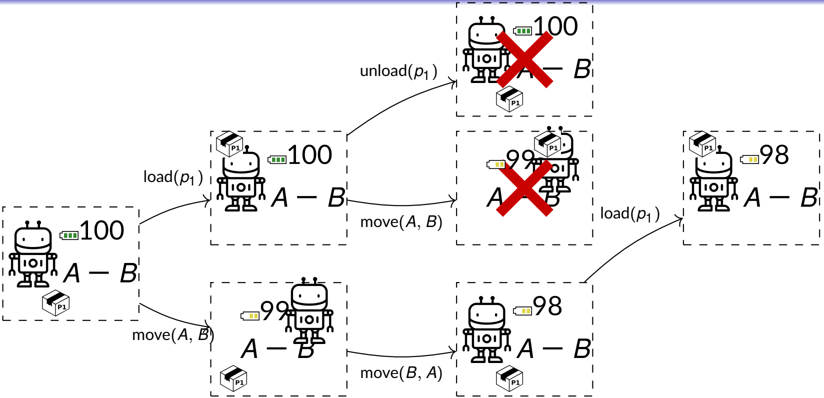
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x		

Example IW(1)



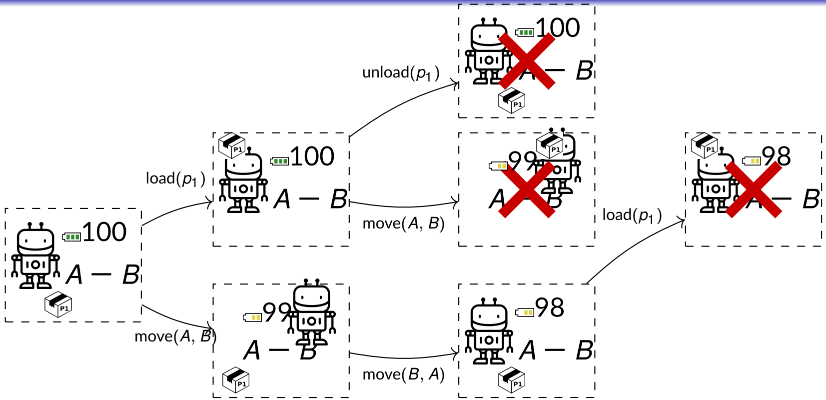
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x	x	

Example IW(1)



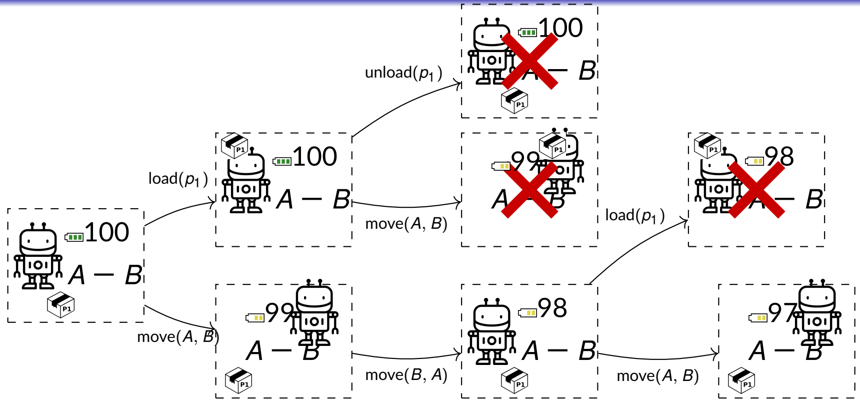
A	B	A	B	R	100	99	98	97
x	x	x		x	x	x	x	

Example IW(1)



A	B	A	B	R	100	99	98	97
x	x	x		x	x	x	x	

Example IW(1)



A	B	A	B	R	100	99	98	97
x	x	x		x	x	x	x	x

So, What Novelty and Dominance Have In Common?

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

Source of Information: $S \times 2^S \mapsto \{0, 1\}$

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

Source of Information: $S \times 2^S \mapsto \{0, 1\}$

Safe dominance pruning $\exists t \in \mathcal{T}\ \forall v \in V \quad s[v] \preceq t[v]$

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

Source of Information: $S \times 2^S \mapsto \{0, 1\}$

Safe dominance pruning	$\exists t \in \mathcal{T}\ \forall v \in V$	$s[v] \preceq t[v]$
Novelty IW(1) pruning	$\forall v \in V\ \exists t \in \mathcal{T}$	$s[v] = t[v]$

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

Source of Information: $S \times 2^S \mapsto \{0, 1\}$

Safe dominance pruning $\quad \exists t \in \mathcal{T} \; \forall v \in V \quad s[v] \preceq t[v]$

Novelty IW(1) pruning $\quad \forall v \in V \; \exists t \in \mathcal{T} \quad s[v] = t[v]$

$\rightarrow$ **Novelty can be interpreted as (unsafe/inadmissible) dominance**

$$\exists t \in \mathcal{T} \; h^*(t) \leq h^*(s)$$

So, What Novelty and Dominance Have In Common?

Both compare new states s against all previously seen states $\mathcal{T}$

Source of Information: $S \times 2^S \mapsto \{0, 1\}$

Safe dominance pruning	$\exists t \in \mathcal{T}\ \forall v \in V \quad s[v] \preceq t[v]$
Novelty IW(1) pruning	$\forall v \in V\ \exists t \in \mathcal{T} \quad s[v] = t[v]$

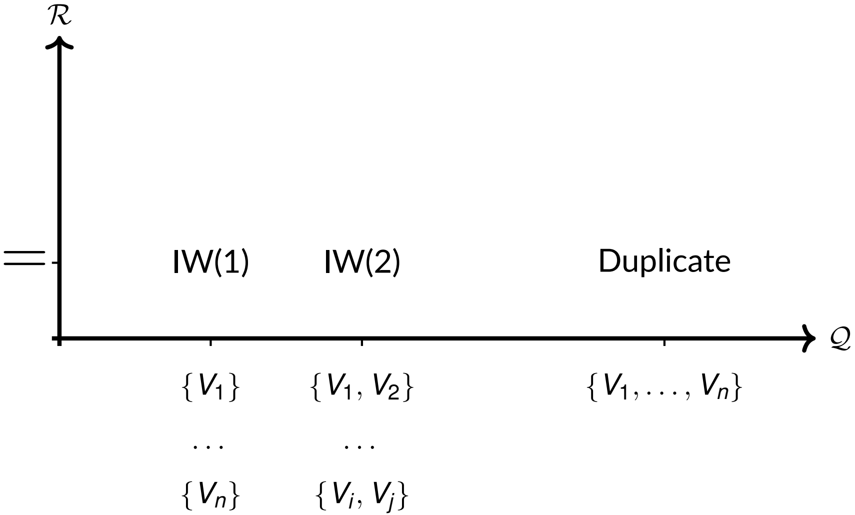
$\rightarrow$ **Novelty can be interpreted as (unsafe/inadmissible) dominance**

$$\exists t \in \mathcal{T}\ h^*(t) \leq h^*(s)$$

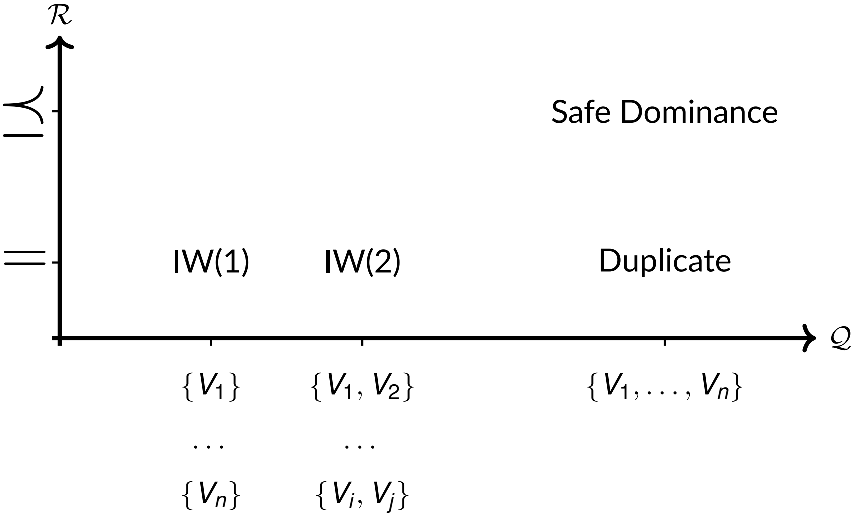
Let $\mathcal{R} = \{\preceq_1, ..., \preceq_k\}$ be a set of relations on P.
Let $\mathcal{Q}$ be a set of subsets of V.

$$\forall Q \in \mathcal{Q} : \exists t \in \mathcal{T} : \forall v \in Q : s[v] \preceq t[v]$$

Unsafe Dominance Pruning (?)

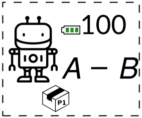


Unsafe Dominance Pruning (?)

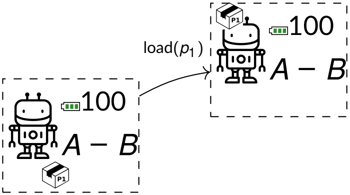


Unsafe Dominance Pruning (?)

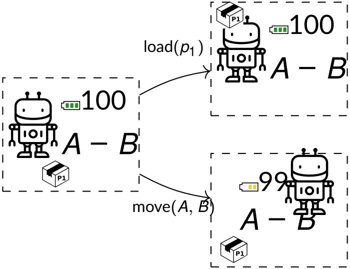
Example IW$\preceq$(1)



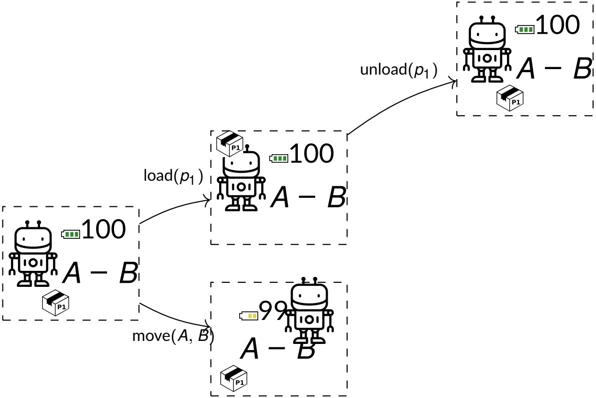
Example IW$\preceq$(1)



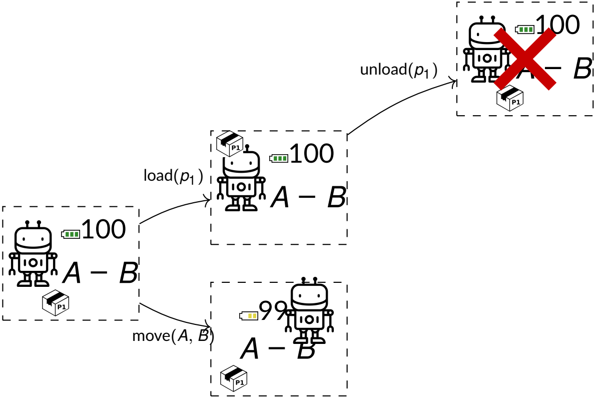
Example IW$\preceq$(1)



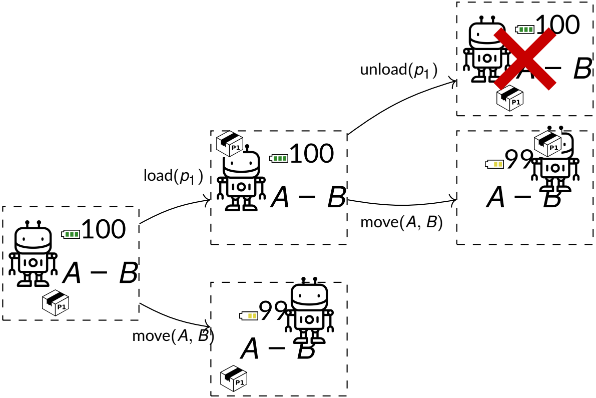
Example IW$\preceq$(1)



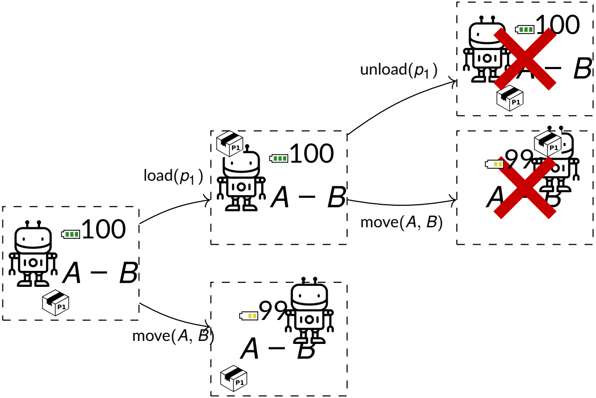
Example IW$\preceq$(1)



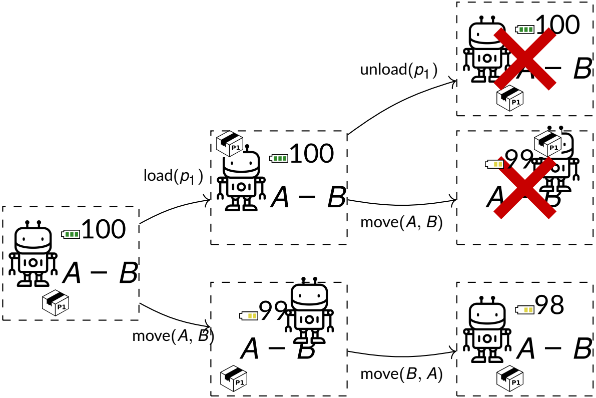
Example IW$\preceq$(1)



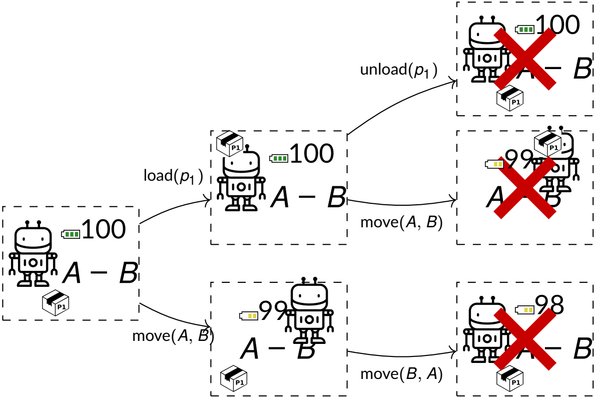
Example IW$\preceq$(1)



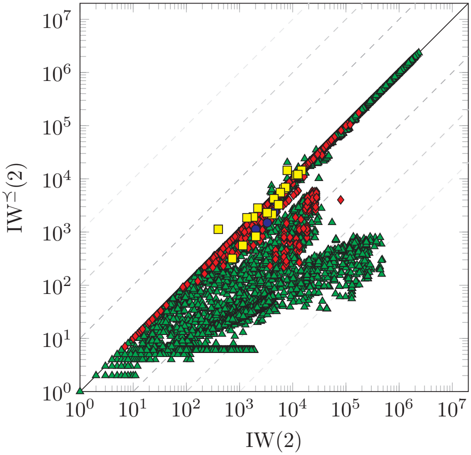
Example IW$\preceq$(1)



Example IW$\preceq$(1)



Results IW(2) with dominance



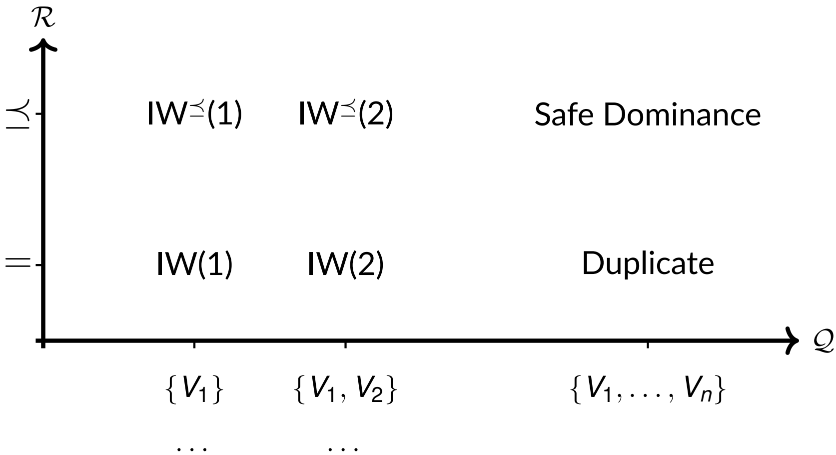
Novelty Heuristics (?)

Not "heuristic" functions in the traditional sense:

$S \times 2^S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$

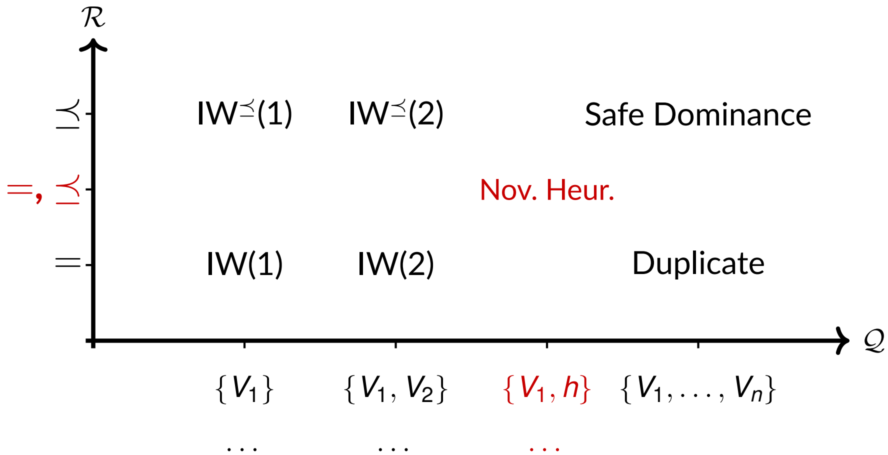
Novelty Heuristics (?)

Not “heuristic” functions in the traditional sense:
$S \times 2^{S} \mapsto \mathbb{R}_0^+ \cup \{\infty\}$



Novelty Heuristics (?)

Not "heuristic" functions in the traditional sense:
$S \times 2^S \mapsto \mathbb{R}_0^+ \cup \{\infty\}$



Overview of Results:

We analyze three variants:

1. Changing $\mathcal{R}$: = vs. $\preceq$
    - Decreases the number of novel states
    - Expansions similar to baseline
    - Performance decreases due to overhead

Overview of Results:

We analyze three variants:

1. Changing $\mathcal{R}$: = vs. $\preceq$
   - Decreases the number of novel states
   - Expansions similar to baseline
   - Performance decreases due to overhead

2. Changing $\mathcal{Q}$ $\rightarrow$Best configuration in practice: choose subsets of variables that appear together in action preconditions

Overview of Results:

We analyze three variants:

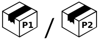
1. Changing $\mathcal{R}$: = vs. $\preceq$
   - Decreases the number of novel states
   - Expansions similar to baseline
   - Performance decreases due to overhead

2. Changing $\mathcal{Q}$ $\rightarrow$Best configuration in practice: choose subsets of variables that appear together in action preconditions

3. Changing quantification of non-novel states (count number of states seen with the same fact to estimate the probability that the state is really dominated)
   - Our non-novel priority is superior to the previous one!
   - But, not good synergy with changing $\mathcal{Q}$

Using Dominance for Agile Planning

	at A	at B
at A	0	−1
at B	−1	0

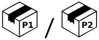


	at A	in R	at B
at A	0	1	2
in R	−3	0	1
at B	−3	−3	0

- We can also use dominance to identify “sub-goals” from which it is safe to restart the search! (torralba:ijcai-18)
- When we deliver a package we have gotten “closer to the goal”, so we can restart the search from there

Using Dominance for Agile Planning

	at A	at B
at A	0	-1
at B	-1	0



	at A	in R	at B
at A	0	**10**	**20**
in R	-12	0	**10**
at B	-14	-12	0

- We can also use dominance to identify “sub-goals” from which it is safe to restart the search! (torralba:ijcai-18)
- When we deliver a package we have gotten “closer to the goal”, so we can restart the search from there

Enforced Hill Climbing

input : Task $\Pi = (V, A, I, G)$, heuristic h
output: Plan or fail

$s = I$
plan = $\langle\rangle$
while $s \not\models G$ **do**
    Run **breadth-first search** from s until finding t **with** $h(s) < h(t)$
    if *succeed* **then**
        plan += sequence of actions from s to t
        $s \leftarrow t$
    else
        return fail
return plan

Enforced Hill Climbing

input : Task $\Pi = (V, A, I, G)$, heuristic h
output: Plan or fail

$s = I$
plan = $\langle\rangle$
while $s \not\models G$ **do**
    Run **breadth-first search** from s until finding t **with** $h(s) < h(t)$
    if *succeed* **then**
        plan += sequence of actions from s to t
        $s \leftarrow t$
    else
        return fail
return plan

- Very effective in domains with the right state space topology
- Incomplete in the presence of unrecognized dead-ends

Enforced Hill Climbing

input : Task $\Pi = (V, A, I, G)$, heuristic h
output: Plan or fail

$s = I$
plan = $\langle\rangle$
while $s \not\models G$ **do**
  Run **breadth-first search** from s until finding t **with** $h(s) < h(t)$
  if *succeed* **then**
    plan += sequence of actions from s to t
    $s \leftarrow t$
  else
    return fail
return plan

- Very effective in domains with the right state space topology
- Incomplete in the presence of unrecognized dead-ends

$\rightarrow$Role of h: is a state **better than** my current state?

Dominance Enforced Hill Climbing

input : Task $\Pi = (V, A, I, G)$, search algorithm X, dom relation $\preceq$
output: Plan or fail

$s = I$
plan = $\langle\rangle$
while $s \not\models G$ **do**
    Run **X** from s until finding t **with** $h(s) \prec h(t)$
    if *succeed* **then**
        plan += sequence of actions from s to t
        $s \leftarrow t$
    else
        return fail
return plan

Dominance Enforced Hill Climbing

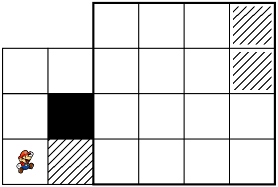
input : Task $\Pi = (V, A, I, G)$, search algorithm X, dom relation $\preceq$
output: Plan or fail

```
s = I
plan = ⟨⟩
while s ⊭ G do
    Run X from s until finding t with h(s) ≺ h(t)
    if succeed then
        plan += sequence of actions from s to t
        s ← t
    else
        return fail
return plan
```

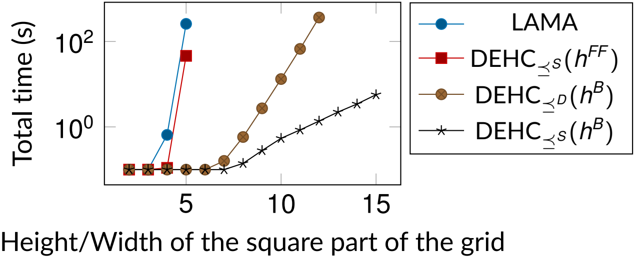
Use dominance to compare states:

- **Guarantees completeness** if $\preceq$ is dead-end safe
- If $\preceq$ is a (satisficing) dominance relation, we may do pruning in X $\rightarrow$Never goes back

Modified Running Example



- Fuel is consumed when moving into stripped tiles



→ Dominance distinguishes which sub-goals are safe!

Sketches (?)

$$
\begin{array}{ll}
\{\neg H, p>0\} \mapsto \{p{\downarrow}, t?\} & \text{; go to nearest pkg} \\
\{\neg H, p=0\} \mapsto \{H\} & \text{; pick it up} \\
\{H, t>0\} \mapsto \{t{\downarrow}\} & \text{; go to target} \\
\{H, n>0, t=0\} \mapsto \{H?, n{\downarrow}, p?\} & \text{; deliver pkg}
\end{array}
$$

- General language for representing the subgoal structure
- Given a start state s, and a candidate state s', the sketch tells whether s' is a sub-goal for reaching the goal from s

Sketches (?)

$$\begin{array}{ll} \{\neg H, p > 0\} \mapsto \{p\downarrow, t?\} & \text{; go to nearest pkg} \\ \{\neg H, p = 0\} \mapsto \{H\} & \text{; pick it up} \\ \{H, t > 0\} \mapsto \{t\downarrow\} & \text{; go to target} \\ \{H, n > 0, t = 0\} \mapsto \{H?, n\downarrow, p?\} & \text{; deliver pkg} \end{array}$$

- General language for representing the subgoal structure
- Given a start state s, and a candidate state s', the sketch tells whether s' is a sub-goal for reaching the goal from s

Source of Information: $S \times S \mapsto \{0, 1\}$

Sketches (?)

$$\{\neg H, p>0\} \mapsto \{p\downarrow, t?\} \quad \text{; go to nearest pkg}$$
$$\{\neg H, p=0\} \mapsto \{H\} \quad \text{; pick it up}$$
$$\{H, t>0\} \mapsto \{t\downarrow\} \quad \text{; go to target}$$
$$\{H, n>0, t=0\} \mapsto \{H?, n\downarrow, p?\} \quad \text{; deliver pkg}$$

- General language for representing the subgoal structure
- Given a start state s, and a candidate state s', the sketch tells whether s' is a sub-goal for reaching the goal from s

Source of Information: $S \times S \mapsto \{0, 1\}$

$\rightarrow$We do have an automatic way of finding "safe" sketches for any single task!

Sketches (?)

$$
\begin{array}{ll}
\{\neg H, p > 0\} \mapsto \{p\downarrow, t?\} & \text{; go to nearest pkg} \\
\{\neg H, p = 0\} \mapsto \{H\} & \text{; pick it up} \\
\{H, t > 0\} \mapsto \{t\downarrow\} & \text{; go to target} \\
\{H, n > 0, t = 0\} \mapsto \{H?, n\downarrow, p?\} & \text{; deliver pkg}
\end{array}
$$

- General language for representing the subgoal structure
- Given a start state s, and a candidate state s', the sketch tells whether s' is a sub-goal for reaching the goal from s

Source of Information: $S \times S \mapsto \{0, 1\}$

$\rightarrow$We do have an automatic way of finding “safe” sketches for any single task! (for tasks with informative QDFs)

Sketches (?)

$$
\begin{array}{ll}
\{\neg H, p > 0\} \mapsto \{p{\downarrow}, t?\} & \text{; go to nearest pkg} \\
\{\neg H, p = 0\} \mapsto \{H\} & \text{; pick it up} \\
\{H, t > 0\} \mapsto \{t{\downarrow}\} & \text{; go to target} \\
\{H, n > 0, t = 0\} \mapsto \{H?, n{\downarrow}, p?\} & \text{; deliver pkg}
\end{array}
$$

- General language for representing the subgoal structure
- Given a start state s, and a candidate state s', the sketch tells whether s' is a sub-goal for reaching the goal from s

Source of Information: $S \times S \mapsto \{0, 1\}$

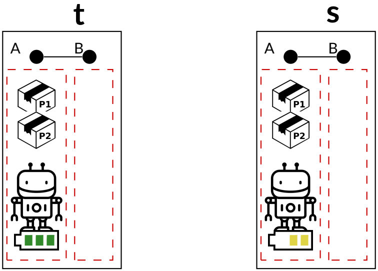
$\rightarrow$We do have an automatic way of finding "safe" sketches for any single task! (for tasks with informative QDFs)

Open Question: Can we use this to verify if a sketch is safe for a new instance?

Contrastive Analysis

Contrastive: *showing the differences between things*

What are the advantages and disadvantages of *t* over *s*?



t is at least as good as s
Disadvantage of s: has less battery

How can we compare states against each other in general ways?

A Family of Contrastive Analysis Methods

→**What does it mean to compare states?**

- Symmetry: Are **A** and **B** equivalent?

A Family of Contrastive Analysis Methods

→**What does it mean to compare states?**

- Symmetry: Are **A** and **B** equivalent?
- Dominance: Is **A** at least as good as **B**? (Torralba, Hoffmann, 2015)
- Quantitative: How much better/worse is **A** compared to **B**? (Torralba 2017)

A Family of Contrastive Analysis Methods

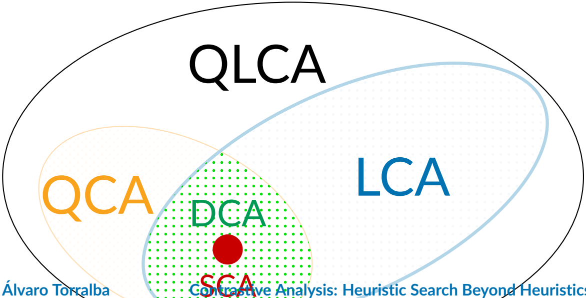
→**What does it mean to compare states?**

- Symmetry: Are **A** and **B** equivalent?
- Dominance: Is **A** at least as good as **B**? (Torralba, Hoffmann, 2015)
- Quantitative: How much better/worse is **A** compared to **B**? (Torralba 2017)
- Logic: Why isn't **A** better than **B**?

A Family of Contrastive Analysis Methods

→**What does it mean to compare states?**

- Symmetry: Are **A** and **B** equivalent?
- Dominance: Is **A** at least as good as **B**? (Torralba, Hoffmann, 2015)
- Quantitative: How much better/worse is **A** compared to **B**? (Torralba 2017)
- Logic: Why isn't **A** better than **B**?
- Quantitative + Logic



What's the Most we can get from Comparing States?

Optimally Efficient Algorithms: Explore the least amount of states given their sources of information

- A^* is optimally efficient

What's the Most we can get from Comparing States?

Optimally Efficient Algorithms: Explore the least amount of states given their sources of information

- A^* is optimally efficient **if your only source of information is a traditional heuristic function**
- **Dominance pruning** ([Torralba,Hoffmann, 2015; Torralba, 2017])
  $\rightarrow$ discard states that are worse than others
- Is A^* with dominance pruning optimally efficient?
  $\rightarrow$only over algorithms that only do pruning! ([Torralba, 2021])

What's the Most we can get from Comparing States?

Optimally Efficient Algorithms: Explore the least amount of states given their sources of information

- A^* is optimally efficient **if your only source of information is a traditional heuristic function**
- **Dominance pruning** ([Torralba,Hoffmann, 2015; Torralba, 2017])
  $\rightarrow$ discard states that are worse than others
- Is A^* with dominance pruning optimally efficient?
  $\rightarrow$only over algorithms that only do pruning! ([Torralba, 2021])

What is the best way to use CA information?

Develop algorithms that can fully reason about the seen states

Beyond Classical Planning

Markov Decision Processes:

- Actions with Stochastic Effects
- Maximize reward and/or minimize cost

→ Recent advances on evaluation functions for finding optimal policies! (Klößner et al., ICAPS'21, SOCS'21)
→Setting up framework for M&S abstractions **Wednesday, 11:40**

Extend Dominance/Contrastive Analysis to more general planning formalisms

Further Uses of State Comparisons

One can use Dominance/Contrastive Analysis beyond improving search algorithms!

- Policy Testing: Enhance metamorphic oracles! (Eisenhut,Torralba,Christakis,Hoffmann, 2023) **Tuesday, 16:00**
- Explainability
  - One can do explanations based on plan properties (Eifler et al., AAAI'20, ICAPS'20), or model reconciliation (Sreedharan, Chakraborti, Kambhampati, AIJ'21).
  - Can we use our dominance/contrastive analysis techniques in the context of explanations to an end user?

Explore further uses of dominance/contrastive analysis

Conclusions

- There is plenty of research to be done on state-space search!
  → **Beyond heuristic functions!**
  → How an agent should think about possible courses of action?

Conclusions

- There is plenty of research to be done on state-space search!

  → **Beyond heuristic functions!**

  → How an agent should think about possible courses of action?

- State-space search algorithms can reason among the entire set of seen states:

  - **Dominance Analysis**: some states are better than others!
  - **Contrastive Analysis**: compare advantages and disadvantages

Conclusions

- There is plenty of research to be done on state-space search!

  → **Beyond heuristic functions!**

  → How an agent should think about possible courses of action?

- State-space search algorithms can reason among the entire set of seen states:

  - **Dominance Analysis**: some states are better than others!
  - **Contrastive Analysis**: compare advantages and disadvantages

- Seemingly unrelated methods can be related to each other if they have the same "signature"

  - Novelty and dominance pruning
  - Sketches and sub-goal detection

Conclusions

- There is plenty of research to be done on state-space search!

  → **Beyond heuristic functions!**

  → How an agent should think about possible courses of action?

- State-space search algorithms can reason among the entire set of seen states:
  - **Dominance Analysis**: some states are better than others!
  - **Contrastive Analysis**: compare advantages and disadvantages

- Seemingly unrelated methods can be related to each other if they have the same "signature"
  - Novelty and dominance pruning
  - Sketches and sub-goal detection

- Ability of comparing states useful for a variety of purposes!

References I