ON THE OPTIMAL EFFICIENCY OF A* WITH DOMINANCE PRUNING

ÁLVARO TORRALBA

AALBORG UNIVERSITET

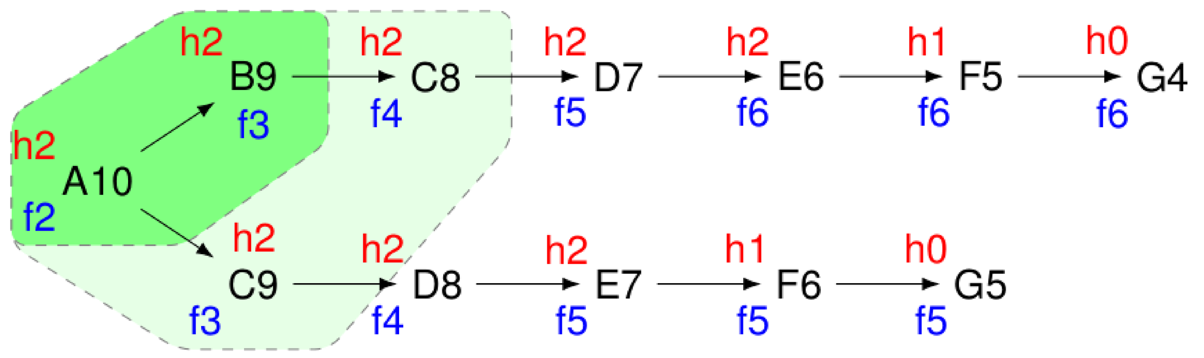
MOTIVATION

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

SEARCH ALGORITHMS

UDXBB: Unidirectional Deterministic Expansion-based Black-Box
→Can only get information of the state space by expanding nodes

A*: →Expands nodes based on f-value: $f(n_s) = g(n_s) + h(s)$
→Family of algorithms: tie-breaking may pick any node with min f



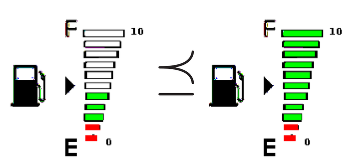
A* IS 1-OPTIMAL ON CONSISTENT INSTANCES (DP, 1985)

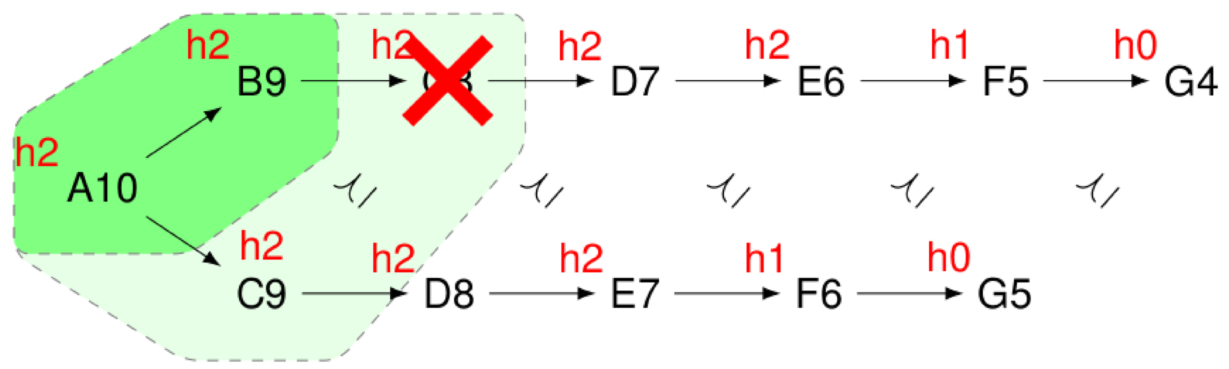
A* is 1-optimal on consistent instances Let N be the set of states expanded by any admissible *UDXBB* algorithm, then there exists a tie-breaking of A* that expands subset of N.
Consistent Heuristic: $h(s) - h(t) \le c(s,t)$

DOMINANCE PRUNING

Dominance relation directly compares pairs of states

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





UDXBB WITH DOMINANCE PRUNING

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

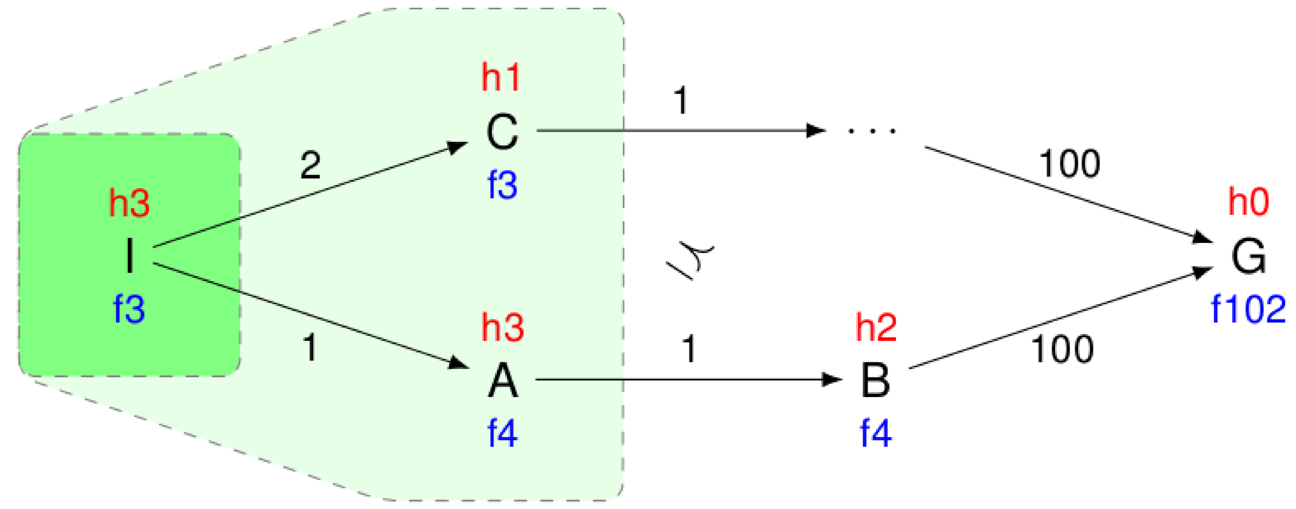
→No access to $\preceq$: can only use dominance for pruning according to the rule above

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

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

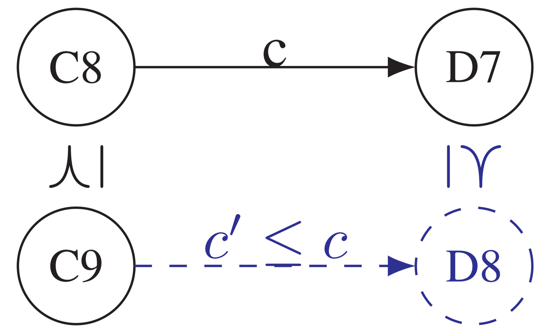
→Sometimes it is better to expand a node even if it can be pruned

NEW DEFINITION OF CONSISTENT INSTANCES

1. Consistent heuristic: $h(s) - h(t) \le c(s,t)$
2. Dominance relation $\preceq$ is a transitive cost-simulation relation

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

   Cost-simulation: Whenever $s \preceq t$ and $s \xrightarrow{a} s'$, either $s' \preceq t$ or there exists $t \xrightarrow{a'} t'$ such that $s' \preceq t'$ and $c(a) \le c(a')$

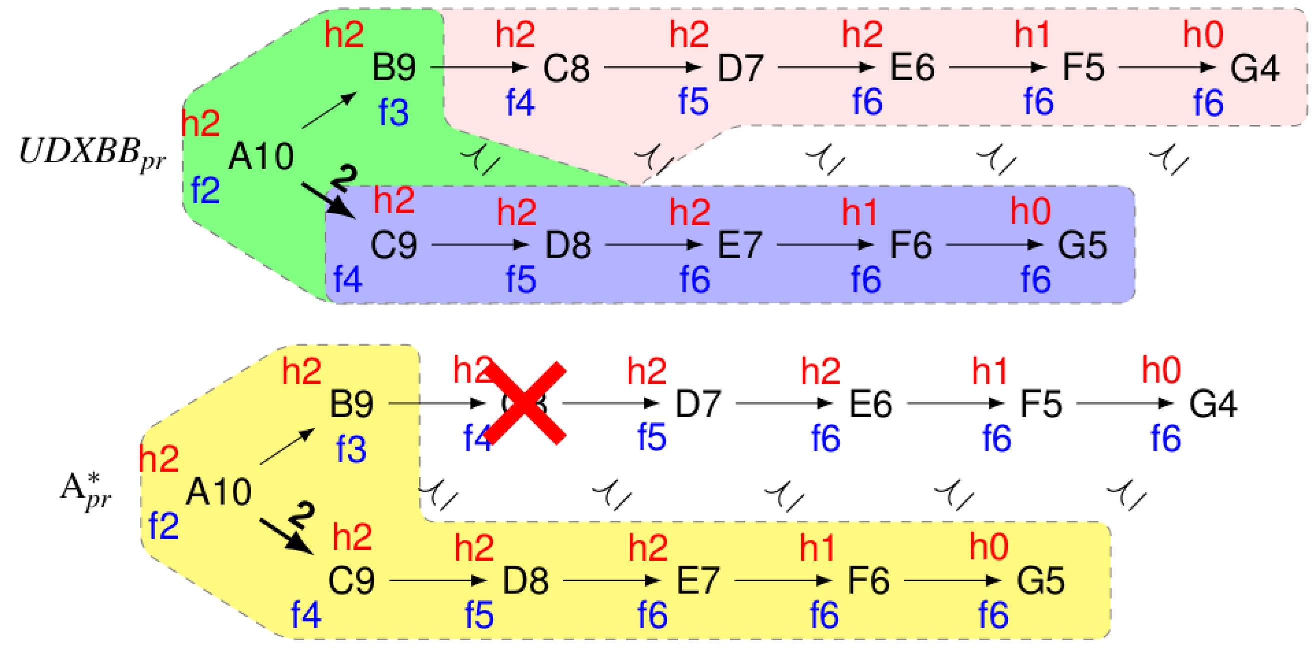


3. Heuristic and dominance relation are consistent with each other

$$s \preceq t \implies h(t) \le h(s)$$

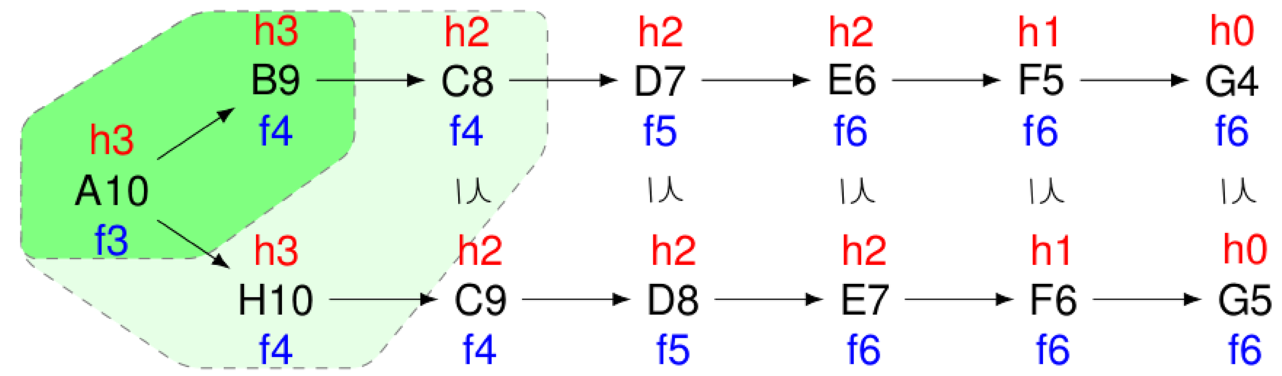
A$^*_{pr}$ IS #-OPTIMAL

A$^*_{pr}$ is not 1-optimal, because it can expand a diferent set of nodes. **A$^*_{pr}$ is #-optimal on consistent instances** (it expands fewer or equal nodes)



TIE-BREAKING STRATEGIES

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



We consider two tie-breaking strategies:

1. 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

2. Prefer nodes with lower g value

   →We prove that it is optimally efficient until the last f-layer

CONCLUSIONS

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