On the Optimal Efficiency of A^* with Dominance Pruning

Álvaro Torralba

AALBORG UNIVERSITET

AAAI'2021

Motivation

- A^*: canonical choice for solving shortest path problems
- A^* is optimally efficient in node expansions (Dechter and Pearl, 1985)

Motivation

- A^*: canonical choice for solving shortest path problems
- A^* is optimally efficient in node expansions (Dechter and Pearl, 1985)

- Dominance pruning methods

Motivation

- A^*: canonical choice for solving shortest path problems
- A^* is optimally efficient in node expansions (Dechter and Pearl, 1985)

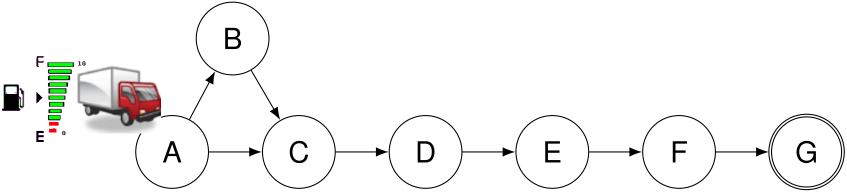
- Dominance pruning methods $\rightarrow$ new source of information!

Motivation

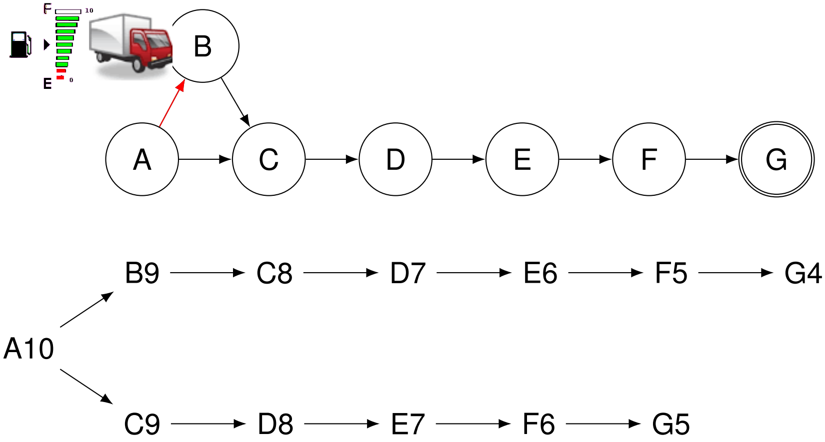
- A^*: canonical choice for solving shortest path problems
- A^* is optimally efficient in node expansions (Dechter and Pearl, 1985)
- Dominance pruning methods $\rightarrow$ new source of information!
- We use dominance pruning in A^*:
  - Is this a good choice?
  - Could we achieve more pruning with different expansion orders?
  - What tie-breaking strategies are good for dominance pruning?

Outline

Running Example



Running Example



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
$\rightarrow h(s)$ estimates the distance from s to the goal $h^*(s)$, $h(s) \leq h^*(s)$

h2

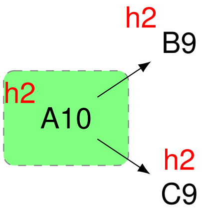
A10

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
$\rightarrow h(s)$ estimates the distance from s to the goal $h^*(s)$, $h(s) \leq h^*(s)$

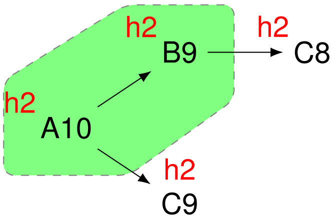


DXBB Algorithms

$UDXBB$: 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
$\rightarrow h(s)$ estimates the distance from s to the goal $h^*(s)$, $h(s) \leq h^*(s)$

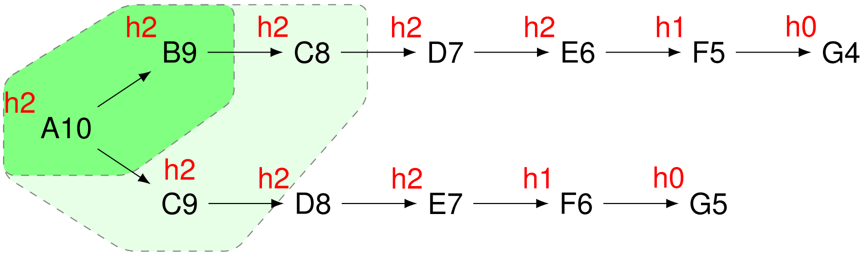


DXBB Algorithms

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

Access to the state space Θ only via node expansions

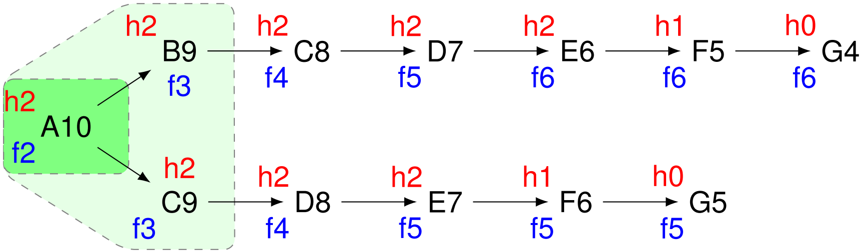
Additionally the algorithm is given an admissible heuristic function h
$\rightarrow h(s)$ estimates the distance from s to the goal $h^*(s)$, $h(s) \leq h^*(s)$



A*

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

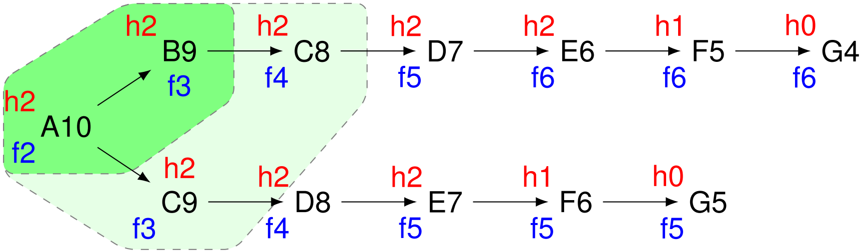
Family of algorithms: tie-breaking strategy may pick any node with minimum f-value



A^*

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

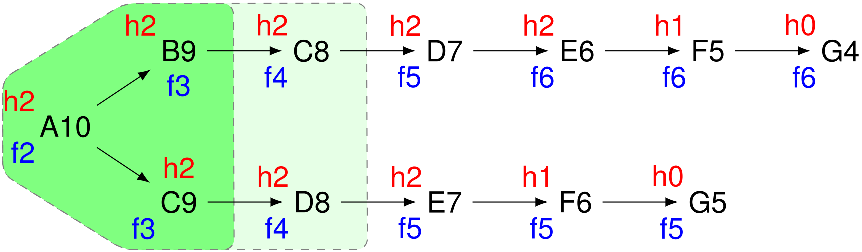
Family of algorithms: tie-breaking strategy may pick any node with minimum f-value



A^*

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

Family of algorithms: tie-breaking strategy may pick any node with minimum f-value



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

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 $I_{\overline{AD}}$	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 $I_{\overline{CON}}$	A^* is 0-optimal	A^* is 0-optimal	A^* is 0-optimal

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

*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 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 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 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)

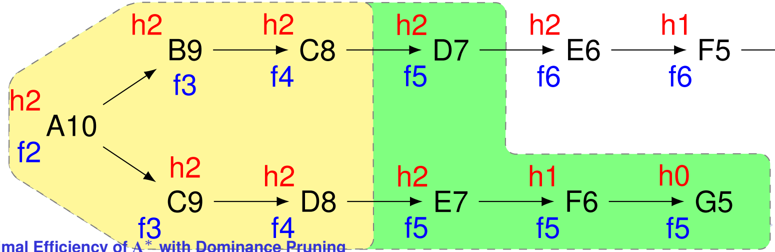
A* is Optimally Efficient (Dechter and Pearl, 1985)

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 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: $f(n) < f^*$



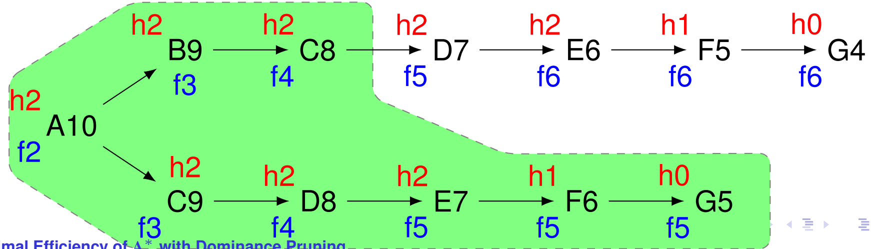
A^* is Optimally Efficient (Dechter and Pearl, 1985)

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 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: $f(n) < f^*$



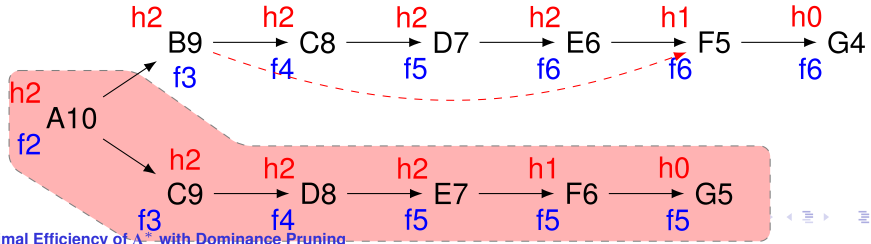
A^* is Optimally Efficient (Dechter and Pearl, 1985)

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 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: $f(n) < f^*$



Outline

Dominance

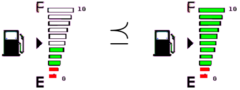
New source of information: dominance relation directly compares pairs of states

$$t \text{ dominates } s \ (s \preceq t) \text{ implies that } h^*(t) \leq h^*(s)$$

Dominance

New source of information: dominance relation directly compares pairs of states

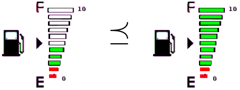
$$t \text{ dominates } s \ (s \preceq t) \text{ implies that } h^*(t) \leq h^*(s)$$

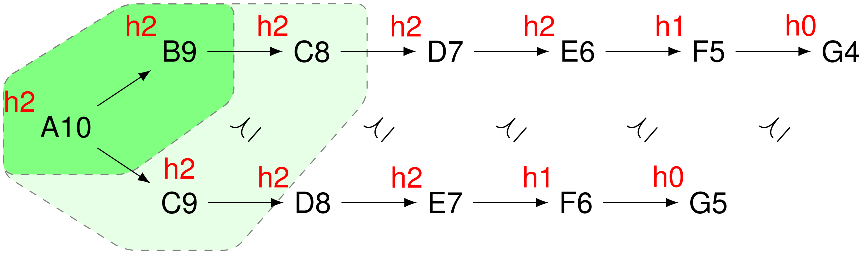


Dominance

New source of information: dominance relation directly compares pairs of states

t dominates s ($s \preceq t$) implies that $h^*(t) \leq h^*(s)$

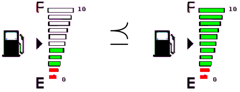


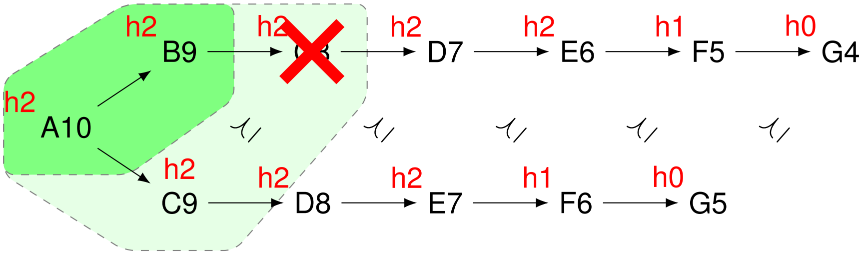


Dominance

New source of information: dominance relation directly compares pairs of states

t dominates s ($s \preceq t$) implies that $h^*(t) \leq h^*(s)$





UDXBB with Dominance Pruning

$UDXBB_{pr}$

UDXBB algorithms that can prune a node n_s whenever another n_t has been seen such that $g(n_t) \leq g(n_s)$ and t dominates s.

UDXBB with Dominance Pruning

$UDXBB_{pr}$

UDXBB algorithms that can prune a node n_s whenever another n_t has been seen such that $g(n_t) \leq g(n_s)$ and t dominates s.

No access to $\preceq$: can only use dominance for pruning according to the rule above

UDXBB with Dominance Pruning

$UDXBB_{pr}$

UDXBB algorithms that can prune a node n_s whenever another n_t has been seen such that $g(n_t) \leq g(n_s)$ and t dominates s.

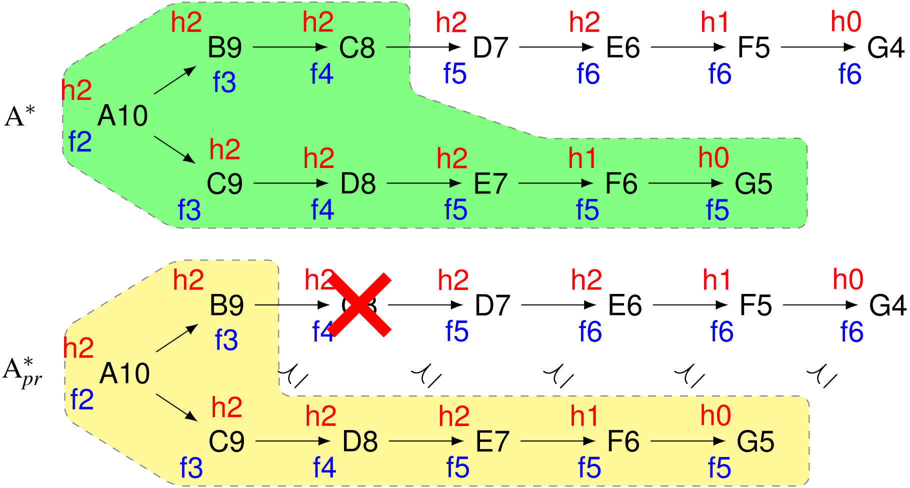
No access to $\preceq$: can only use dominance for pruning according to the rule above

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

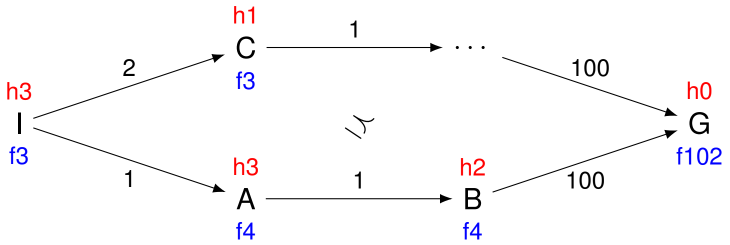
Outline

A^*_{pr} is 1-optimal over A^* (until last f-layer)



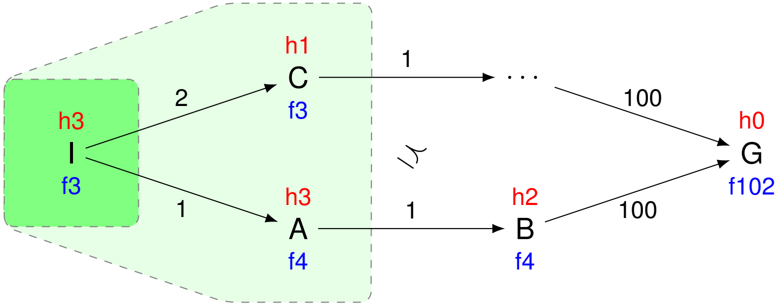
A^*_{pr} is not optimally efficient in general

→The expansion order of A^* may not be optimal



A^*_{pr} is not optimally efficient in general

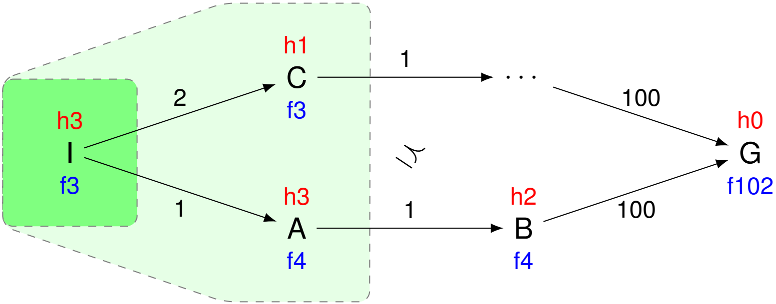
→The expansion order of A^* may not be optimal



A^*_{pr}: $\langle I, C, \ldots, A, B, G \rangle$

A^*_{pr} is not optimally efficient in general

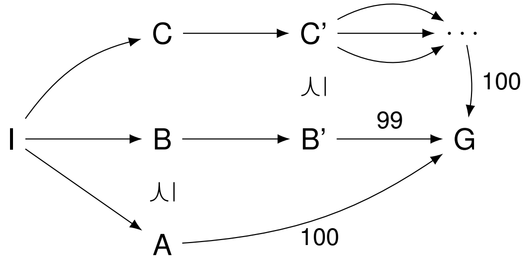
→The expansion order of A^* may not be optimal



A^*_{pr}: $\langle I, C, \ldots, A, B, G\rangle$
optimal: $\langle I, A, B, G\rangle$

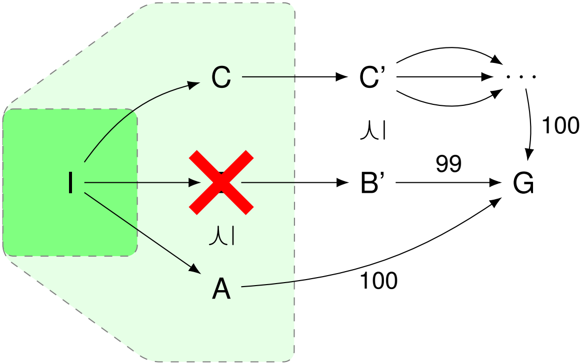
A^*_{pr} is not optimally efficient in general (take 2)

→Sometimes it could be worth it to expand a node even if it can be pruned



A^*_{pr} is not optimally efficient in general (take 2)

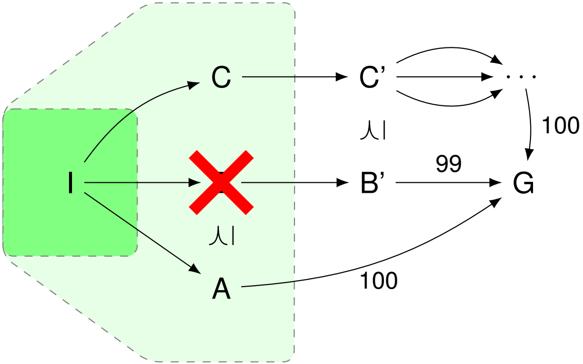
→Sometimes it could be worth it to expand a node even if it can be pruned



A^*_{pr}: $\langle I, A, C, C', \ldots, G \rangle$

A^*_{pr} is not optimally efficient in general (take 2)

→Sometimes it could be worth it to expand a node even if it can be pruned



A^*_{pr}: $\langle I, A, C, C', \ldots, G \rangle$

optimal: $\langle I, A, B, C, G \rangle$

Consistency

Consistent instances

An instance $I = \langle \Theta, h, \preceq \rangle$ is consistent if:

(i) h is consistent

(ii) $\preceq$ is a transitive cost-simulation relation

(iii) $\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

Consistent Dominance Relations

$\preceq$ is a transitive cost-simulation relation

Consistent Dominance Relations

$\preceq$ is a transitive cost-simulation relation

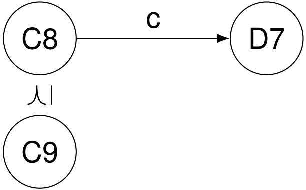
Transitive: $C \preceq B$ and $B \preceq A$ implies $C \preceq A$

Consistent Dominance Relations

$\preceq$ is a transitive cost-simulation relation

Transitive: $C \preceq B$ and $B \preceq A$ implies $C \preceq A$

Cost-simulation:

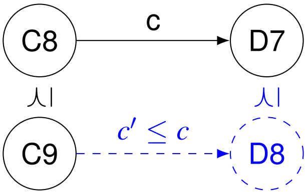


Consistent Dominance Relations

$\preceq$ is a transitive cost-simulation relation

Transitive: $C \preceq B$ and $B \preceq A$ implies $C \preceq A$

Cost-simulation:

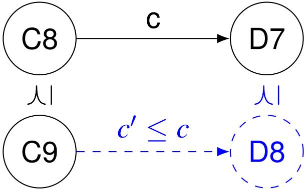


Consistent Dominance Relations

$\preceq$ is a transitive cost-simulation relation

Transitive: $C \preceq B$ and $B \preceq A$ implies $C \preceq A$

Cost-simulation:



→State-of-the-art methods on dominance pruning derive transitive cost-simulation relations

Consistency between Heuristics and Dominance

$\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

Consistency between Heuristics and Dominance

$\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

If $h(s) < h(t)$ then $h(t) \leq h^*(t) \leq h^*(s)$, so we can raise $h(s)$ to $h(t)$!

Consistency between Heuristics and Dominance

$\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

If $h(s) < h(t)$ then $h(t) \leq h^*(t) \leq h^*(s)$, so we can raise $h(s)$ to $h(t)$!

Thm: If $\preceq$ is consistent with h, whenever n_s can be pruned by n_t then $f(n_t) \leq f(n_s)$

Consistency between Heuristics and Dominance

$\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

If $h(s) < h(t)$ then $h(t) \leq h^*(t) \leq h^*(s)$, so we can raise $h(s)$ to $h(t)$!

Thm: If $\preceq$ is consistent with h, whenever n_s can be pruned by n_t then $f(n_t) \leq f(n_s)$

Conjecture: Most consistent heuristics and dominance relations are consistent unless dominance considers larger subsets of variables

Consistency between Heuristics and Dominance

$\preceq$ is consistent with h: $s \preceq t \implies h(t) \leq h(s)$

If $h(s) < h(t)$ then $h(t) \leq h^*(t) \leq h^*(s)$, so we can raise $h(s)$ to $h(t)$!

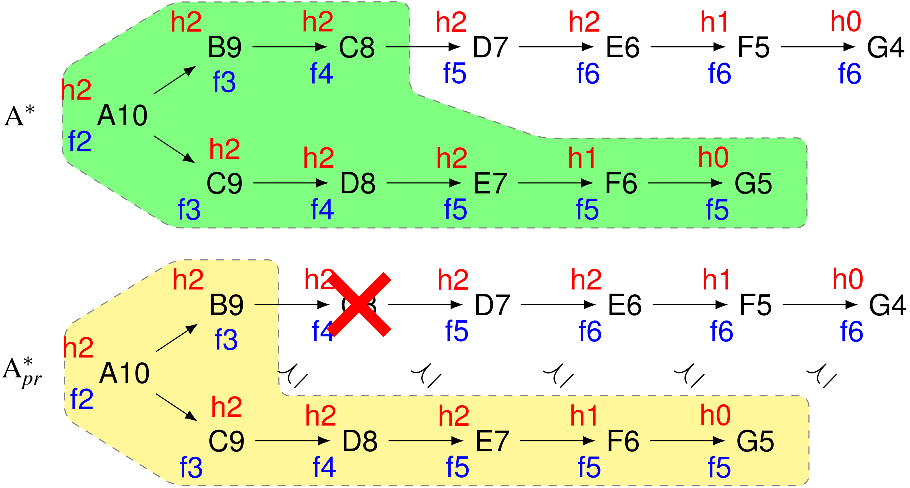
Thm: If $\preceq$ is consistent with h, whenever n_s can be pruned by n_t then $f(n_t) \leq f(n_s)$

Conjecture: Most consistent heuristics and dominance relations are consistent unless dominance considers larger subsets of variables

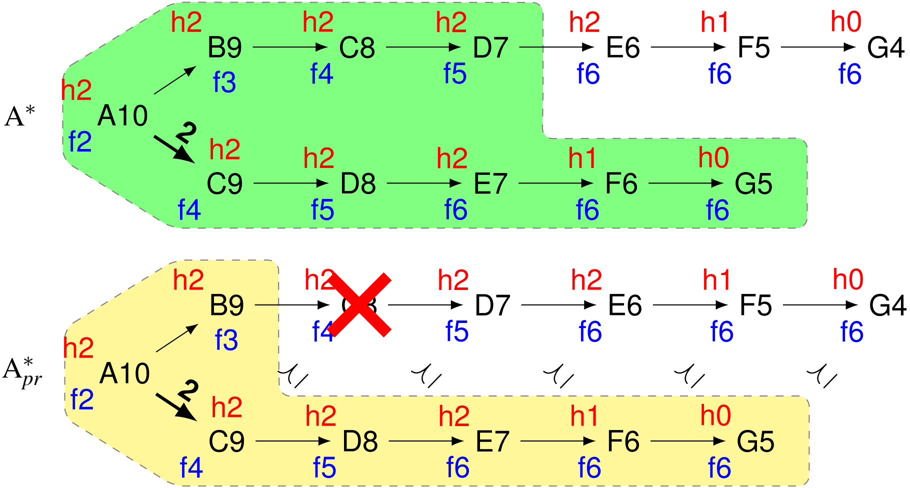
The information of h and $\preceq$ is still complementary!

- When $h(s) < h(t)$, s is more promising but there is no guarantee
- When $t \preceq s$, we are certain that t is as good as s (but if $s \not\preceq t$ we know nothing)

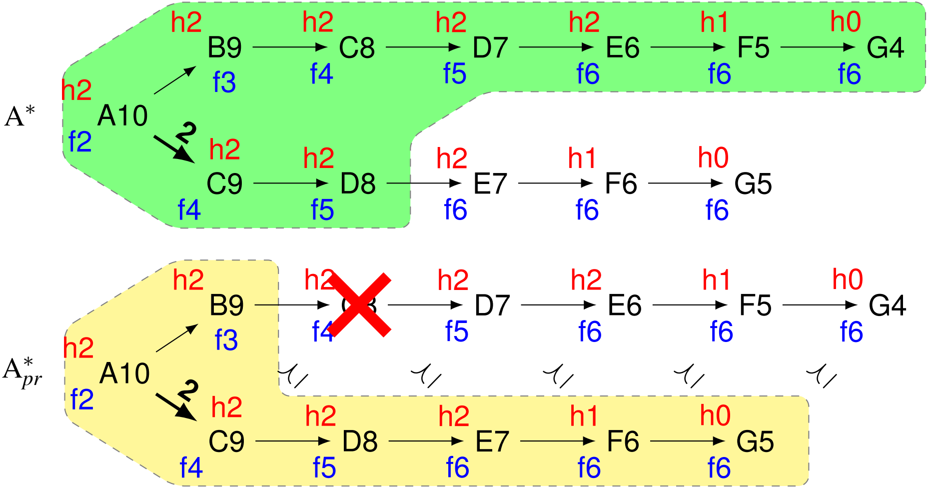
Is A^*_{pr} 1-optimal on consistent instances?



A^*_{pr} is NOT 1-optimal on consistent instances



A^*_{pr} is NOT 1-optimal on consistent instances



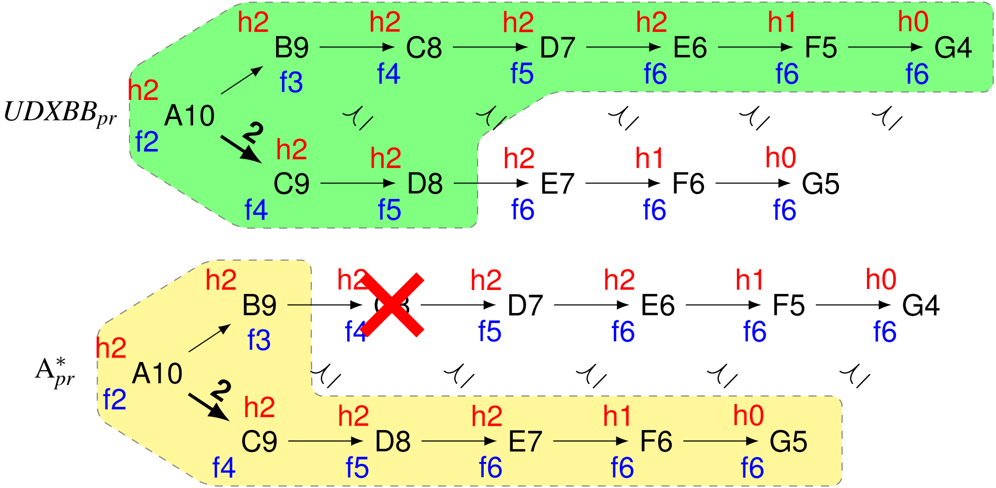
A^*_{pr} is #-optimal

A^*_{pr} is #-optimal on consistent instances wrt. $UDXBB_{pr}$

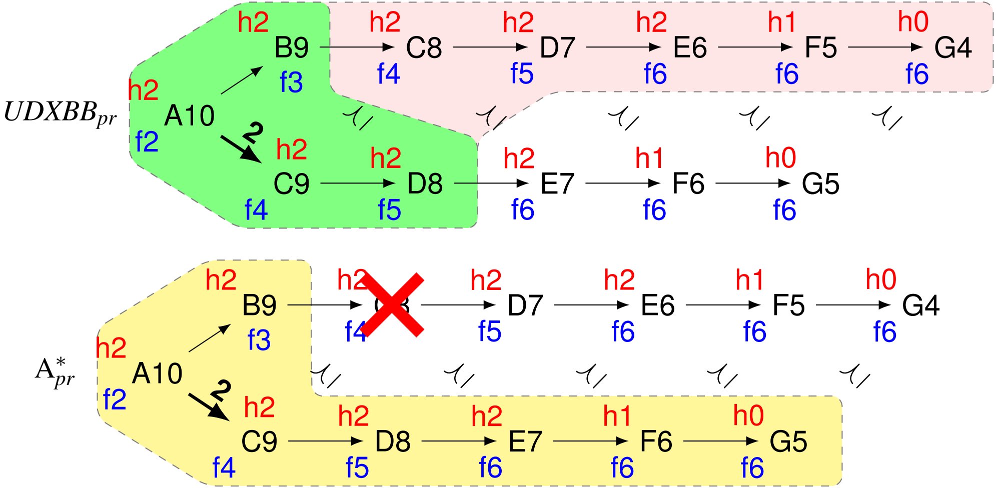
Let N be the set of states expanded by any admissible $UDXBB_{pr}$ algorithm, then there exists a tie-breaking of A^*_{pr} that expands N' with $|N'| \leq |N|$.

$\rightarrow$It may not expand a subset of nodes but it will expand the least amount of nodes!

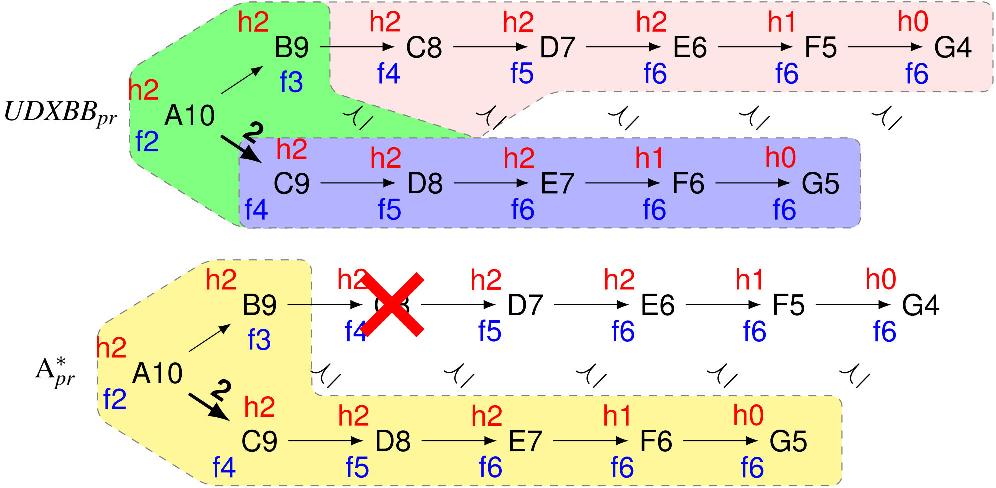
A^*_{pr} is #-optimal on consistent instances



A^*_{pr} is #-optimal on consistent instances



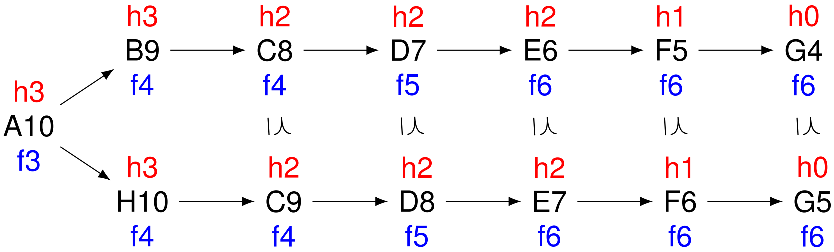
A^*_{pr} is #-optimal on consistent instances



Outline

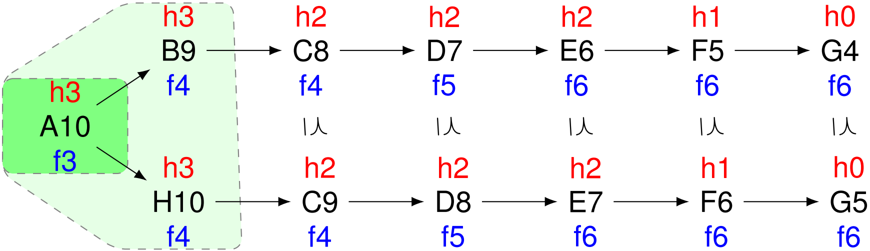
Tie-Breaking

- In A^* tie-breaking is only relevant in the last f-layer
- In A^*_{pr}, tie-breaking is relevant in all layers



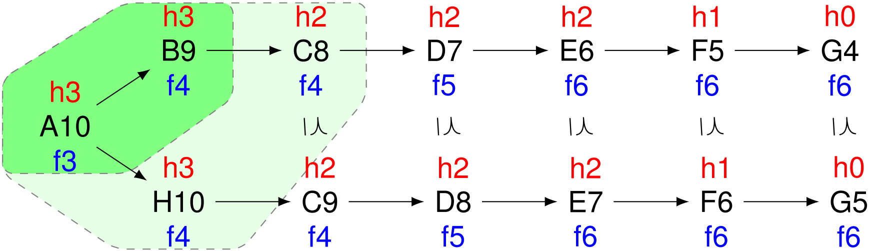
Tie-Breaking

- In A^* tie-breaking is only relevant in the last f-layer
- In A^*_{pr}, tie-breaking is relevant in all layers



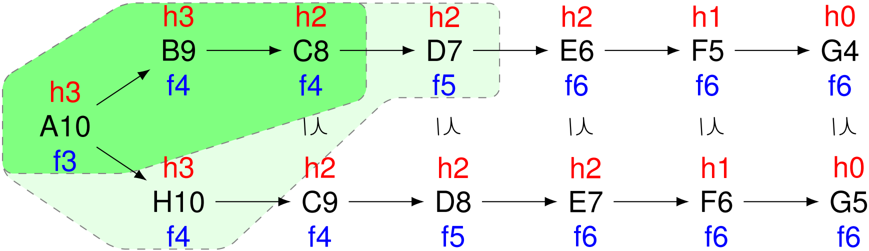
Tie-Breaking

- In A^* tie-breaking is only relevant in the last f-layer
- In A^*_{pr}, tie-breaking is relevant in all layers



Tie-Breaking

- In A^* tie-breaking is only relevant in the last f-layer
- In A^*_{pr}, tie-breaking is relevant in all layers



Tie-Breaking Strategies

$A^*_{h<,pr}$

Prefer nodes with lower h value

- Standard in most implementations of A^*
- Advantage: follow heuristic in the last f-layer

Tie-Breaking Strategies

$\mathrm{A}^*_{h<,pr}$

Prefer nodes with lower h value

- Standard in most implementations of A^*
- Advantage: follow heuristic in the last f-layer

→We prove that it is not optimally efficient until the last f-layer

Tie-Breaking Strategies

$A^*_{h<,pr}$

Prefer nodes with lower h value

- Standard in most implementations of A^*
- Advantage: follow heuristic in the last f-layer

$\rightarrow$We prove that it is not optimally efficient until the last f-layer

$A^*_{g<,pr}$

Prefer nodes with lower g value
$\rightarrow$We prove that it is optimally efficient until the last f-layer

Tie-Breaking Strategies

$A^*_{h<,pr}$

Prefer nodes with lower h value

- Standard in most implementations of A^*
- Advantage: follow heuristic in the last f-layer

$\rightarrow$We prove that it is not optimally efficient until the last f-layer

$A^*_{g<,pr}$

Prefer nodes with lower g value
$\rightarrow$We prove that it is optimally efficient until the last f-layer

Trade off:

- follow h in the last f-layer
- follow $\preceq$ up to the last f-layer

Conclusion

- Dominance pruning introduces a new source of information for heuristic search 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
- A^*_{pr} is #-optimally efficient on consistent instances
- Until last layer is better to break ties in favor of minimum g-value