New Results in Bounded-Suboptimal Search

Maximilian Fickert[1] and Tianyi Gu[2] and Wheeler Ruml[2]

1 UNIVERSITÄT DES SAARLANDES

2 University of New Hampshire

heuristic search: a planning approach

Planning as Heuristic Graph Search

heuristic search: a planning approach

planning models the environment as a state space problem and finds a sequence of actions that accomplishes some objective

Planning as Heuristic Graph Search

heuristic search: a planning approach

planning models the environment as a state space problem and finds a sequence of actions that accomplishes some objective

heuristic search:

{states, actions} $\rightarrow$ {V, E}
planning problem $\rightarrow$ find a path from s_{init} to $\{s_{goal}\}$
guide graph search by a heuristic estimate of cost-to-goal

Planning as Heuristic Graph Search

heuristic search: a planning approach

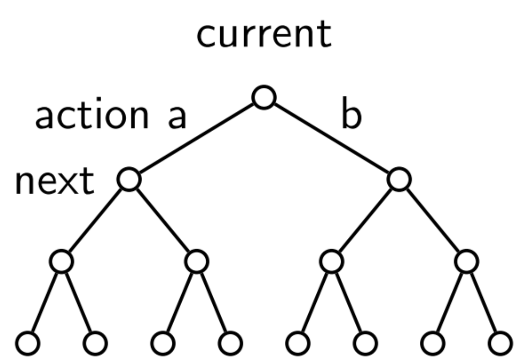
planning models the environment as a state space problem and finds a sequence of actions that accomplishes some objective

heuristic search:

{states, actions} $\rightarrow$ {V, E}

planning problem $\rightarrow$ find a path from s_{init} to $\{s_{goal}\}$

guide graph search by a heuristic estimate of cost-to-goal



Planning as Heuristic Graph Search

A*: expands the node with minimal f value
returns optimal path
optimal search can take too long!
because it must expand every node with $f < C^*$,
there can be many such nodes[1]

[1] How Good is Almost Perfect, Malte Helmert and Gabriele Roger, AAAI, 2008.

A*: expands the node with minimal f value
returns optimal path
optimal search can take too long!
because it must expand every node with $f < C^*$,
there can be many such nodes[1]

What if we don't have time?

[1] How Good is Almost Perfect, Malte Helmert and Gabriele Roger, AAAI, 2008.

optimal: minimize solution cost
expand every node with $f < C^*$

greedy: minimize solving time

anytime: incrementally converge to optimal

bounded-suboptimal: minimize time subject to relative cost bound (factor of optimal)

bounded-cost: minimize time subject to absolute cost bound

contract: minimize cost subject to absolute time bound

utility-based: minimize function of cost and time

Alternatives to Optimal Search: Problem Settings

optimal: minimize solution cost
expand every node with $f < C^*$

greedy: minimize solving time

anytime: incrementally converge to optimal

bounded-suboptimal: minimize time subject to relative cost bound (factor of optimal)

bounded-cost: minimize time subject to absolute cost bound

contract: minimize cost subject to absolute time bound

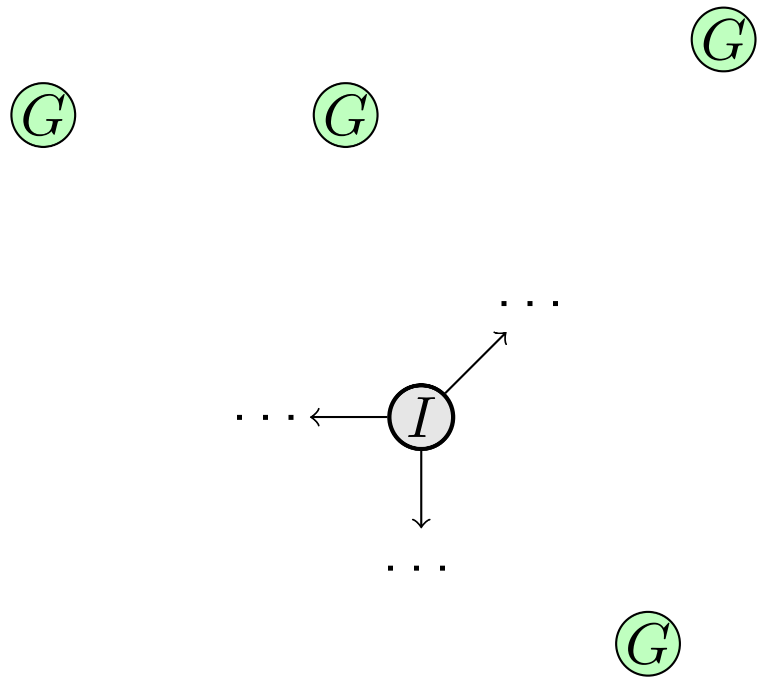
utility function: minimize utility function of cost and time

Overview

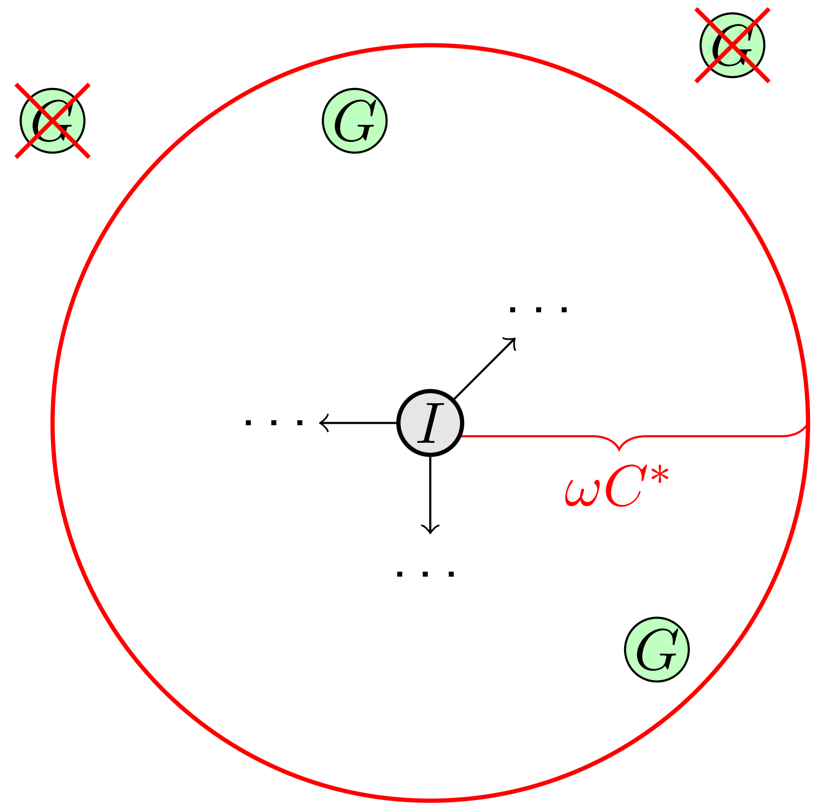
- Introduction
- Bounded-Suboptimal Search
  - EES
  - DPS
  - XES
- New Algorithms
  - DXES
  - RoundRobin
- Results
- Conclusions

Bounded-Suboptimal Search

Bounded-Suboptimal Search: The Problem Setting



Bounded-Suboptimal Search: The Problem Setting



Objective: Find a plan with cost at most ωC^* as fast as possible.

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information:

h: a lower bound on cost-to-go

$$f(n) = g(n) + h(n)$$

traditional A$*$ lower bound

$\hat{h}$: an estimate of cost-to-go

$$\hat{f} = g(n) + \hat{h}(n)$$

unbiased estimates can be more informed

$\hat{d}$: an estimate of distance-to-go

nearest goal is the easiest to find

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

$best_f$: open node giving lower bound on cost
$best_{\hat{f}}$: open node giving estimated optimal cost
$best_{\hat{d}}$: estimated $\omega-$suboptimal node with minimum $\hat{d}$

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

$best_f$: open node giving lower bound on cost
$best_{\hat{f}}$: open node giving estimated optimal cost
$best_{\hat{d}}$: estimated $\omega-$suboptimal node with minimum $\hat{d}$

node to expand next:

1. pursue the nearest goal estimated to lie within the bound
2.
3.

in other words:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2.
3.

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

$best_f$: open node giving lower bound on cost
$best_{\hat{f}}$: open node giving estimated optimal cost
$best_{\hat{d}}$: estimated $\omega-$suboptimal node with minimum $\hat{d}$

node to expand next:

1. pursue the nearest goal estimated to lie within the bound
2. pursue the estimated optimal solution
3.

in other words:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. **else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$
3.

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

$best_f$: open node giving lower bound on cost
$best_{\hat{f}}$: open node giving estimated optimal cost
$best_{\hat{d}}$: estimated $\omega-$suboptimal node with minimum $\hat{d}$

node to expand next:

1. pursue the nearest goal estimated to lie within the bound
2. pursue the estimated optimal solution
3. raise the lower bound on optimal solution cost

in other words:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. **else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$
3. **else** $best_f$

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. **else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$
3. **else** $best_f$

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. **else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$
3. **else** $best_f$

Other EES variants:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. ~~**else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$~~ ?
3. **else** $best_f$

see paper for more details.

State-of-The-Art: 1/2 EES (Thayer and Ruml, 2011)

Three source of heuristic information: h, $\hat{h}$, $\hat{d}$

EES search strategy:

1. **if** $\hat{f}(best_{\hat{d}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{d}}$
2. **else if** $\hat{f}(best_{\hat{f}}) < \omega \cdot f(best_f)$ **then** $best_{\hat{f}}$
3. **else** $best_f$

Problem:

- EES does not consider the uncertainty of its estimates (brittle)

State-of-The-Art: 2/2 DPS (Gilon, Felner, and Stern, 2016)

Best first search on “potential”:

$$\text{potential} = \frac{\text{budget} - \text{cost-so-far}}{\text{cost-to-go}}$$

State-of-The-Art: 2/2 DPS (Gilon, Felner, and Stern, 2016)

Best first search on "potential":

$$\text{potential} = \frac{\text{budget} - \text{cost-so-far}}{\text{cost-to-go}}$$

in other words:

$$ud(n) = \frac{\omega \cdot f_{min} - g(n)}{h(n)}$$

State-of-The-Art: 2/2 DPS (Gilon, Felner, and Stern, 2016)

Best first search on "potential":

$$\text{potential} = \frac{\text{budget} - \text{cost-so-far}}{\text{cost-to-go}}$$

in other words:

$$ud(n) = \frac{\omega \cdot f_{min} - g(n)}{h(n)}$$

does not explicitly optimize search time

Bounded-Cost: XES (Fickert, Gu, and Ruml, 2021)

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

Bounded-Cost: XES (Fickert, Gu, and Ruml, 2021)

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

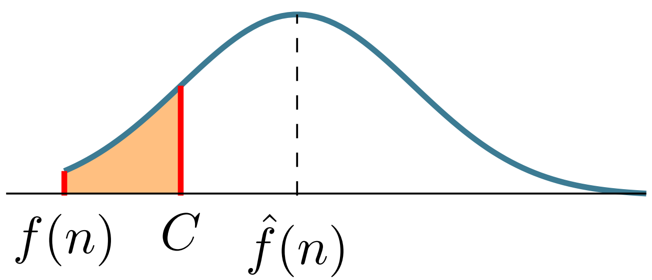
$T(n)$: total search effort, estimated by $d(n)$
penalize nodes distant to goal
$p(n)$: the probability of finding a solution within the bound
advantage nodes likely to have solution within bound

Bounded-Cost: XES (Fickert, Gu, and Ruml, 2021)

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

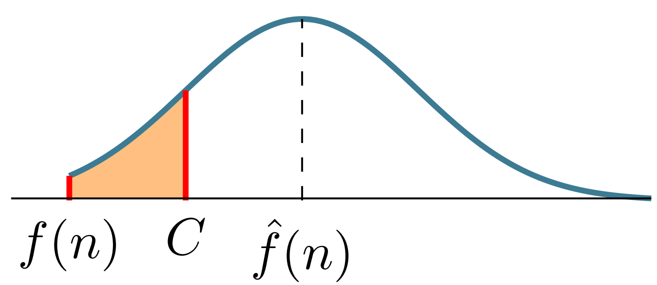
$T(n)$: total search effort, estimated by $d(n)$
penalize nodes distant to goal
$p(n)$: the probability of finding a solution within the bound
advantage nodes likely to have solution within bound
estimated by:



Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$
penalize nodes distant to goal
$p(n)$: the probability of finding a solution within the bound
advantage nodes likely to have solution within bound
estimated by:



Can we adapt XES to bounded-suboptimal setting?

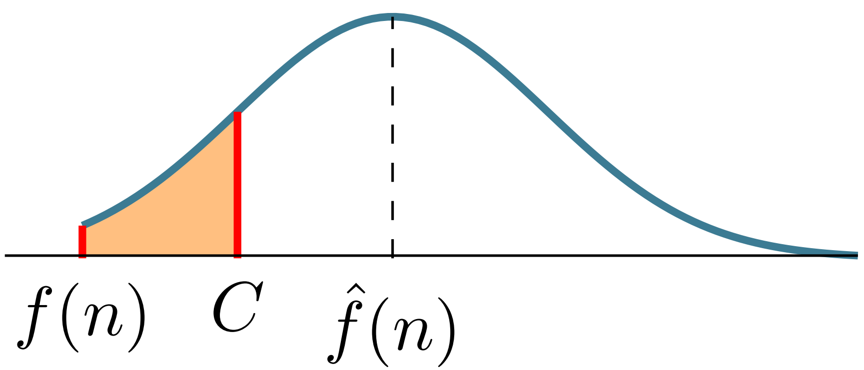
New Algorithms

Our Approach: 1/2 Dynamic XES

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$
$p(n)$: the probability of finding a solution within the bound, estimated by:



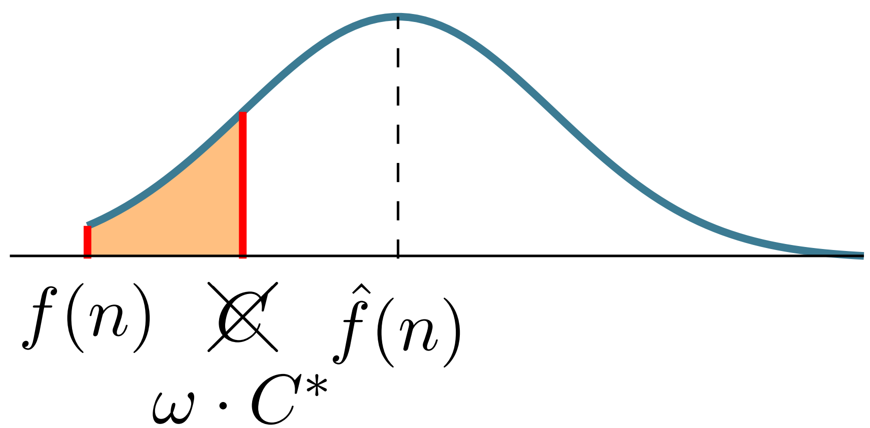
Our Approach: 1/2 Dynamic XES

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$

$p(n)$: the probability of finding a solution within the bound, estimated by:

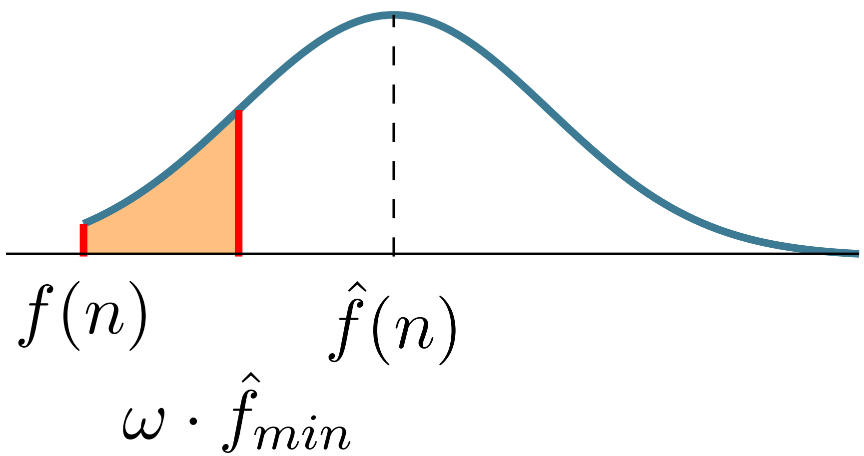


Our Approach: 1/2 Dynamic XES

Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$
$p(n)$: the probability of finding a solution within the estimated bound, estimated by:

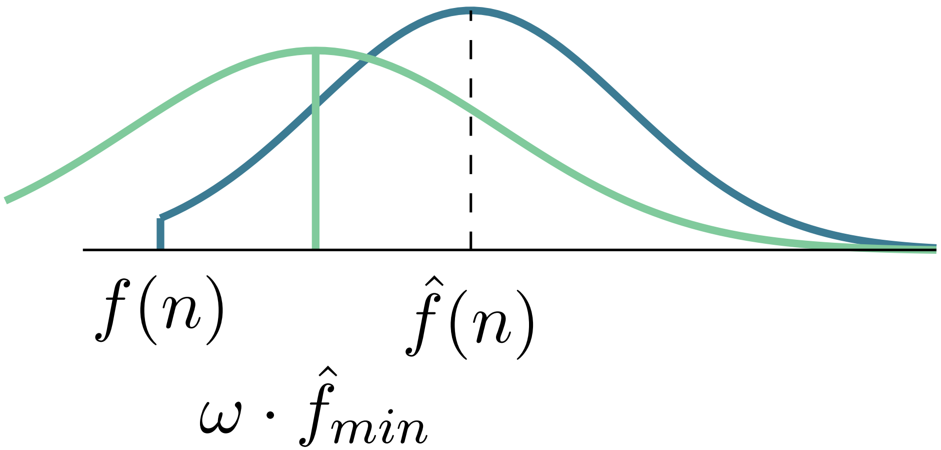


Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$

$p(n)$: the probability of finding a solution within the estimated bound, estimated by:

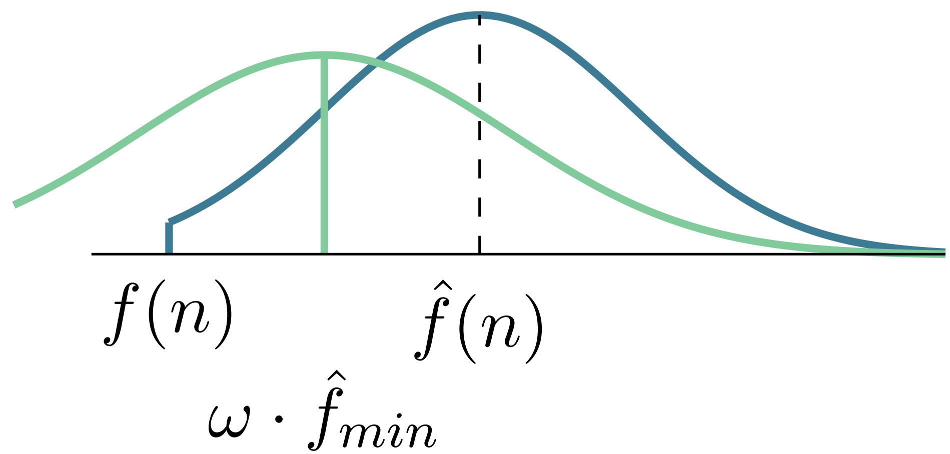


Best first search on expected search effort:

$$xe(n) = \frac{T(n)}{p(n)}$$

$T(n)$: total search effort, estimated by $d(n)$

$p(n)$: the probability of finding a solution within the estimated bound, estimated by:



hard to estimate when raising the bound is useful!

Our Approach: 2/2 A Round-Robin Scheme

Replace EES selection rule with Round-Robin:

- **focal list:** sorted by d(EES) or ud(DPS) or xe(DXES)
- **open list:** sorted by $\hat{f}$
- **cleanup list:** sorted by f

focal and open condition: $f(n) < \omega \cdot f_{min}$

Simple but works well!

Results

Planning Domains:

- Implementation in Fast Downward
- Benchmarks:
  IPC optimal tracks (48 domains)

Search Domains:

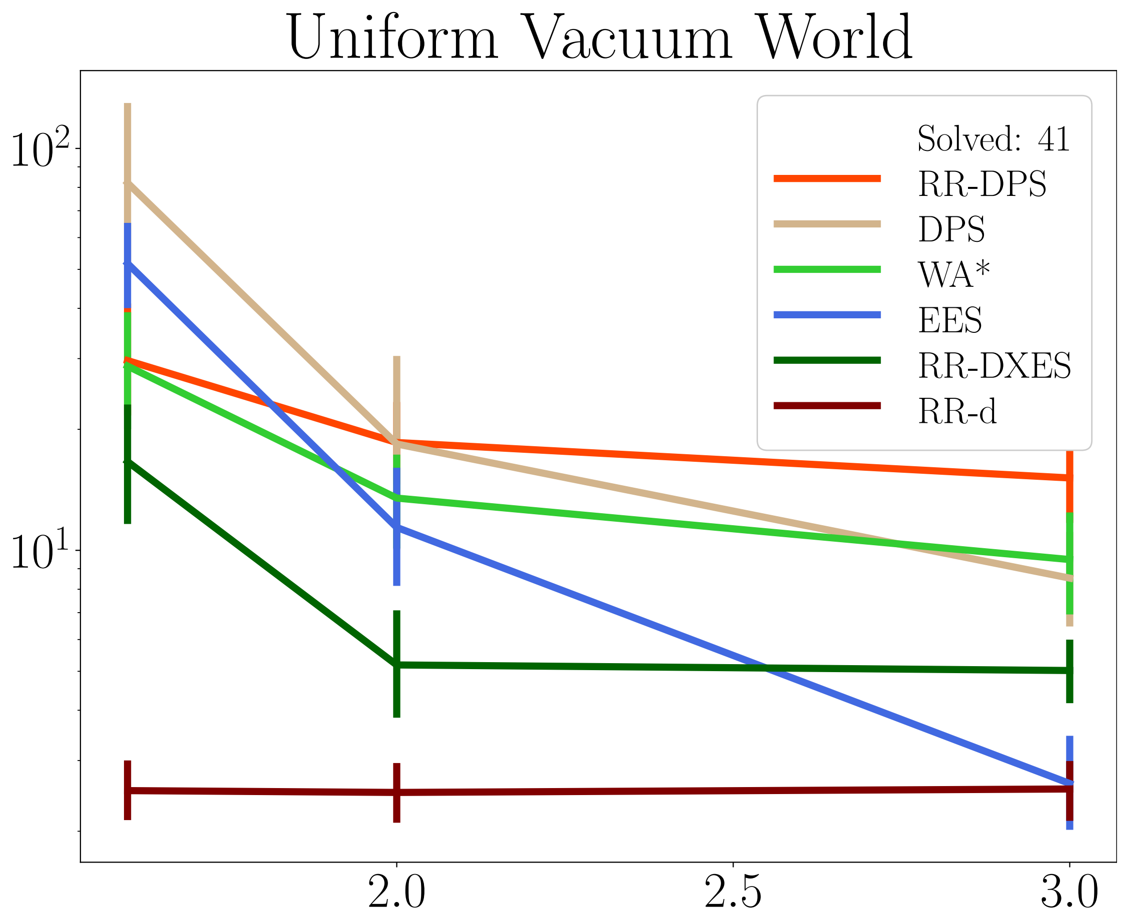
- Sliding-Tile Puzzle, Vaccum World, Pancake, Racetrack

IPC Coverage ($\omega = 1.5$)

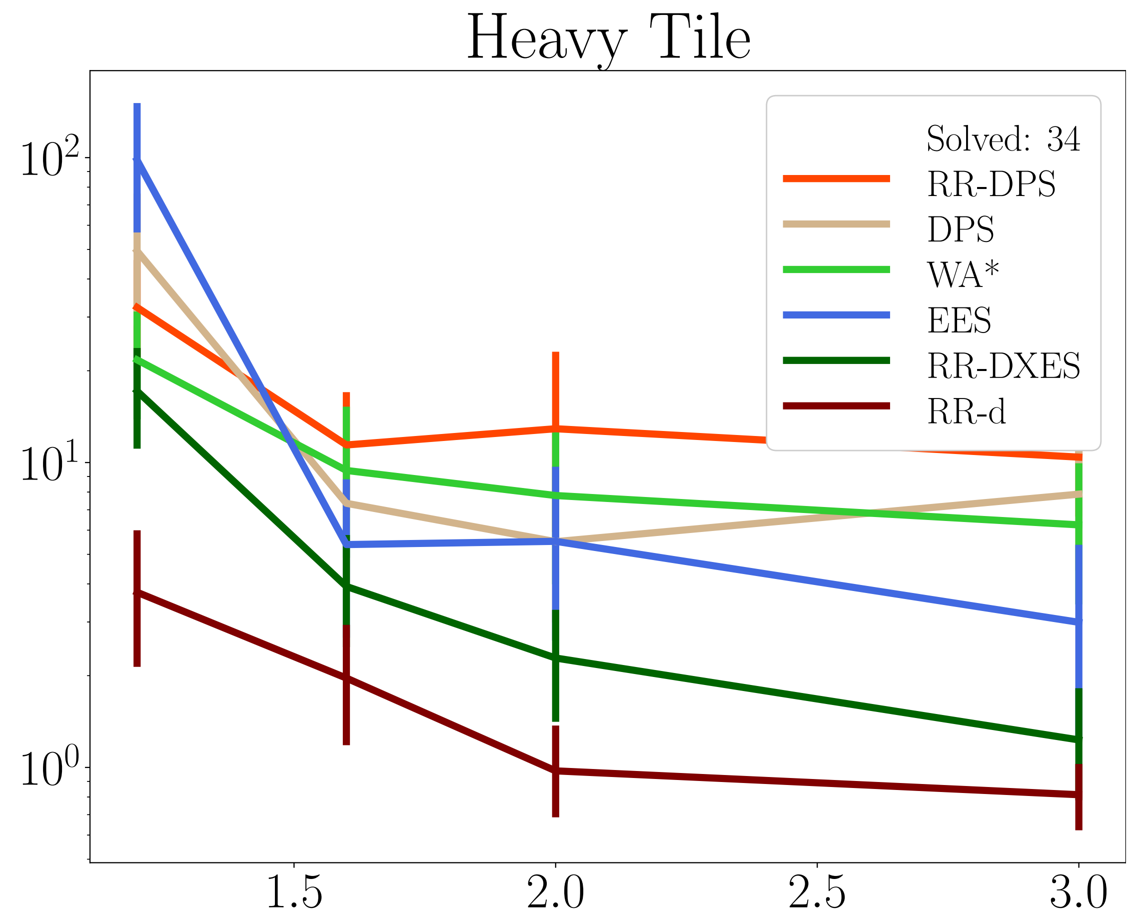
Coverage	WA*	EES	DPS	DXES	RR-DPS	RR-d	RR-DXES
Sum (1652)	995	967	1012	894	982	1025	**1052**
Normalized(%)	58.7	57.0	60.0	51.5	57.9	60.7	**62.5**
Expansions	569	558	472	734	665	383	**371**

$\rightarrow$ RR-DXES and RR-d perform best overall.

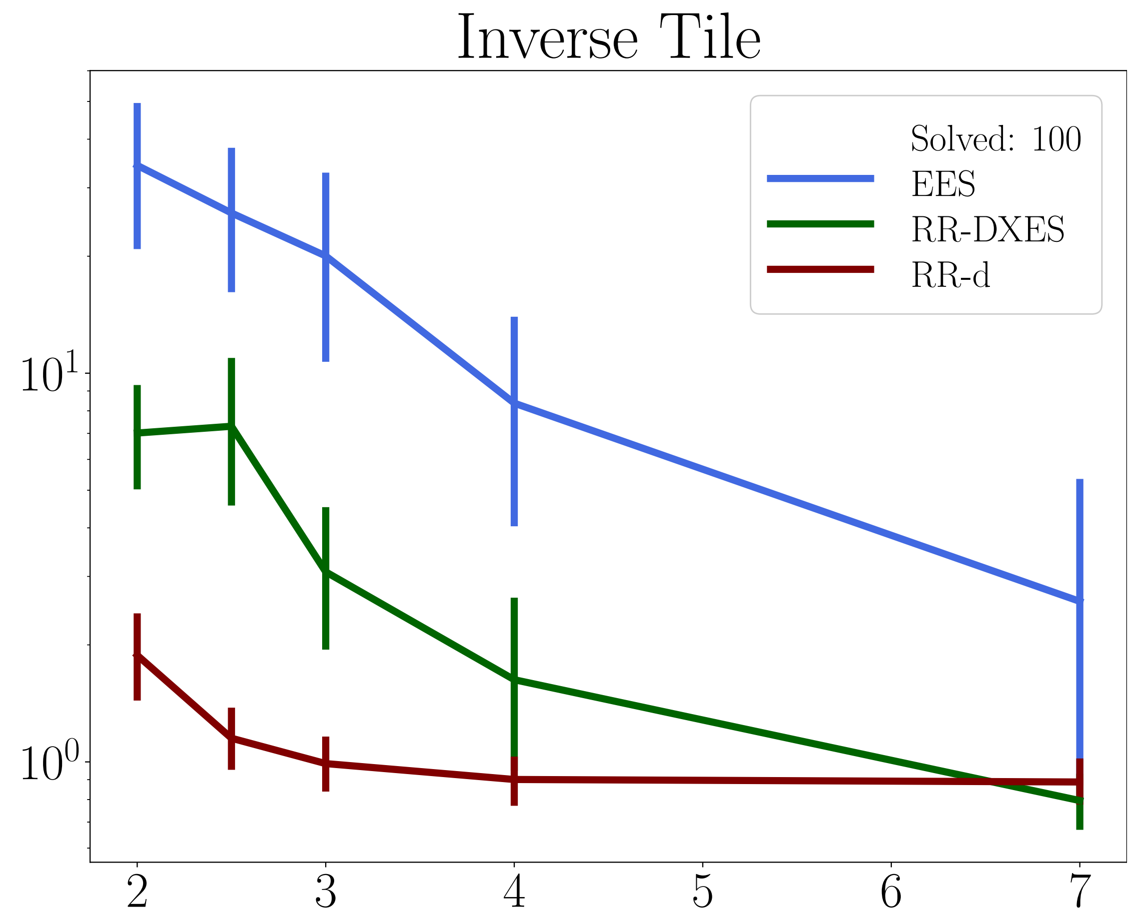
Search Domains



Search Domains



Search Domains



Conclusions

Summary

What to do for bounded-suboptimal search:

- Weighted-A* is the first thing to try
- Round-Robin on d, $\hat{f}$, f is the next to try
- Round-Robin on xe, $\hat{f}$, f perfom well in some domains

Summary

What to do for bounded-suboptimal search:

- Weighted-A* is the first thing to try
- Round-Robin on d, $\hat{f}$, f is the next to try
- Round-Robin on xe, $\hat{f}$, f perfom well in some domains

Still unresolved:

- When to raise bound, and when to pursue solution?
- How to best use belief distribution in bounded-suboptmal search?

Questions?