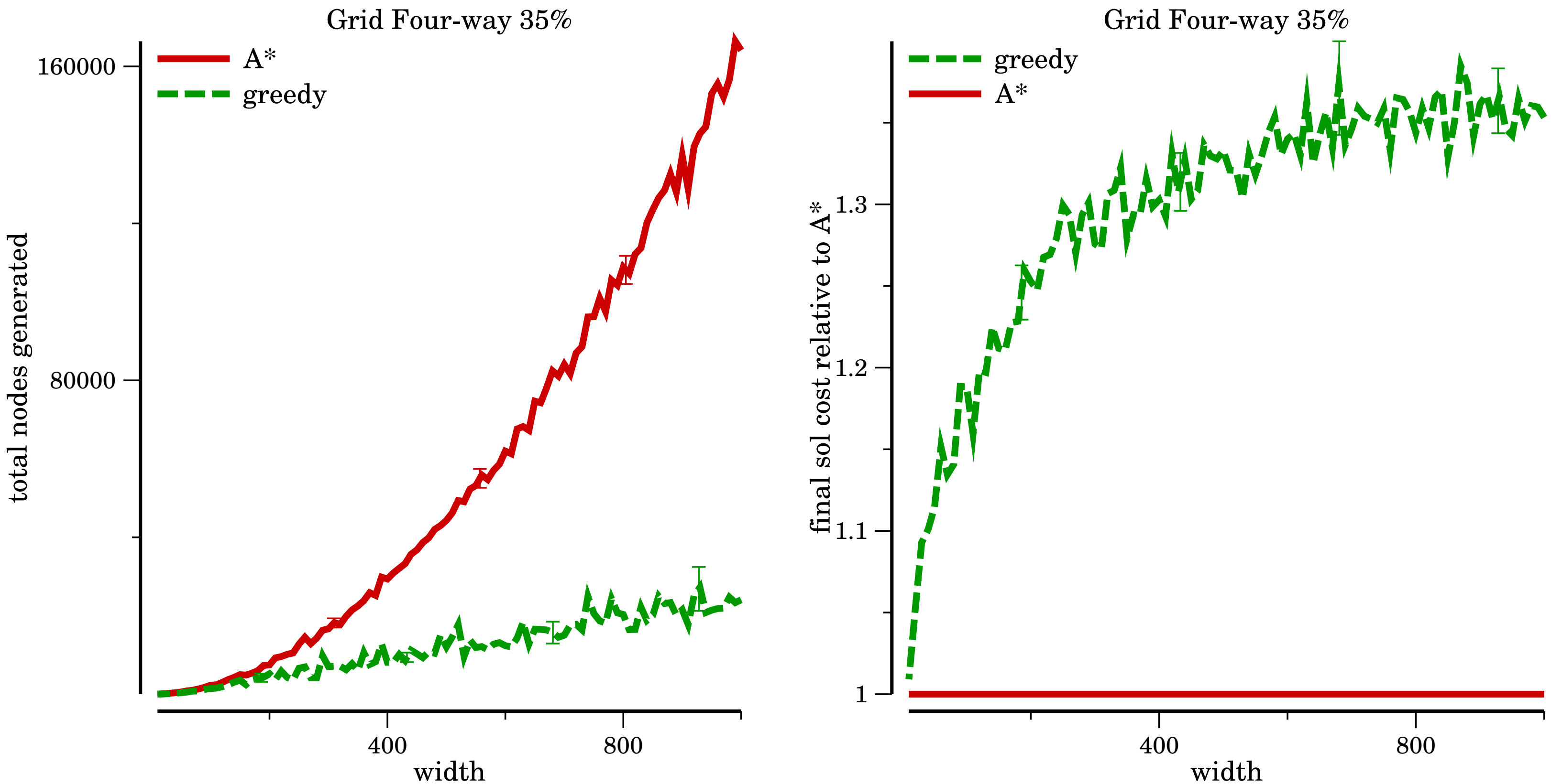
Faster Bounded-Cost Search Using Inadmissible Estimates

Jordan Thayer, Roni Stern, Ariel Felner, Wheeler Ruml

UNIVERSITY of NEW HAMPSHIRE

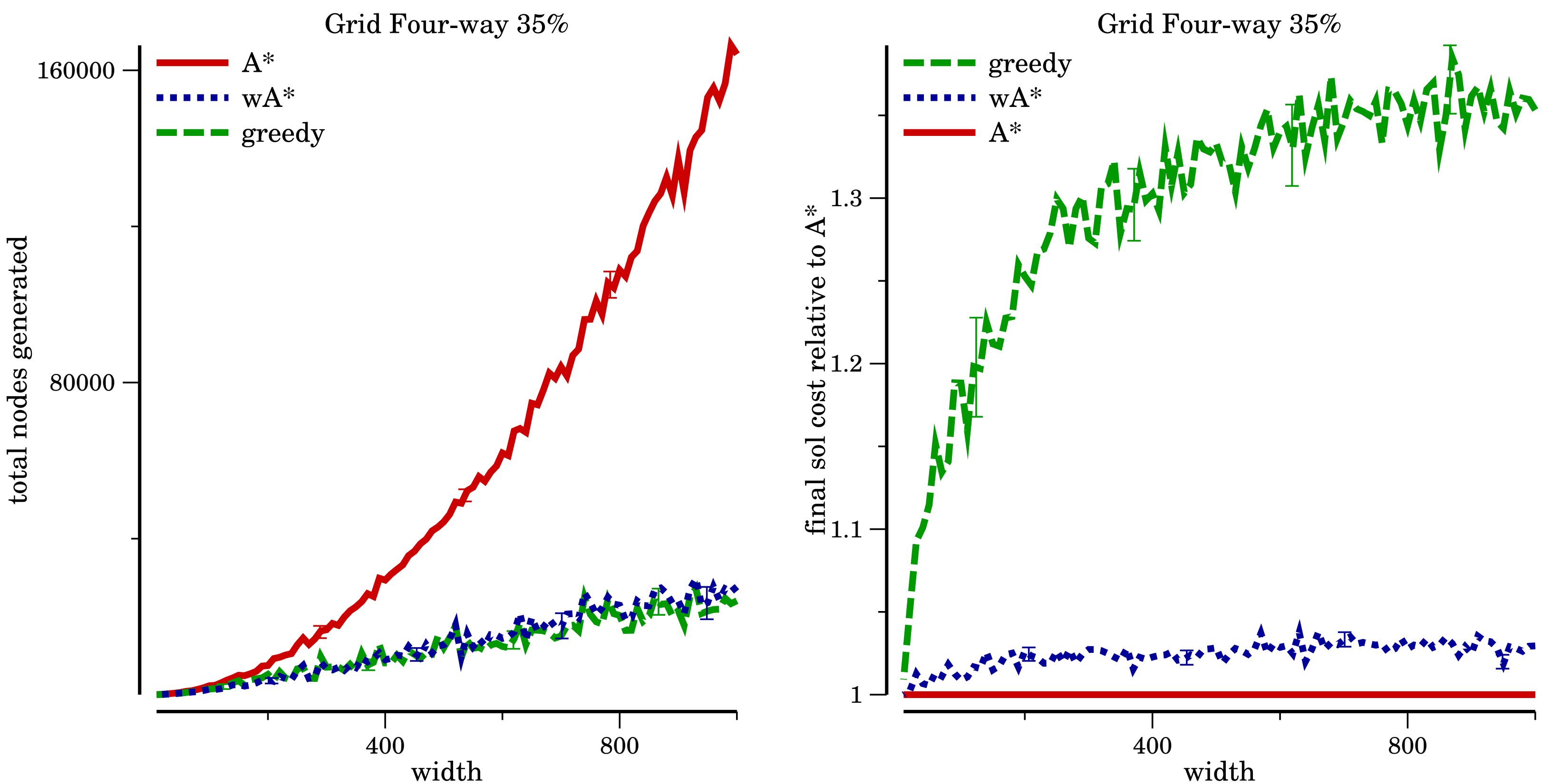
jthayer@sift.net roni.stern@gmail.com felner@bgu.ac.il ruml@cs.unh.edu

Motivation: Optimal Search Hard, Greedy Solutions Expensive



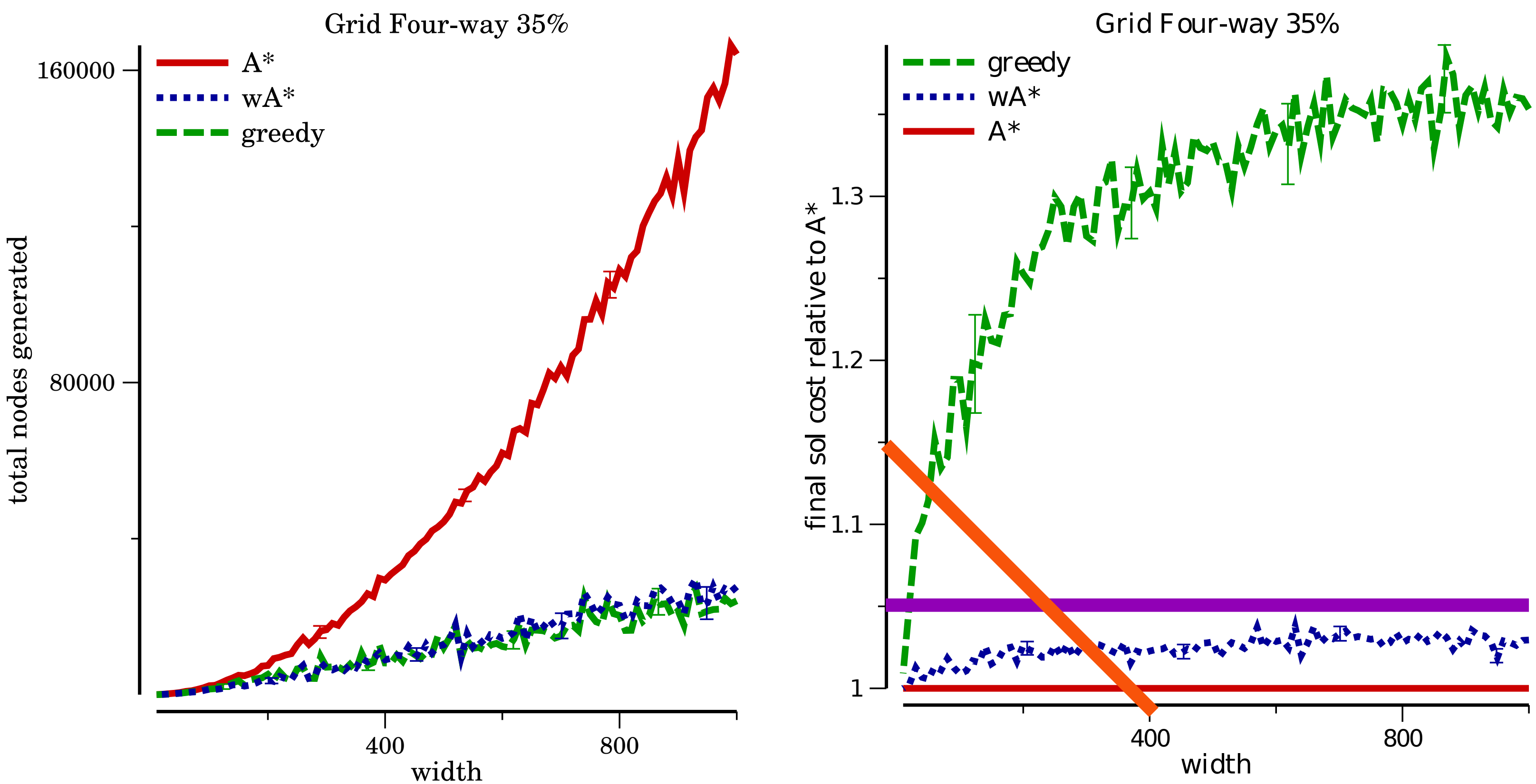
optimal search won't scale
greedy solutions too expensive

Motivation: Bounded Suboptimal Provides Middle Ground



impose bounds to limit cost (relative to optimal)

Motivation: What if I have a Budget instead of w?



What if we have a budget instead of a relative bound?

Bounded Cost Search: find a solution of cost no more than C

Outline

- motivation
- previous approach: potential search
- problem with potential search
- BEES
- bigger picture
- PTS on inadmissible heuristics, BEEPS

best-first search in order of chance of satisfying cost bound C

best-first search in order of chance of satisfying cost bound C

$$PT_C(n) = Pr(g(n) + h^*(n) \leq C)$$

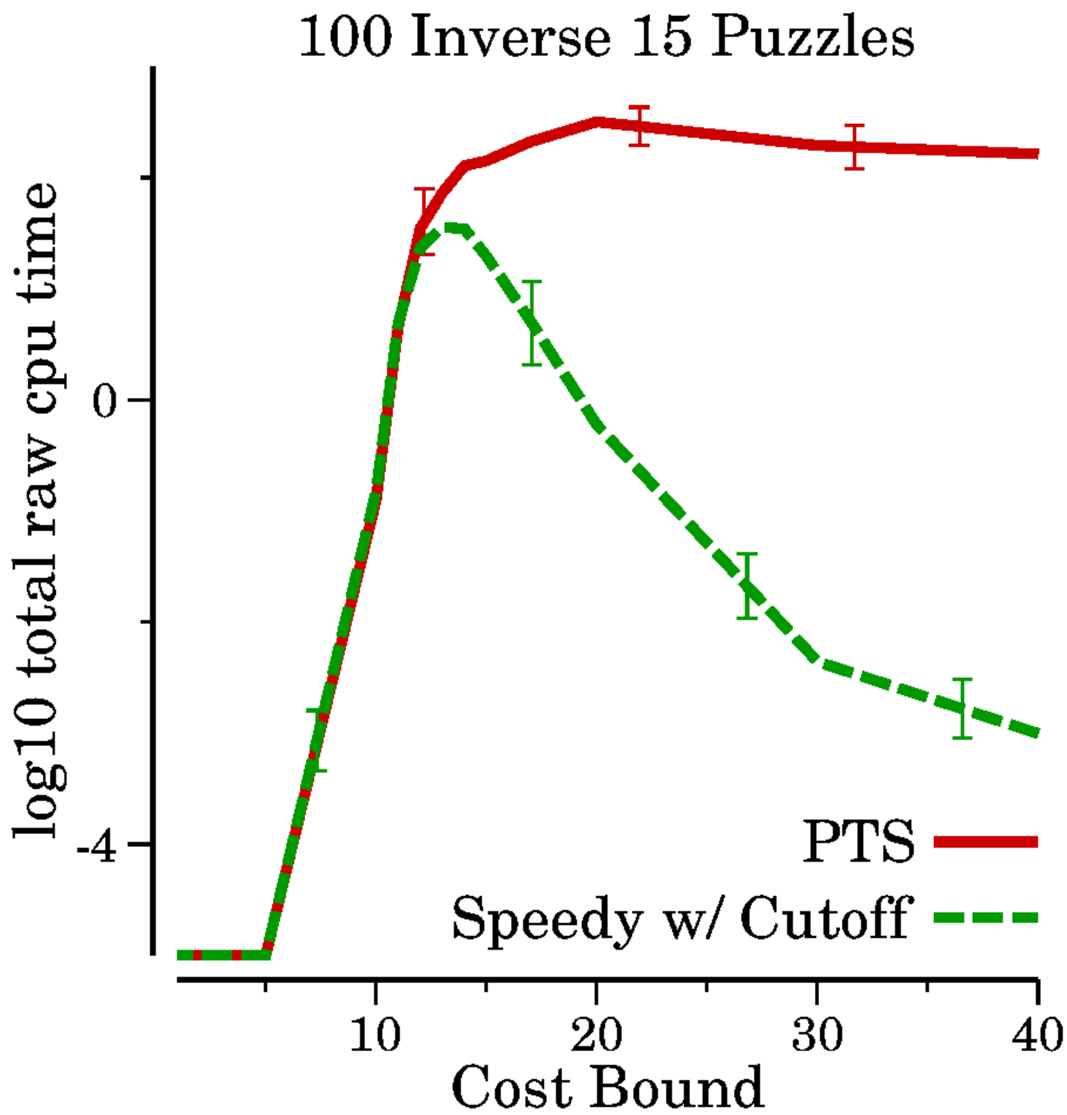
best-first search in order of chance of satisfying cost bound C

$$PT_C(n) = Pr(g(n) + h^*(n) \leq C)$$

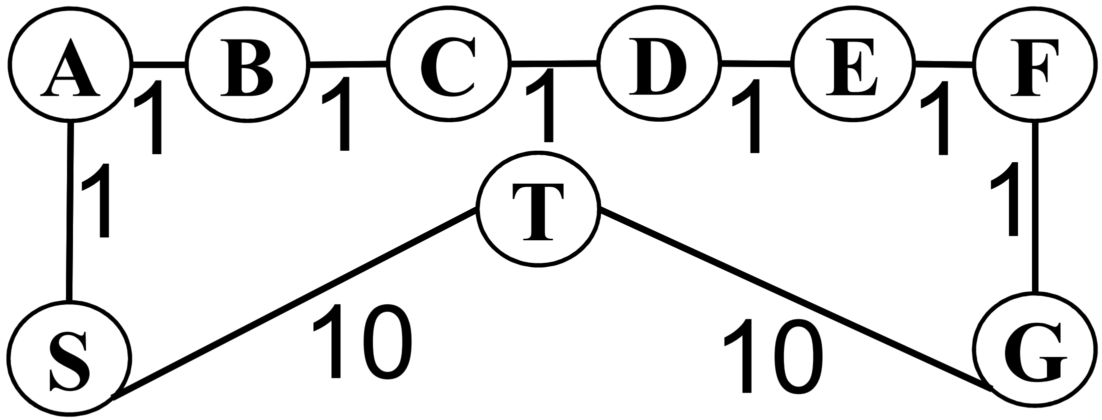
unfortunately, we may not be able to compute that

$f_{lnr}(n) = \frac{h(n)}{C-g(n)}$ produces an equivalent order
under certain assumptions

PTS Performance: Non-Unit Cost Performance is Bad!

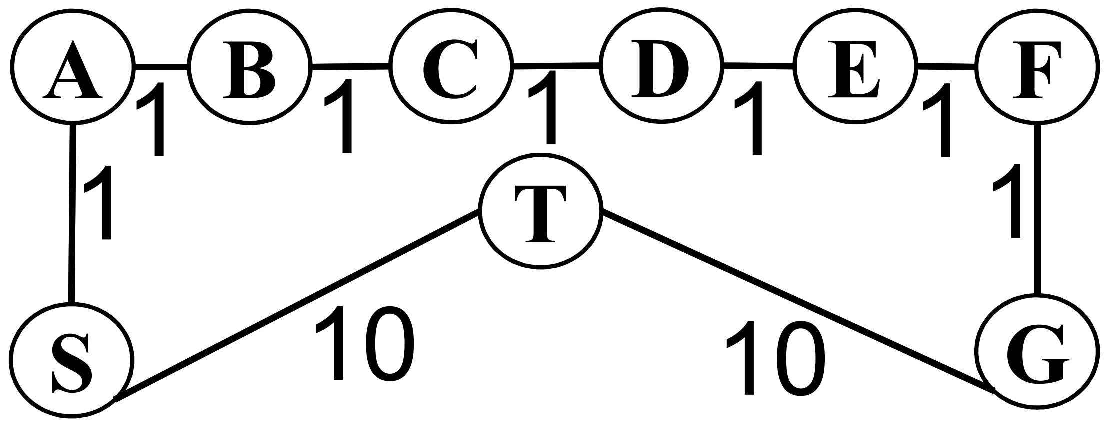


Potential Search Ignores Solution Length



$d(A) = 6,\ h(A) = 6$
$d(T) = 1,\ h(T) = 10$
$C = 20$

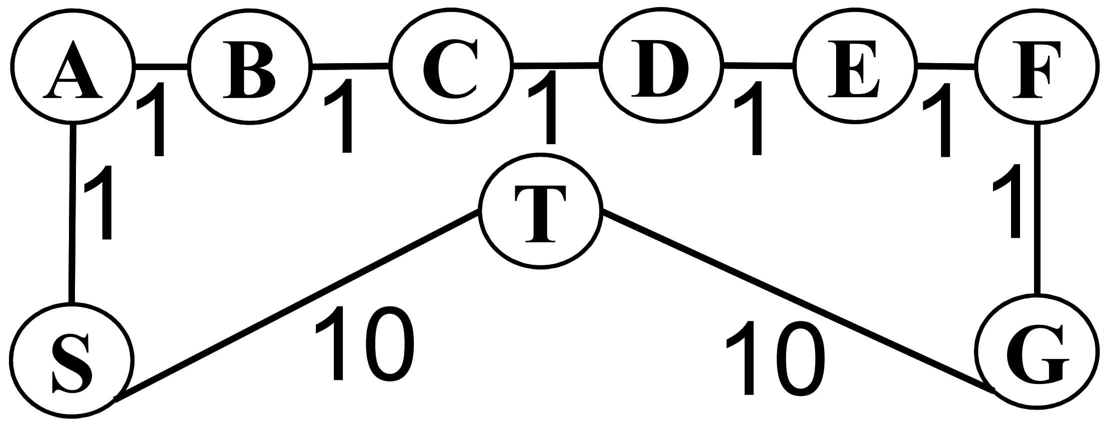
Potential Search Ignores Solution Length



$d(A) = 6,\ h(A) = 6$
$d(T) = 1,\ h(T) = 10$
$C = 20$

$f_{lnr}(A) = \frac{6}{20-1}$
$f_{lnr}(T) = \frac{10}{20-10}$

Potential Search Ignores Solution Length



$d(A) = 6,\ h(A) = 6$
$d(T) = 1,\ h(T) = 10$
$C = 20$

$f_{lnr}(A) = \frac{6}{20-1}$
$f_{lnr}(T) = \frac{10}{20-10}$

PTS prefers A to T

Bounded Cost Explicit Estimation Search

Goals for New Bounded Cost Search

1. avoid reliance on error models (ie f_{lnr})
2. improve performance on non-unit cost domains

   without losing performance in unit cost domains

Bounded-Cost Explicit Estimation Search (BEES)

1. estimate which nodes are within cost bound

2. best-first search of these on estimated actions-to-go

Step 1: Estimate which nodes are within cost bound

$h(n)$: an admissible cost-to-go estimate

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

$f(n) \leq C$: could lead to a solution in bound

focal $= \{n \in open | f(n) \leq C\}$

Step 1: Estimate which nodes are within cost bound

$h(n)$: an admissible cost-to-go estimate
$\widehat{h}(n)$: best-guess estimate of cost-to-go

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

$f(n) \leq C$: could lead to a solution in bound
$\widehat{f}(n) \leq C$: probably leads to a solution in bound

focal $= \{n \in open | \widehat{f}(n) \leq C\}$

What If No Nodes Appear to Be Within Bound?

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

focal $= \{n \in open | \widehat{f}(n) \leq C\}$
open $= \{n | f(n) \leq C\}$

A* provides an efficient way to prove no solution in C

What If No Nodes Appear to Be Within Bound?

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

focal $= \{n \in open | \widehat{f}(n) \leq C\}$
open $= \{n | f(n) \leq C\}$

A* provides an efficient way to prove no solution in C

1. Estimate which nodes are within cost bound
2. Best-first search of these on estimated actions-to-go
3. A* search if we think no solution exists within C

Pseudo-Code for BEES

BEES is a best first search on the following rule

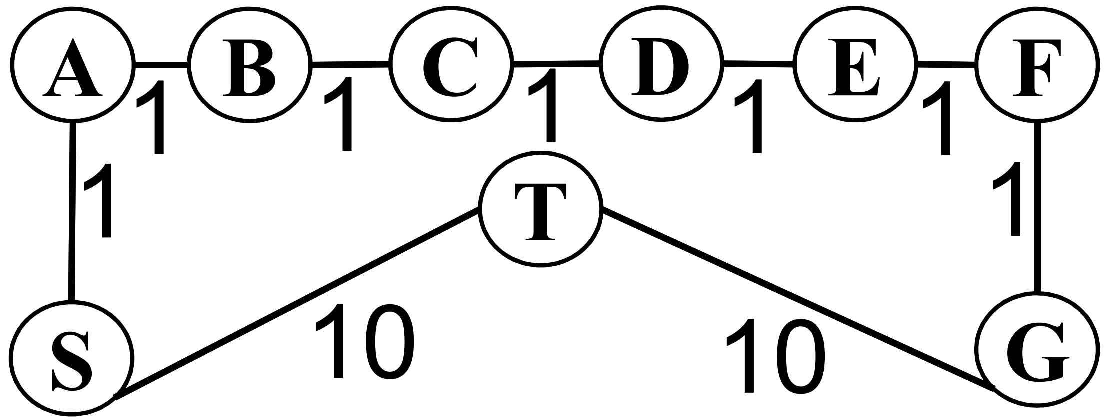
open $= \{n | f(n) \leq C\}$

focal $= \{n \in open | \widehat{f}(n) \leq C\}$

selectNode

1. **if** focal $\neq \{\}$
2. **then** return $n \in$ focal estimated nearest to a goal
3. **else** return $n \in$ open with minimum f

PTS vs. BEES on Explicit Graph

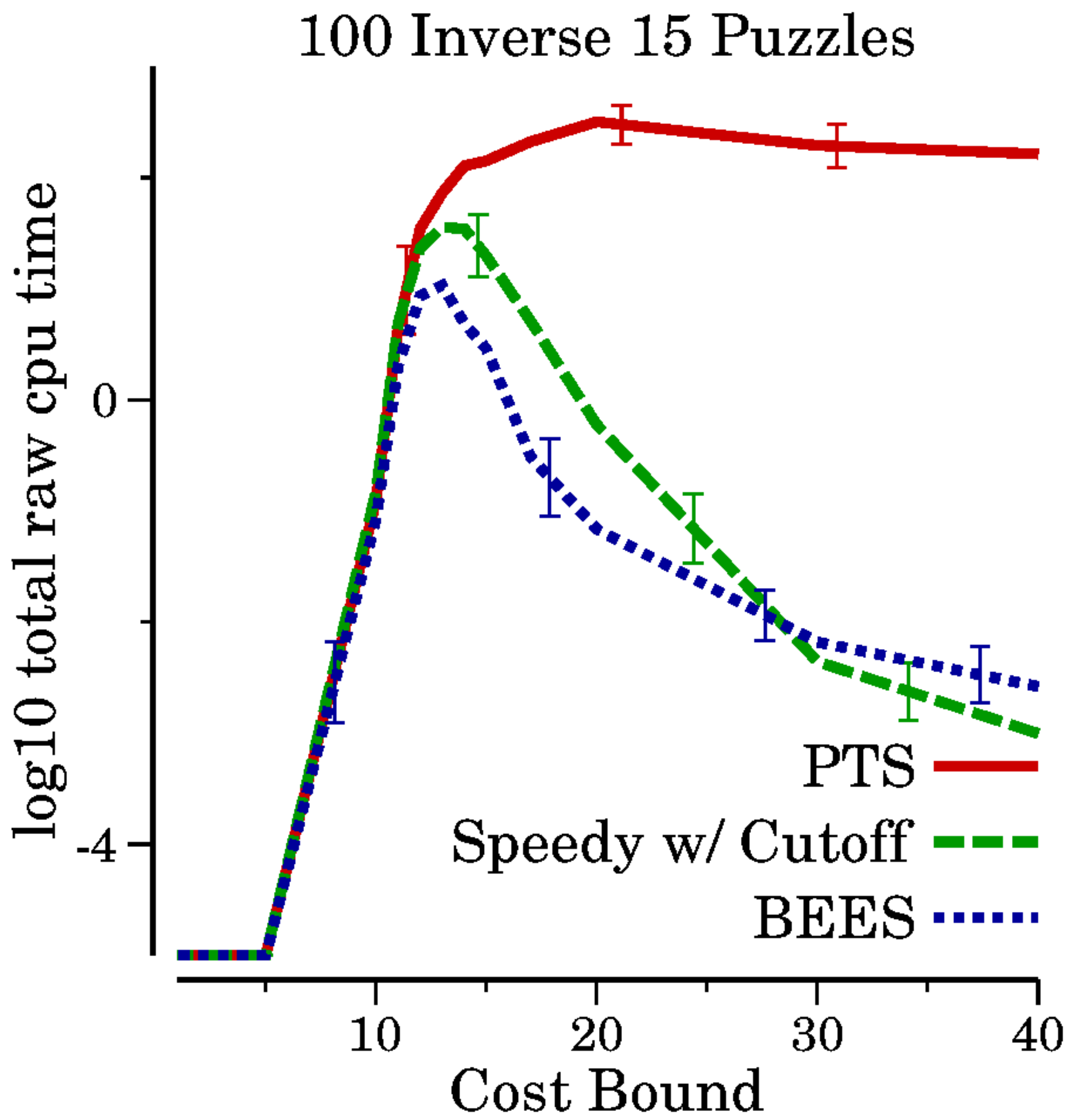


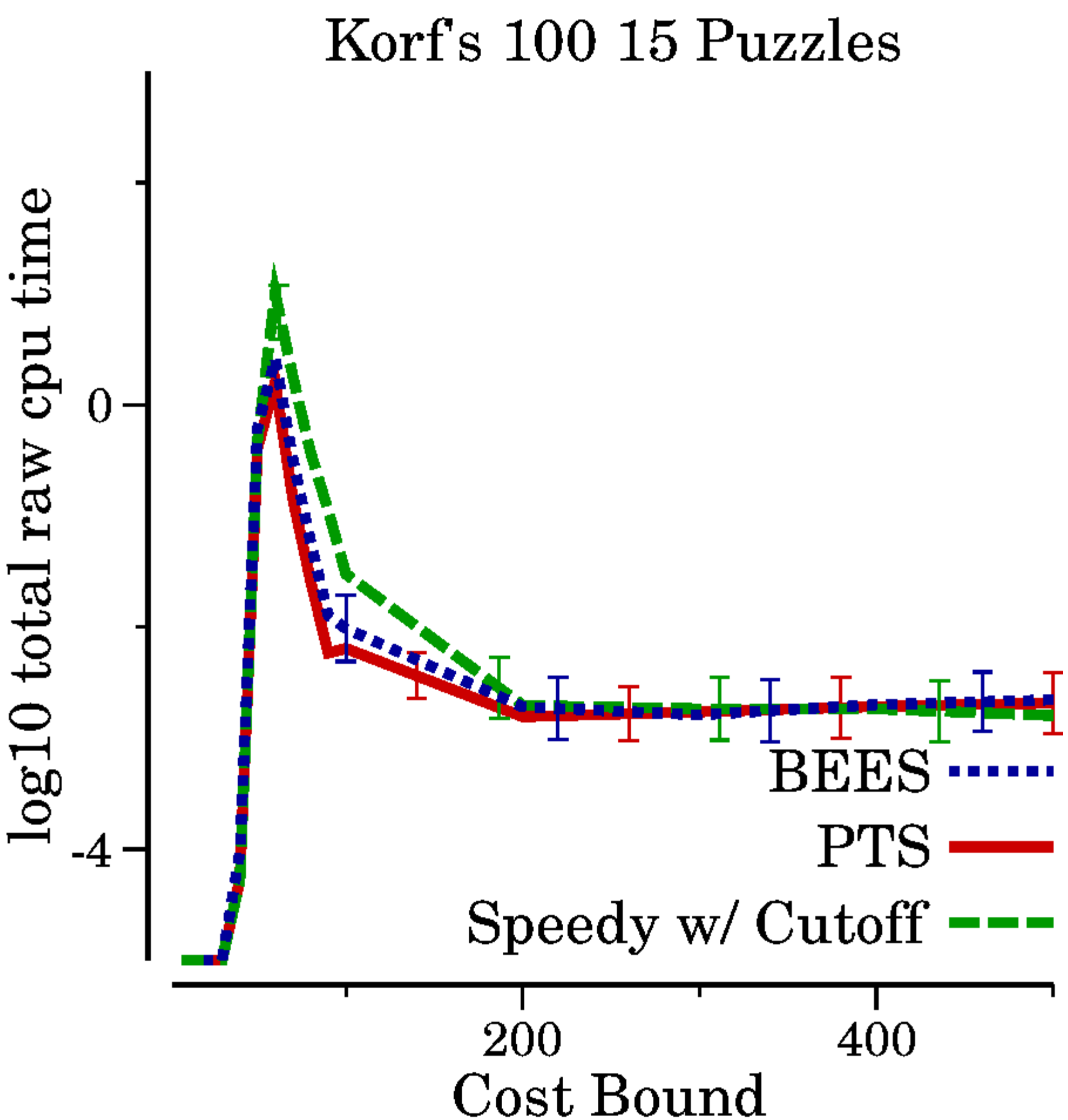
$d(A) = 6,\ h(A) = 6,\ \widehat{f}(A) = 7$
$d(T) = 1,\ h(T) = 10,\ \widehat{f}(T) = 20$
$C = 20\ \ f_{lnr}(A) = \frac{6}{20-1}$

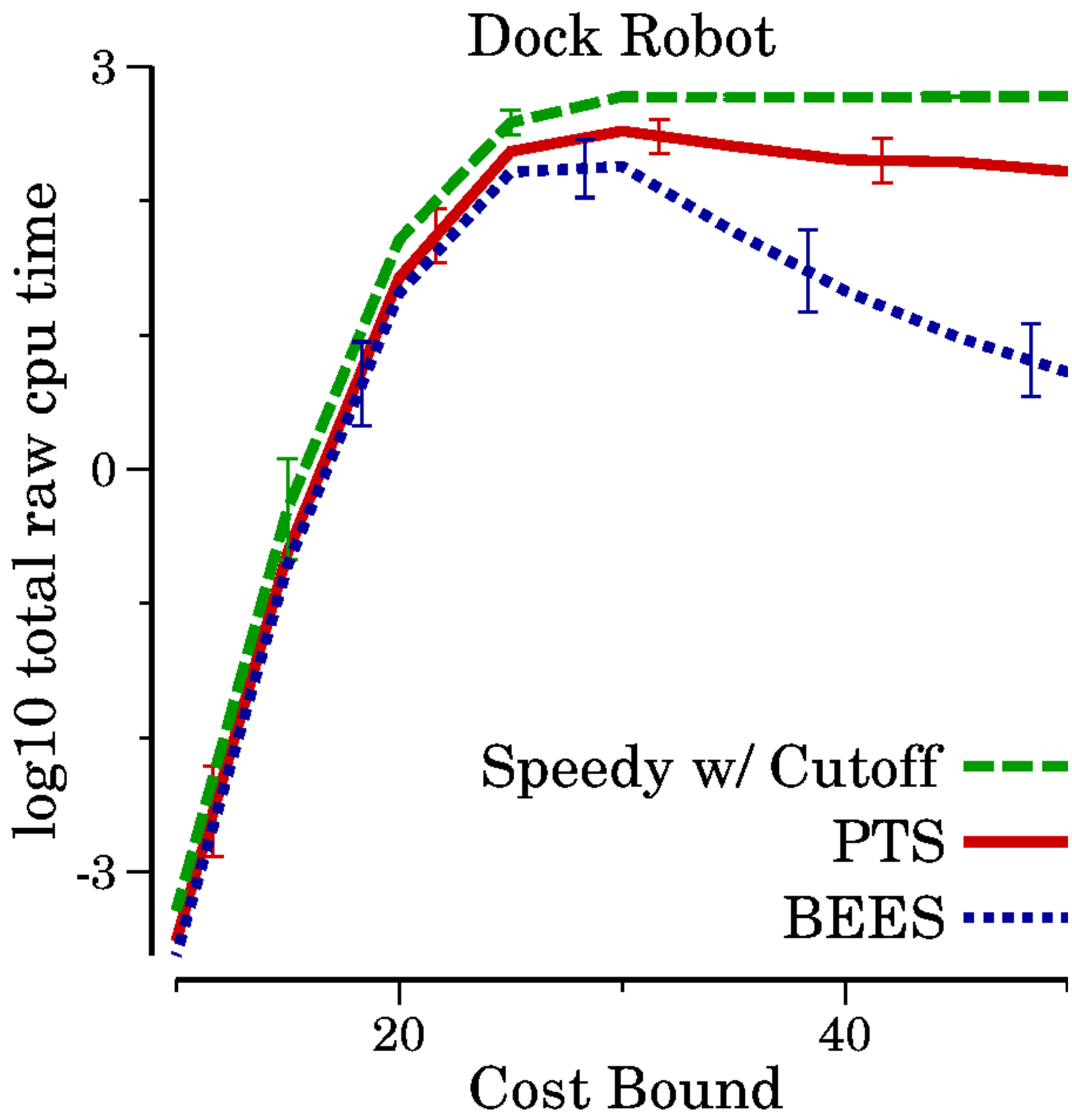
$f_{lnr}(T) = \frac{10}{20-10}$

BEES prefers T, PTS prefers A

Empirical Evaluation



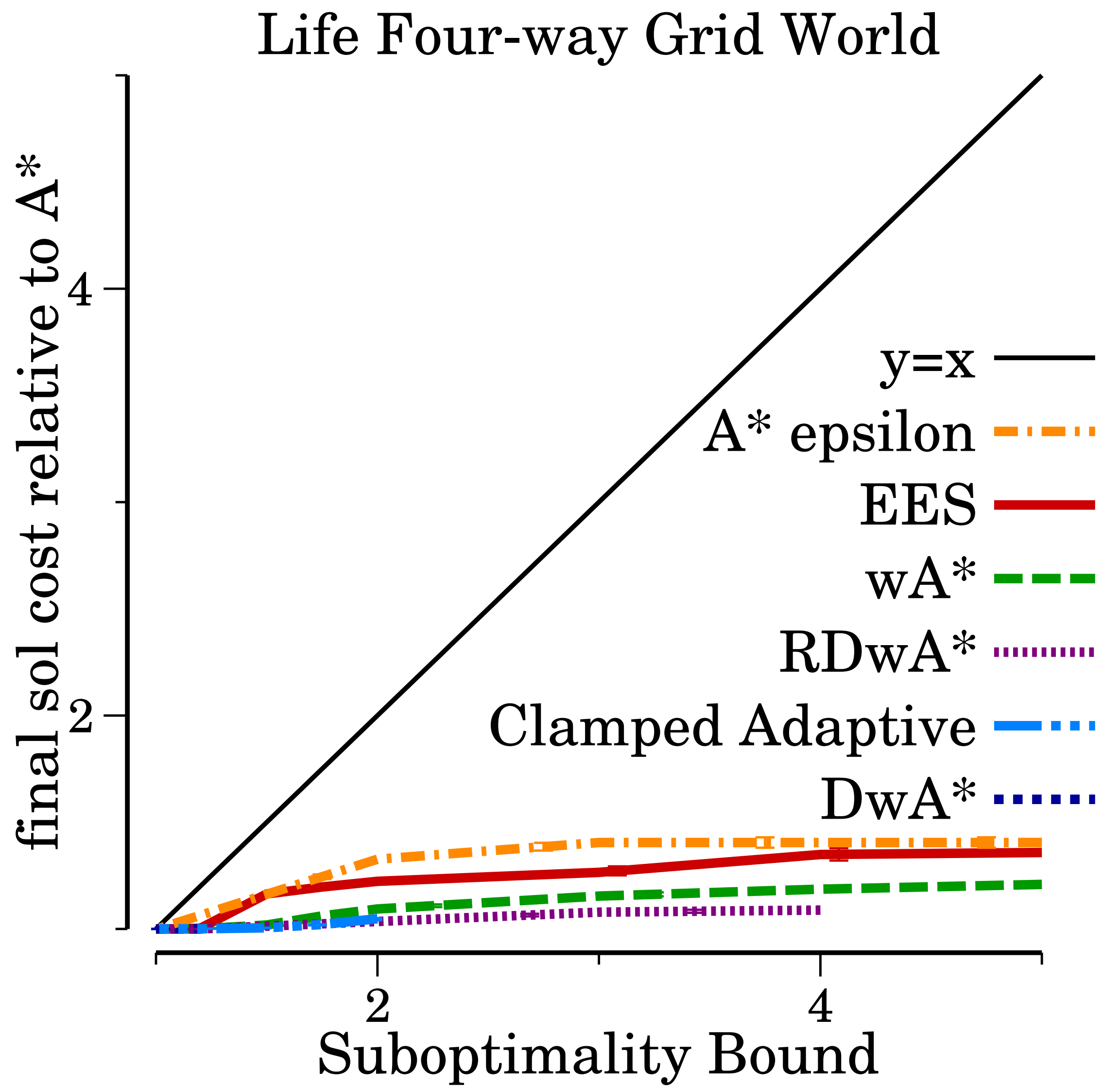

Korf's 100 15 Puzzles
log10 total raw cpu time
0
-4
200
400
Cost Bound
BEES
PTS
Speedy w/ Cutoff


Dock Robot
3
0
-3
log10 total raw cpu time
20
40
Cost Bound
Speedy w/ Cutoff
PTS
BEES

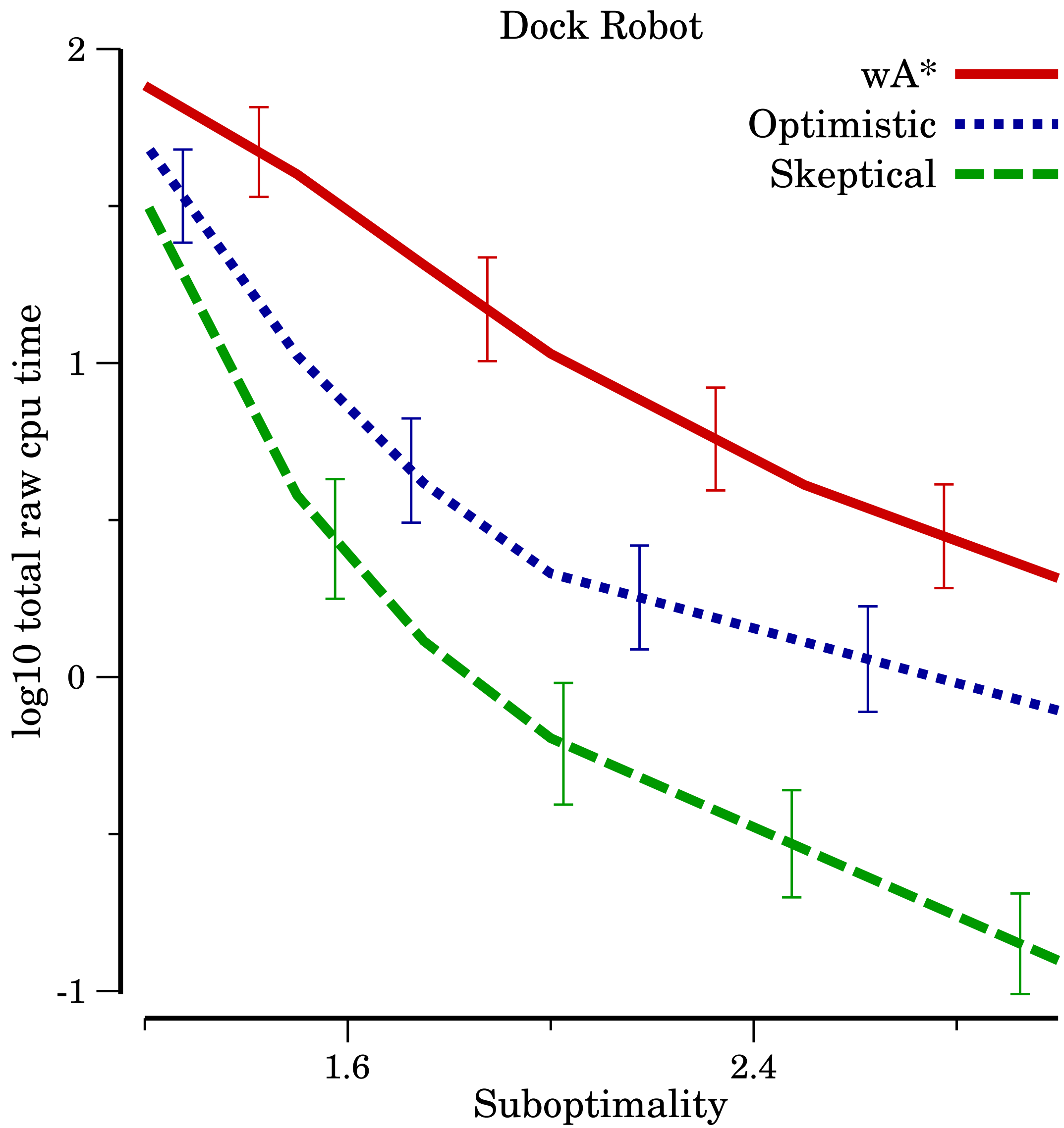
Summary

- BEES outperforms previous state-of-the-art
- when action costs differ, take advantage of d
- inadmissible heuristics can speed up search
- finding solutions, making proofs are different

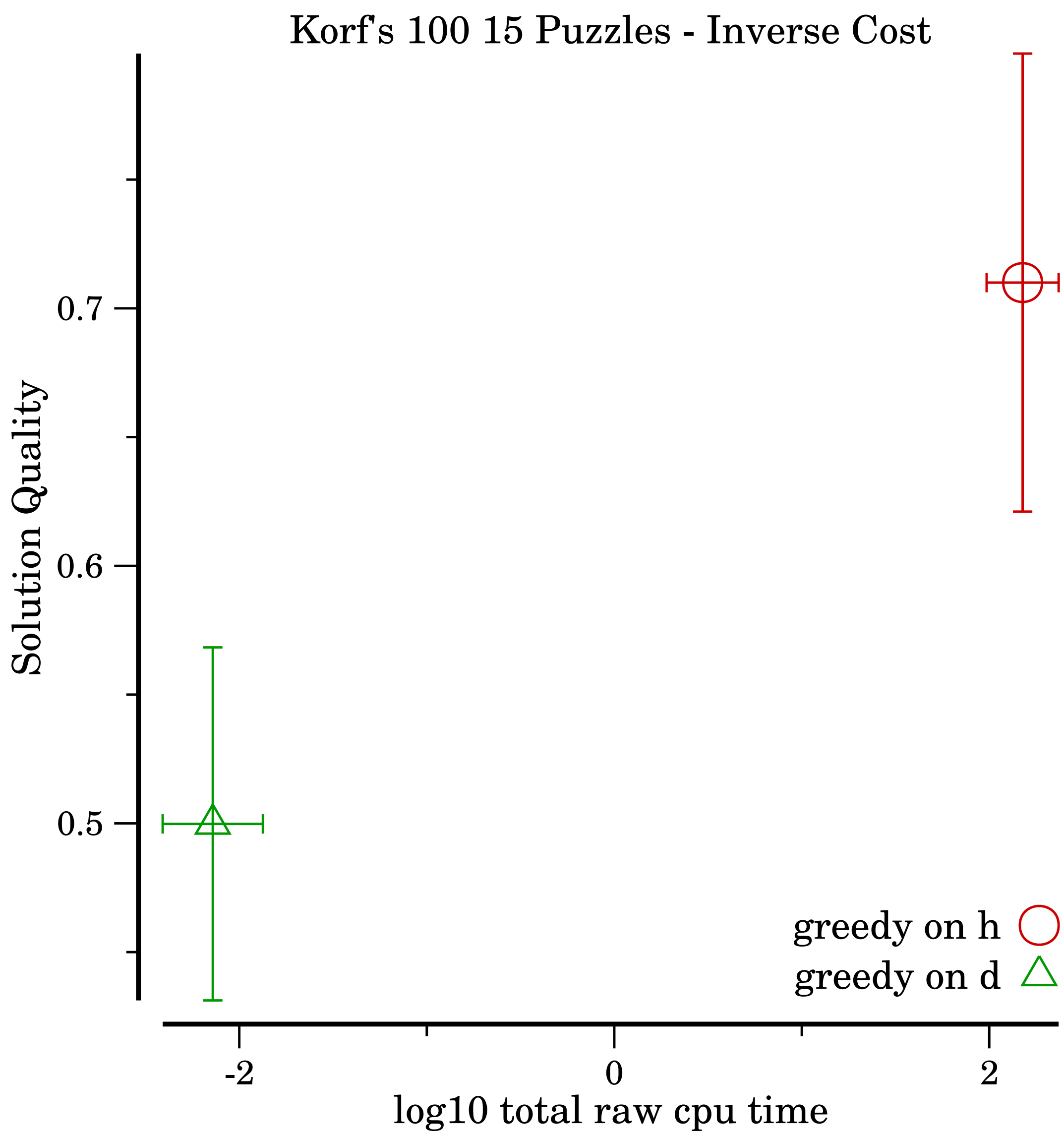
Finding Solutions, Proving Bounds Different Tasks



Inadmissible Heuristics Speed Search



Use d to Go Fast



Summary

- BEES outperforms previous state-of-the-art
- when action costs differ, take advantage of d
- inadmissible heuristics can speed up search
- finding solutions, making proofs are different

Using $\widehat{h}$ in PTS

$$f_{lnr}(n) = \frac{h(n)}{C-g(n)}$$

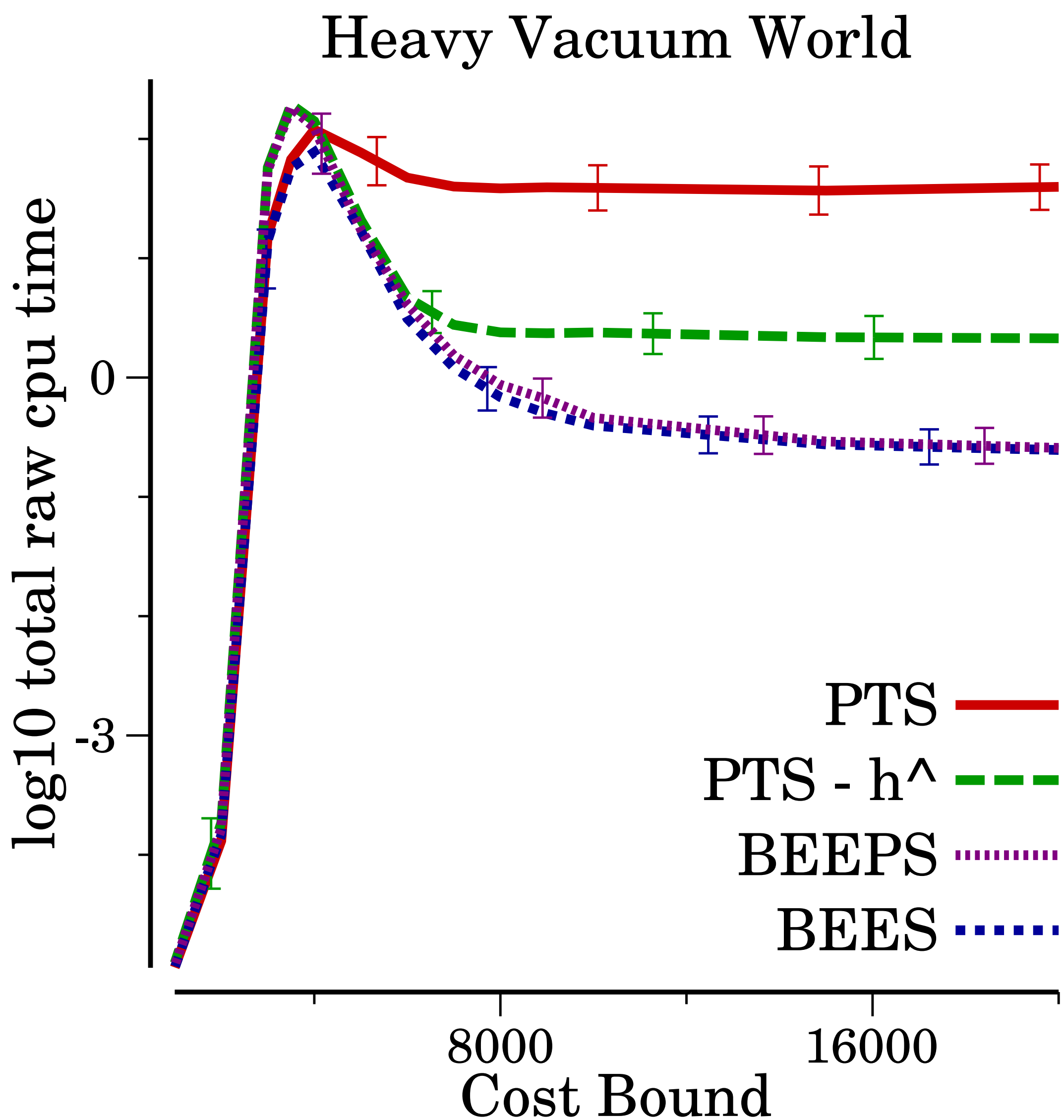
$$f_{lnr}(n) = \frac{\widehat{h}(n)}{C-g(n)}$$

BEEPS - BEES with Potential Measurements

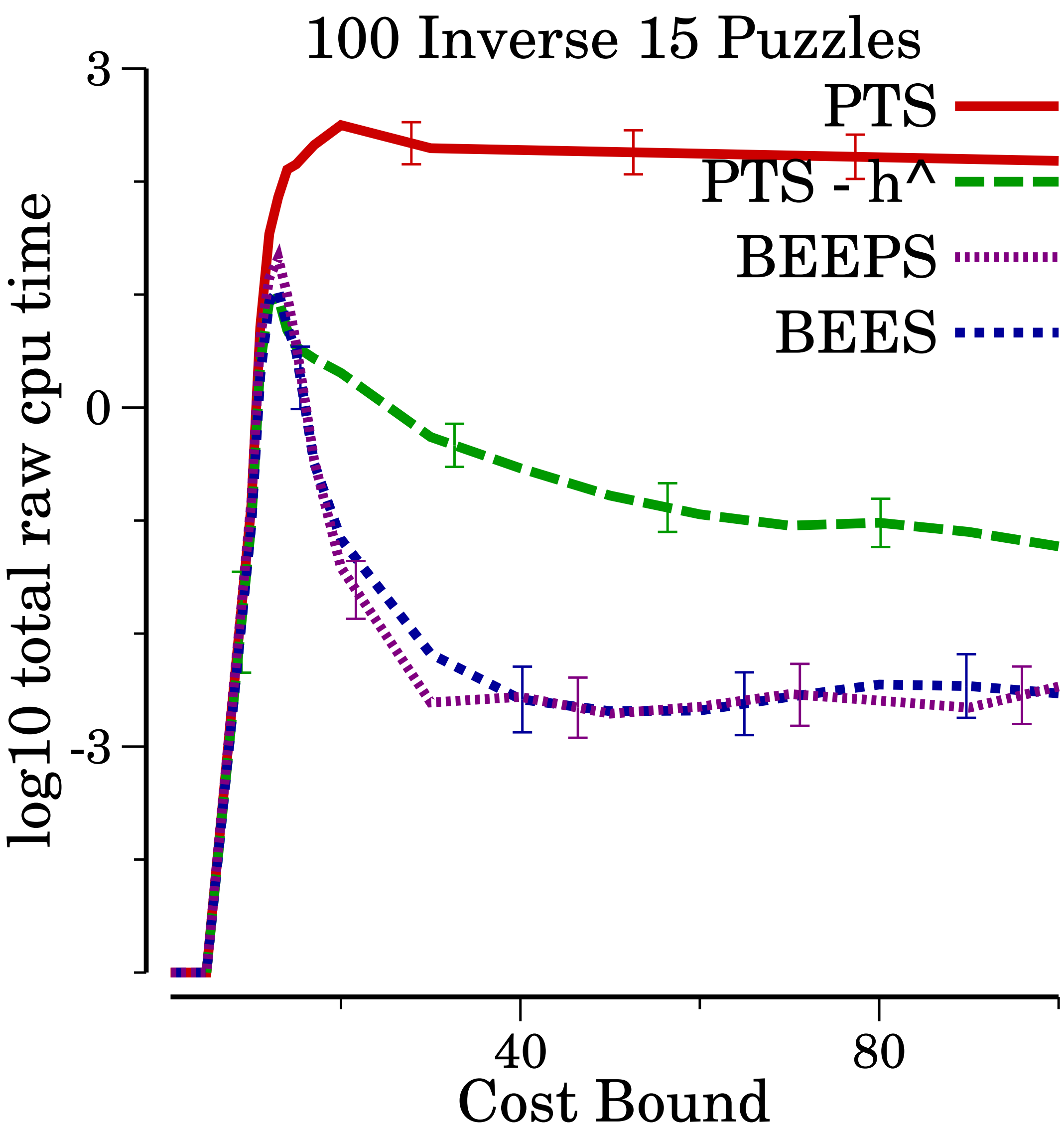
BEES is a best first search on the following rule

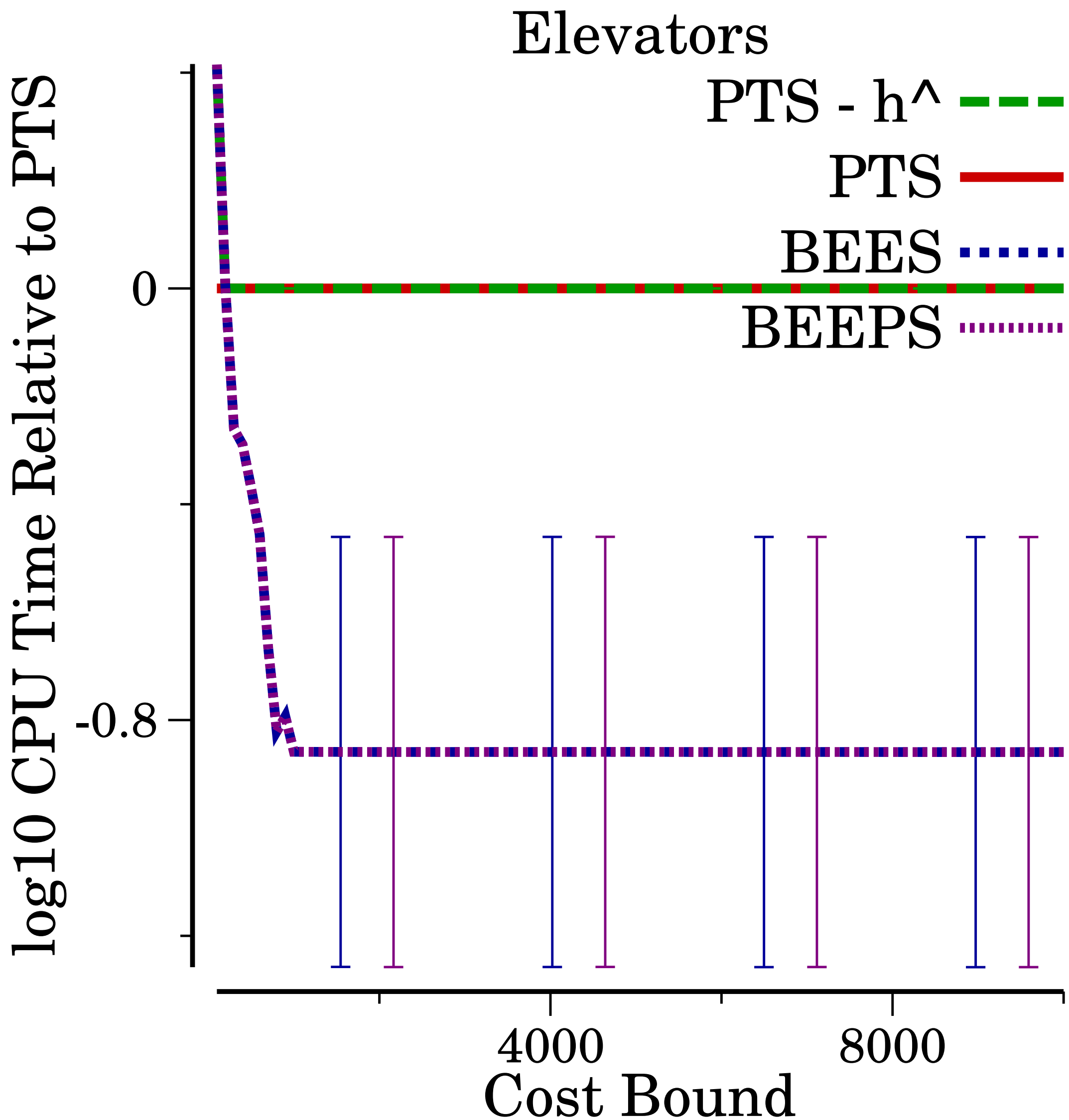
selectNode
1. **if** focal $\neq \{\}$
2. **then** return $n \in$ focal with minimum $\widehat{d}$
3. **else** return $n \in$ open with minimum f_{lnr}

Empirical Results



Empirical Results




Elevators
PTS - h^
PTS
BEES
BEEPS
0
-0.8
log10 CPU Time Relative to PTS
4000
8000
Cost Bound

EES Doesn't Work as Bounded Cost Algorithm