# New Results in Bounded-Suboptimal Search

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

1

2

# Planning as Heuristic Graph Search

Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

heuristic search: a planning approach

# Planning as Heuristic Graph Search

Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

heuristic search: a planning approach

planning is a model-based AI method, it models the environment as a state space and finds a sequence of actions that accomplishes some objective

## Planning as Heuristic Graph Search

Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

heuristic search: a planning approach

planning is a model-based AI method, it models the environment as a state space 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

Introduction
- Heuristic Search
- Problem Settings
- Overview

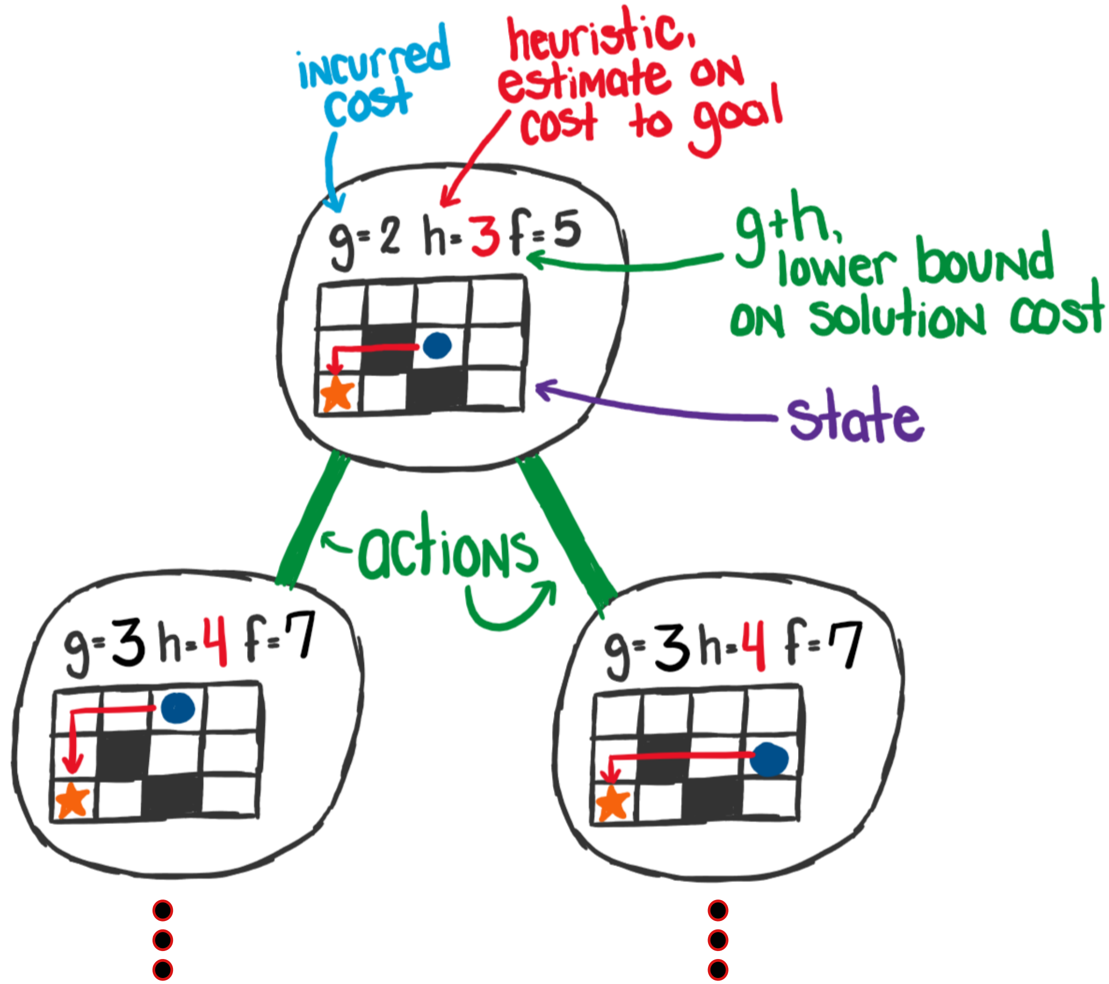
Bounded Suboptimal

New Algorithms

Results

Conclusions


heuristic search associates costs with states, used to guide search



# Planning as Heuristic Graph Search

Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

**A*:** expands the node with minimal $f$ value

returns optimal path

optimal search can take too long!

because it must expand every node with $f < C^*$[1]

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

Introduction
Heuristic Search
Problem Settings
Overview
Bounded Suboptimal
New Algorithms
Results
Conclusions

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

What if we don't have time?

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

# Alternatives to Optimal Search: Problem Settings

Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

**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

# Alternatives to Optimal Search: Problem Settings


Introduction
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions


**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
- Heuristic Search
- Problem Settings
- Overview

Bounded Suboptimal

New Algorithms

Results

Conclusions

- Introduction
- Bounded-Suboptimal Search
- New Algorithms
  - DXES
  - RoundRobin
- Results
- Conclusions

Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions

# Bounded-Suboptimal Search

# Bounded-Suboptimal Search: The Problem Setting

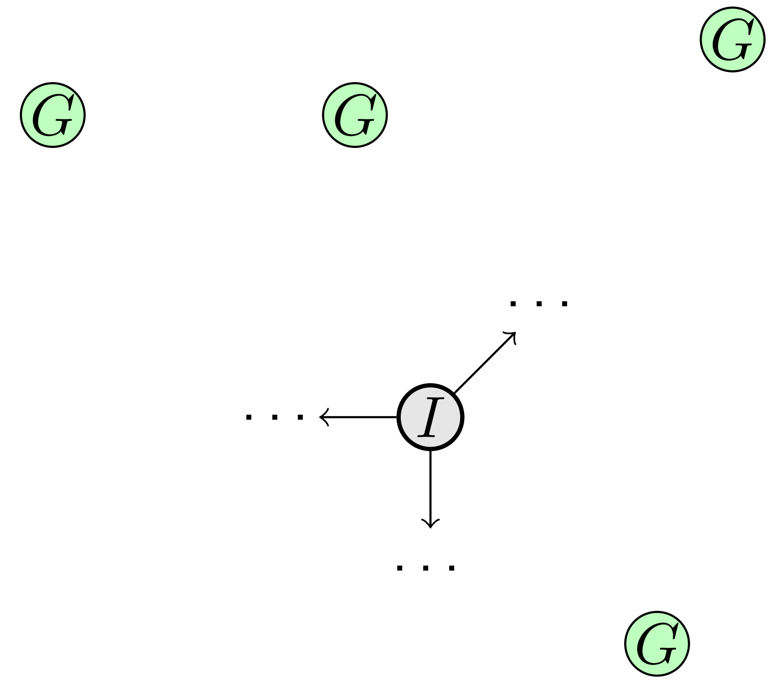

Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

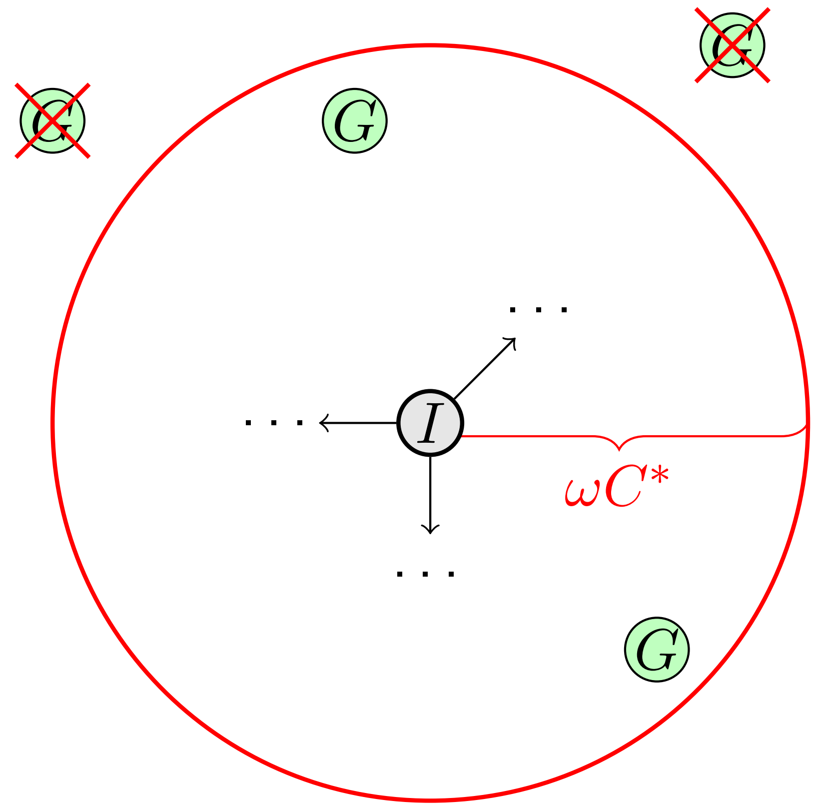
New Algorithms

Results

Conclusions

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES
New Algorithms
Results
Conclusions




*Objective: Find a plan with cost at most* $\omega C^*$ *as fast as possible.*

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES
New Algorithms
Results
Conclusions

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)

Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions

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)

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES
New Algorithms
Results
Conclusions

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)

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms
Results
Conclusions

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.

Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions

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)


Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions


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)


Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions


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)

Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions

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** $best_f$

Problems:

- EES does not consider the uncertainty of its estimates (brittle)
- EES does not spend enough effort on estimating the bound

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES
New Algorithms
Results
Conclusions

Best first search on "potential":

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

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


Introduction

Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES

New Algorithms

Results

Conclusions


Best first search on "potential":

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

does not explicitly optimize search time

Introduction
Bounded Suboptimal
- Problem Setting
- EES
- DPS
- XES
New Algorithms
Results
Conclusions

Best first search on expected search effort:

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

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

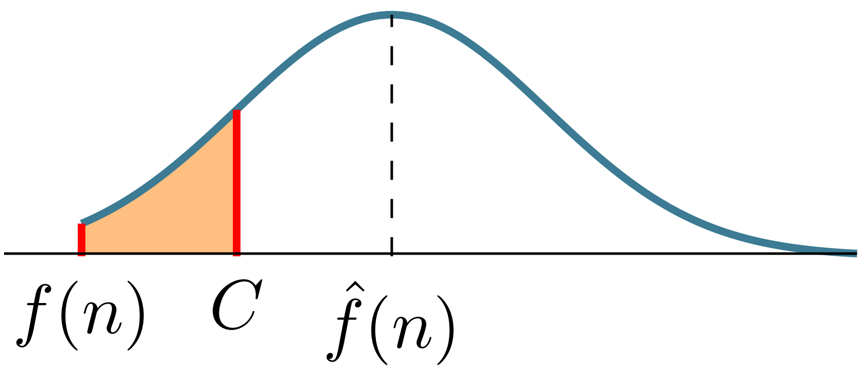


Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted by $d(n)$

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



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

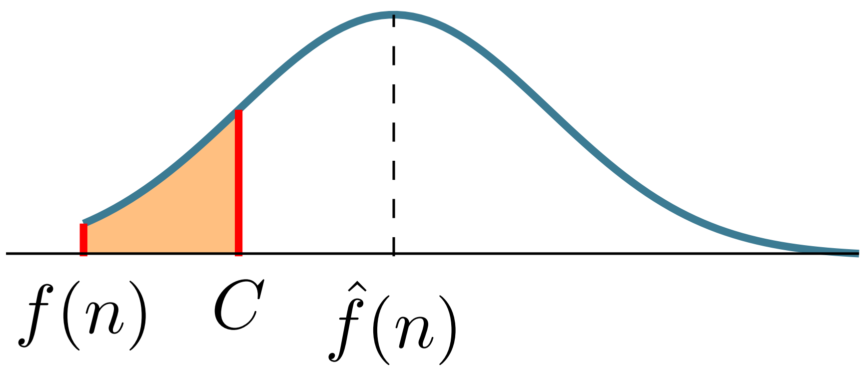


Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted by $d(n)$

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




XES performs better than BEES and PS.
Can we adpat XES to bounded-suboptimal setting?

Introduction

Bounded Suboptimal

New Algorithms
- DXES
- RoundRobin

Results

Conclusions

# New Algorithms

# Our Approach: 1/2 Dynamic XES

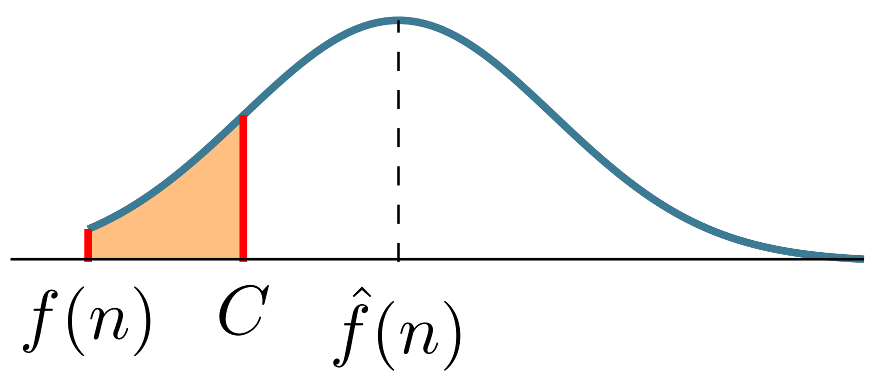
Introduction
Bounded Suboptimal
New Algorithms
- DXES
- RoundRobin
Results
Conclusions

Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted by $d(n)$

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




hard to estimate when raising the bound is useful!

# Our Approach: 1/2 Dynamic XES

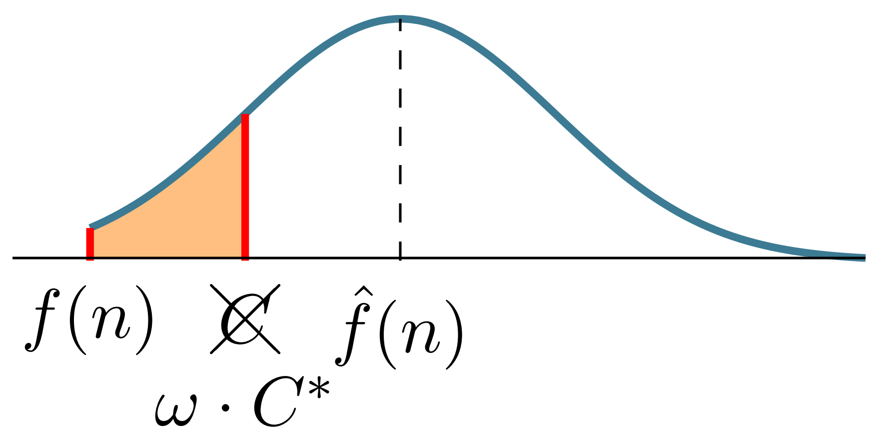


Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted 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: 1/2 Dynamic XES

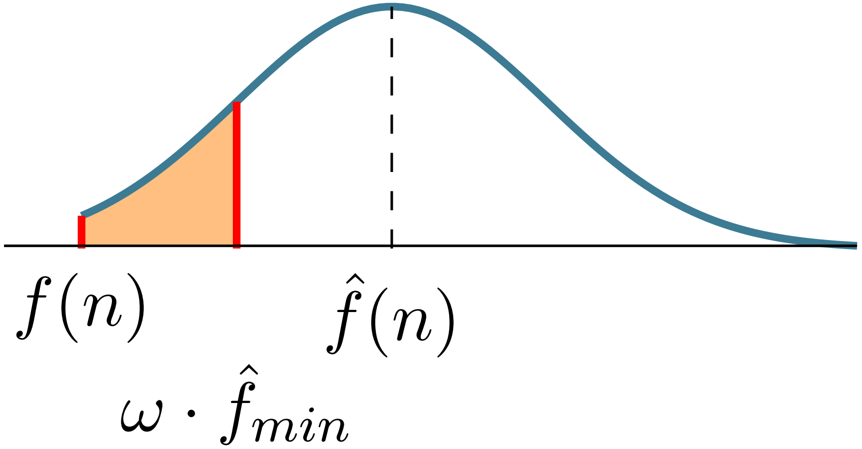

Introduction
Bounded Suboptimal
New Algorithms
- DXES
- RoundRobin
Results
Conclusions


Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted 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: 1/2 Dynamic XES

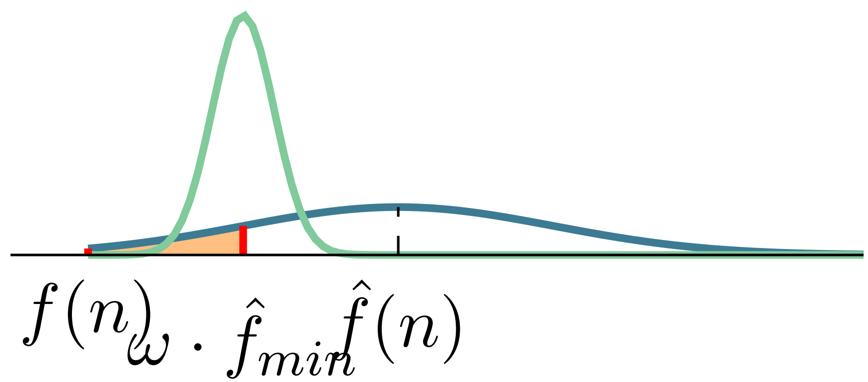


Best first search on expected search effort:

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

$T(n)$: totoal search effort, estiamted 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


Introduction

Bounded Suboptimal

New Algorithms

- DXES
- RoundRobin

Results

Conclusions


Round-Robin on:

**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 work well emparically!

Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search
Conclusions

# Results

Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search
Conclusions

Planning Domains:

- Implementation in Fast Downward[2]
- Benchmarks:
  IPS optimal tracks (48 domains)

Search Domains:

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

[2] Helmert 2006.

# IPC Coverage ($\omega = 1.5$)


Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search

Conclusions

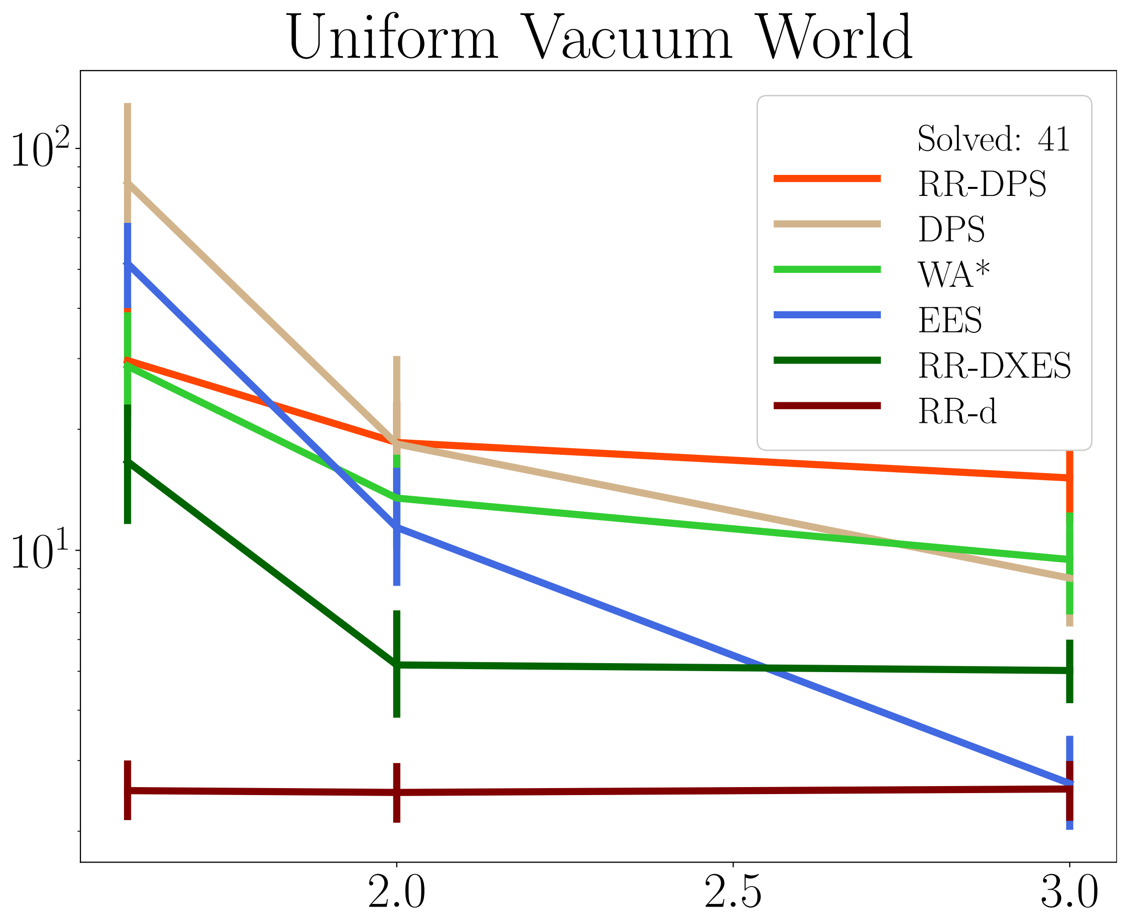

| Coverage | WA* | EES | DPS | DXES | RR-$d$ | RR-DPS | RR-DXES |
|---|---|---|---|---|---|---|---|
| **Sum (1652)** | 995 | 967 | 1012 | 894 | 1025 | 982 | **1052** |
| Normalized(%) | 58.7 | 57.0 | 60.0 | 51.5 | 60.7 | 57.9 | **62.5** |
| Expansions | 569 | 558 | 472 | 734 | 383 | 665 | **371** |

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

# Search Domains


Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search

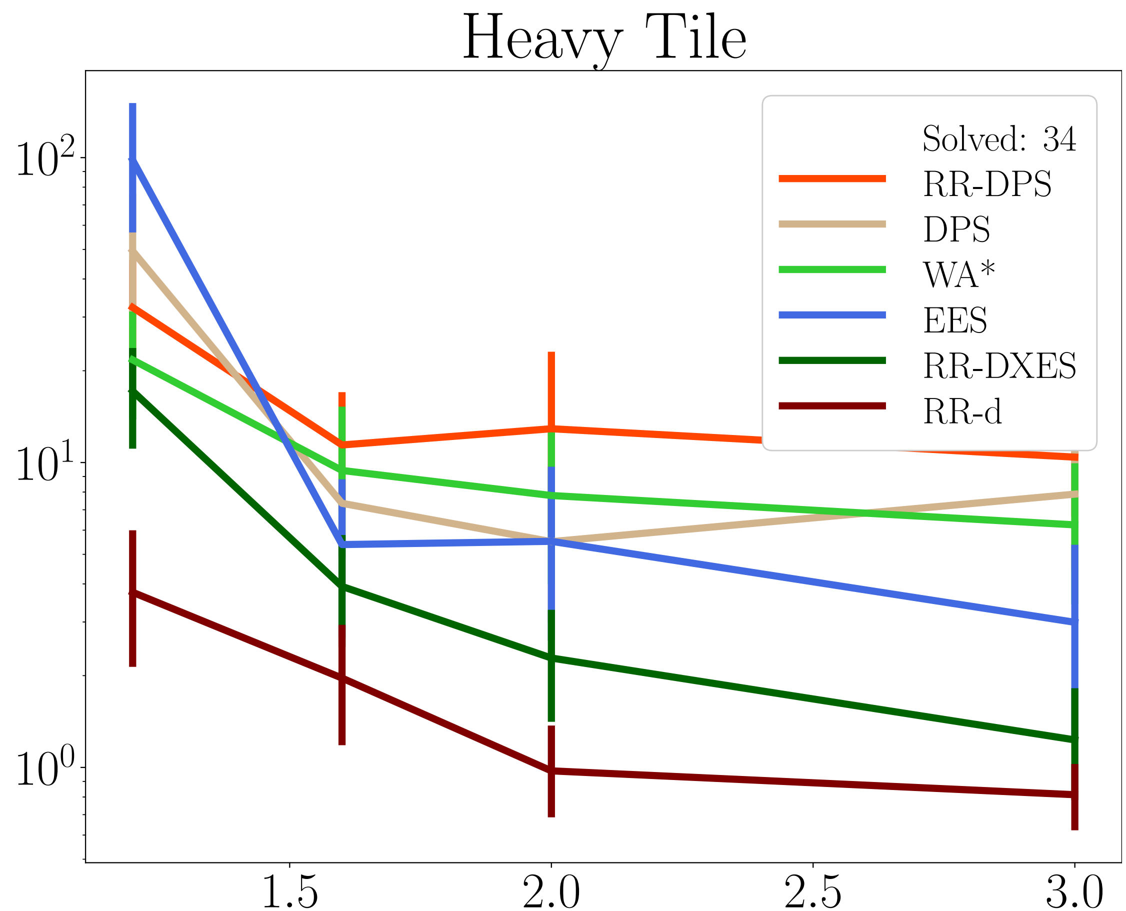
Conclusions

# Search Domains

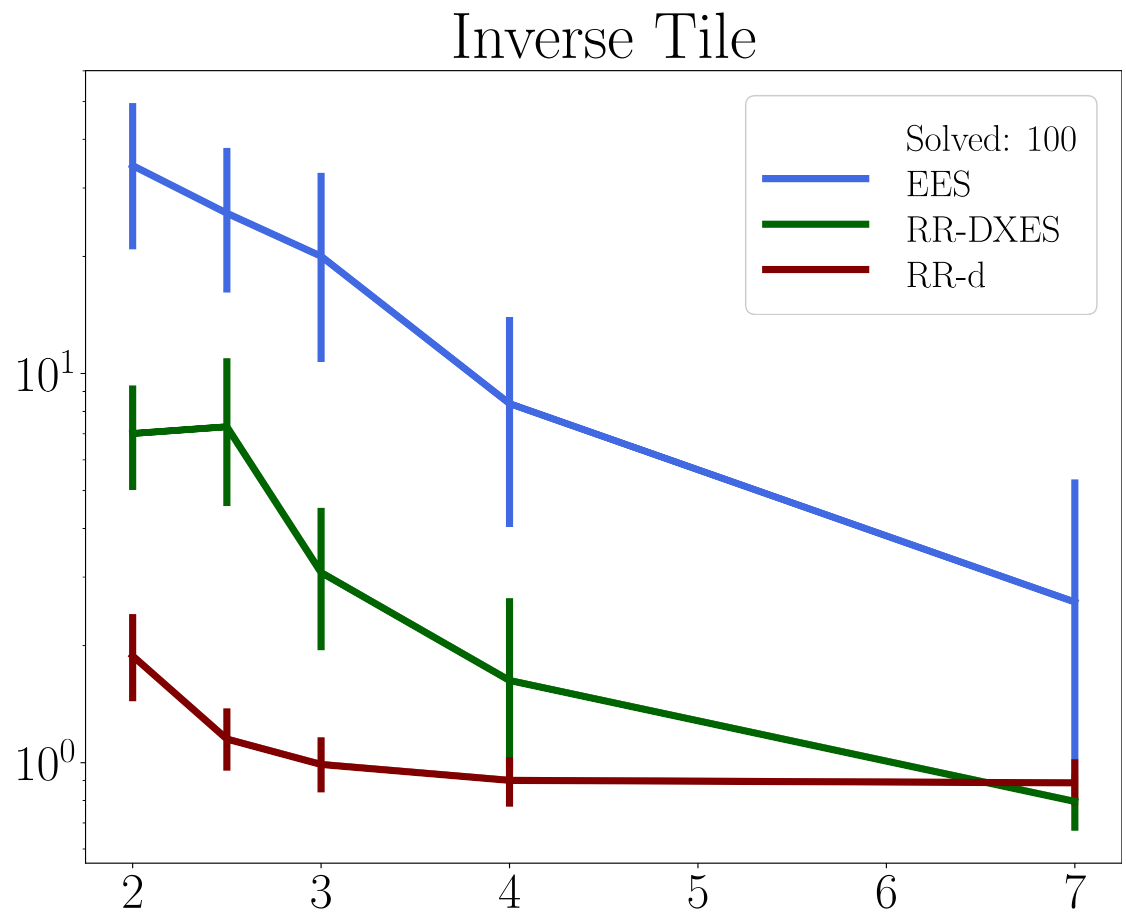
Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search

Conclusions



# Search Domains

Introduction
Bounded Suboptimal
New Algorithms
Results
- Experiments
- Planning
- Search
Conclusions



Introduction
Bounded Suboptimal
New Algorithms
Results
Conclusions
- Summary

# Conclusions

# Summary

Introduction
Bounded Suboptimal
New Algorithms
Results
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

# Summary

Introduction
Bounded Suboptimal
New Algorithms
Results
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

What's next:

- When to raise bound, and when to pursue solution?
- DXES should help, but not quite, why?

# Questions?

Introduction
Bounded Suboptimal
New Algorithms
Results
Conclusions
Questions
Questions?