Recursive Best-First Search with Bounded Overhead

Matthew Hatem and Scott Kiesel and Wheeler Ruml

University of New Hampshire

with support from NSF grant IIS-1150068

Problem Summary

- A* remembers every state it visits, often exceeds memory

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead
    - Only best-first in some cases!

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead
    - Only best-first in some cases!
  - Recursive Best-First Search (Korf 1993)

Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead
    - Only best-first in some cases!
  - Recursive Best-First Search (Korf 1993)
    - Always best-first!

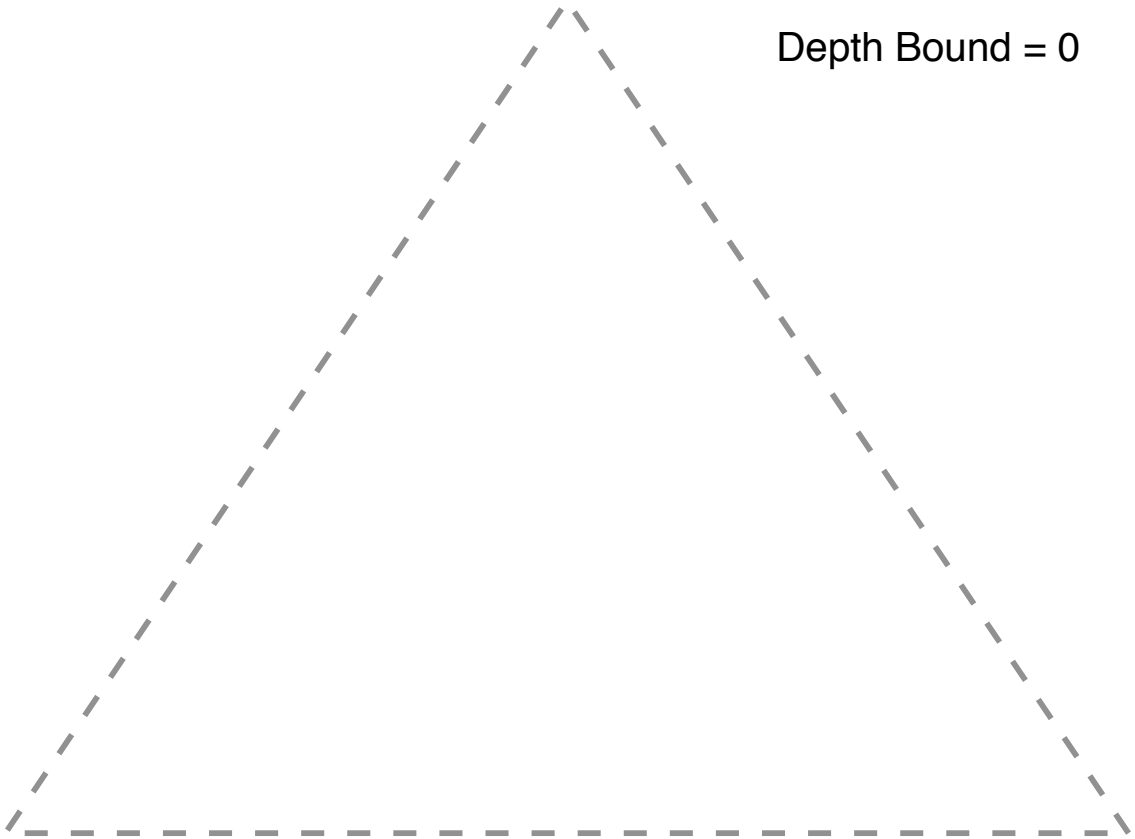
Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead
    - Only best-first in some cases!
  - Recursive Best-First Search (Korf 1993)
    - Always best-first!
    - Suffers from thrashing overhead

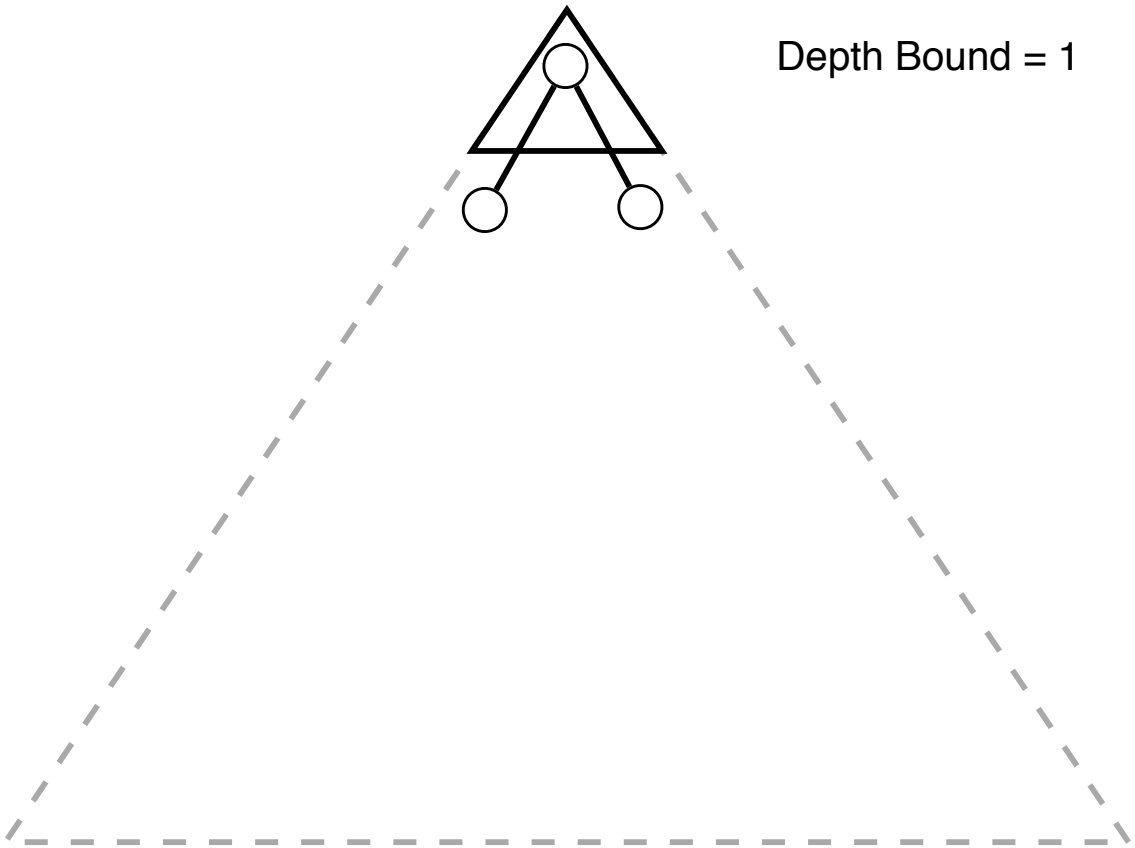
Problem Summary

- A* remembers every state it visits, often exceeds memory
- Motivation for linear-space variants:
  - Iterative Deepening A* (Korf 1985)
    - Has bounded overhead
    - Only best-first in some cases!
  - Recursive Best-First Search (Korf 1993)
    - Always best-first!
    - Suffers from thrashing overhead
      - This is what we fix!

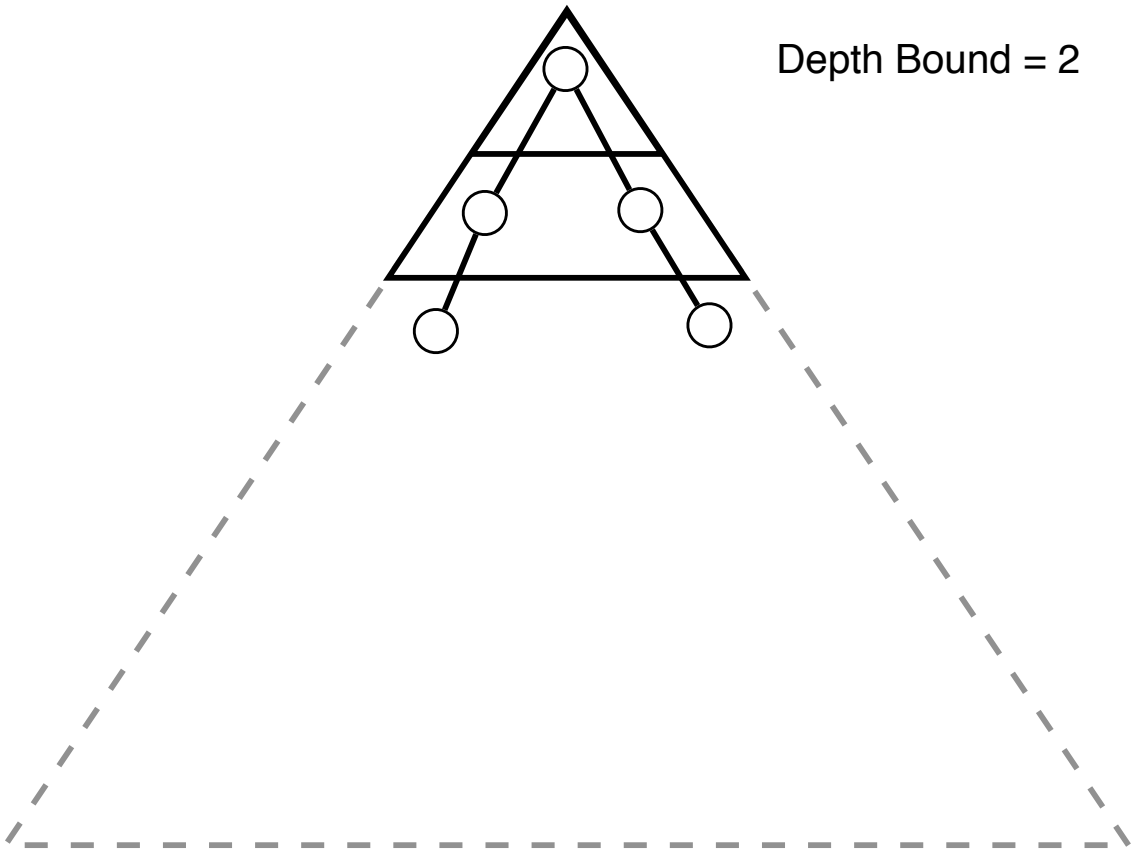
Iterative Deepening Depth-First Search (IDDFS)



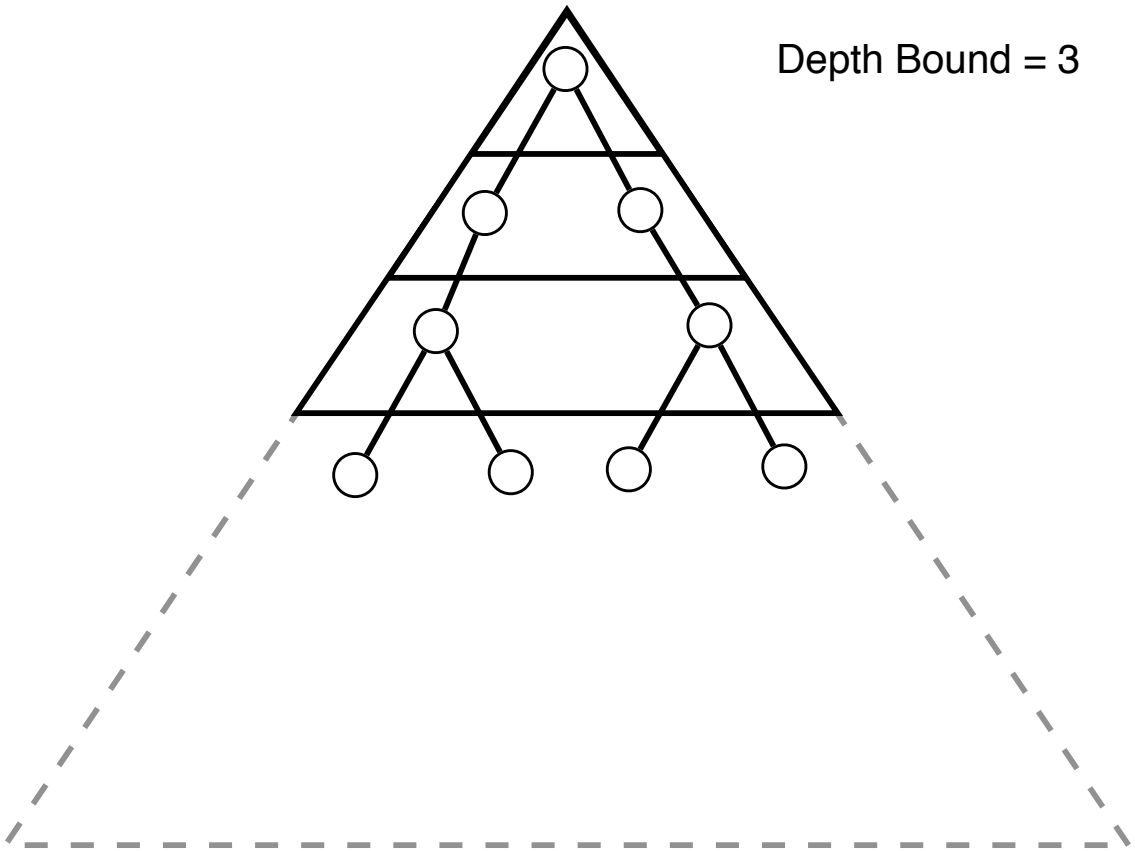
Iterative Deepening Depth-First Search (IDDFS)



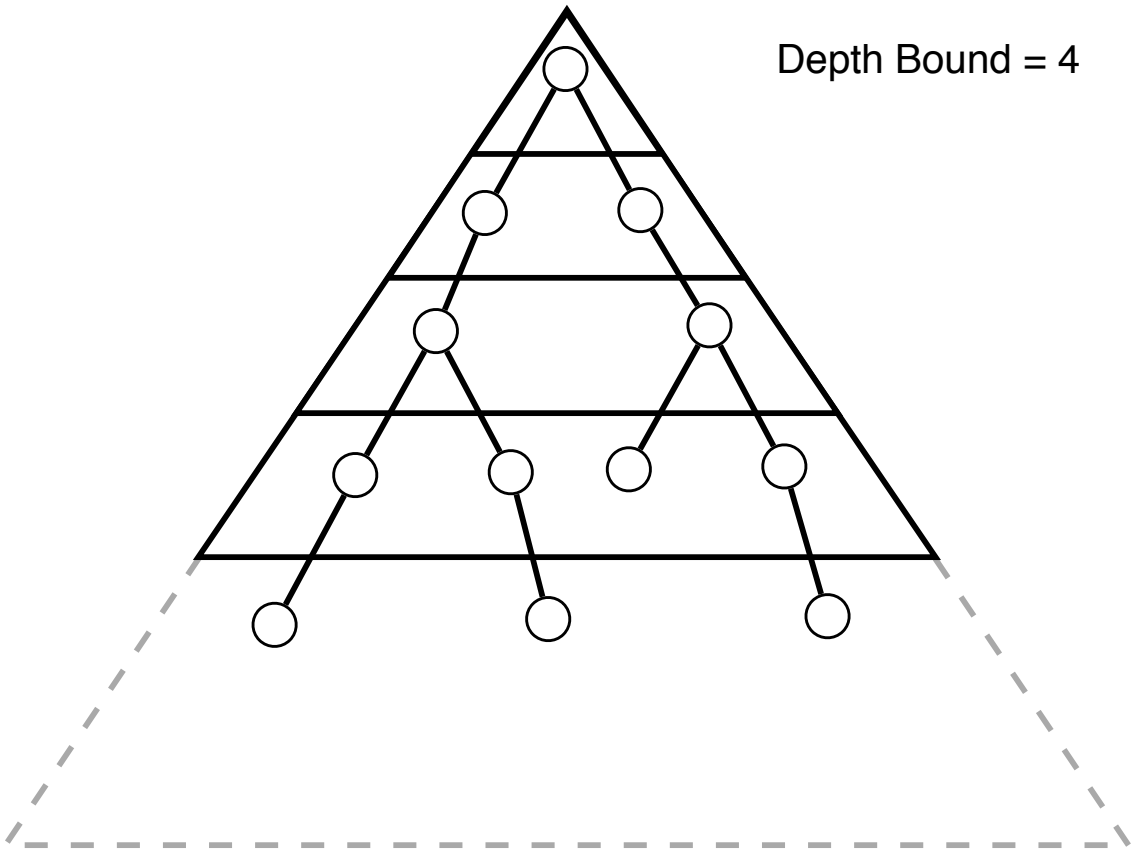
Iterative Deepening Depth-First Search (IDDFS)



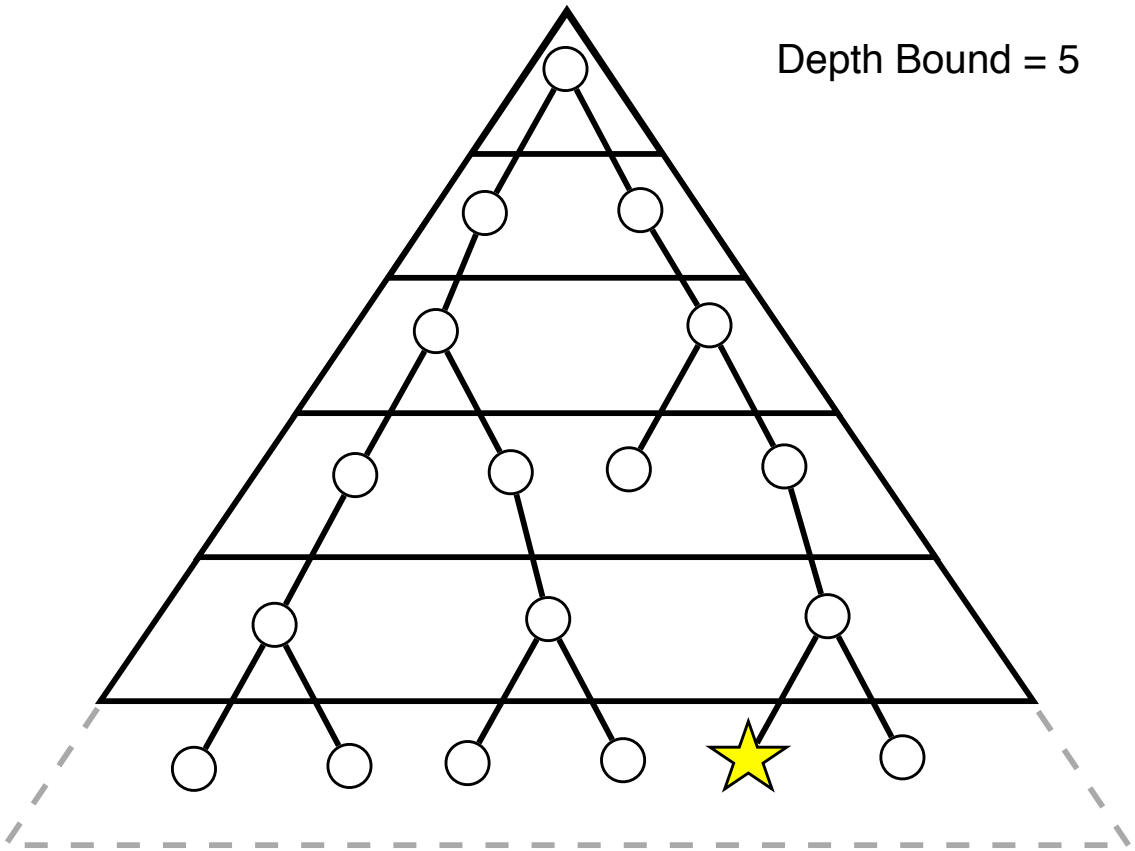
Iterative Deepening Depth-First Search (IDDFS)



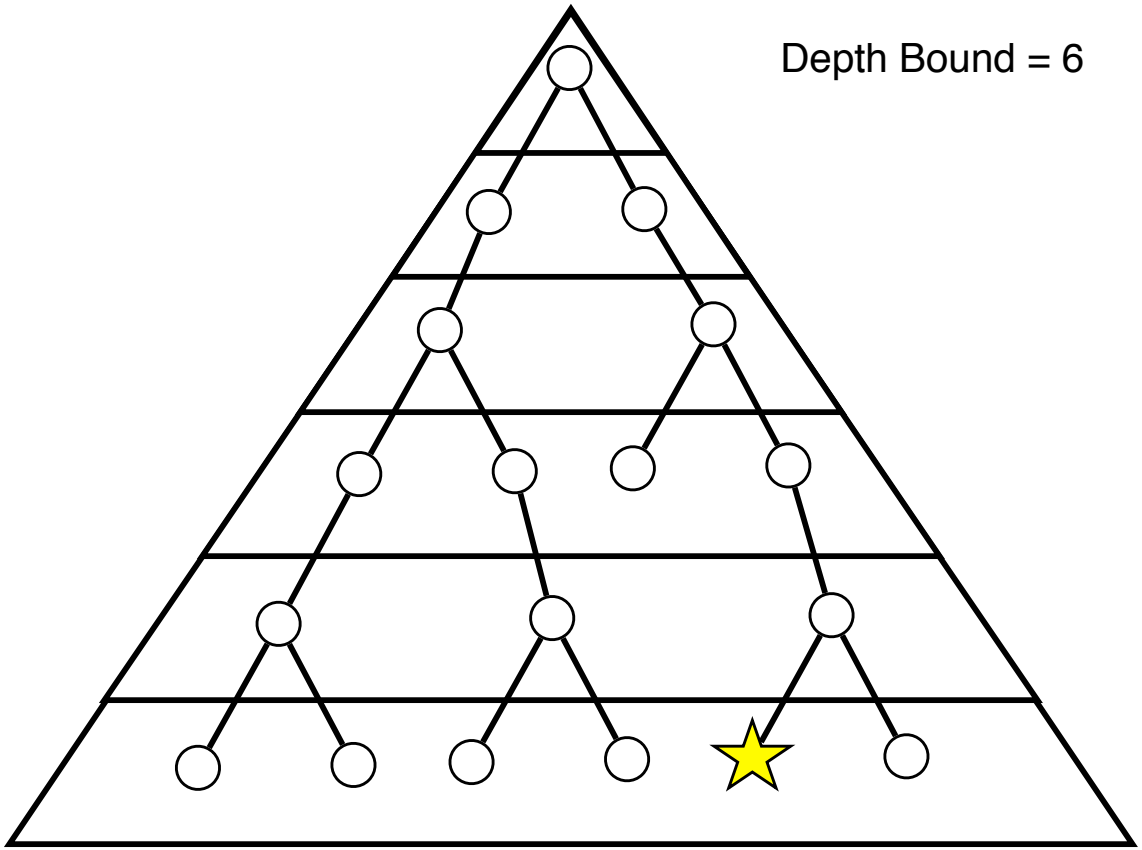
Iterative Deepening Depth-First Search (IDDFS)



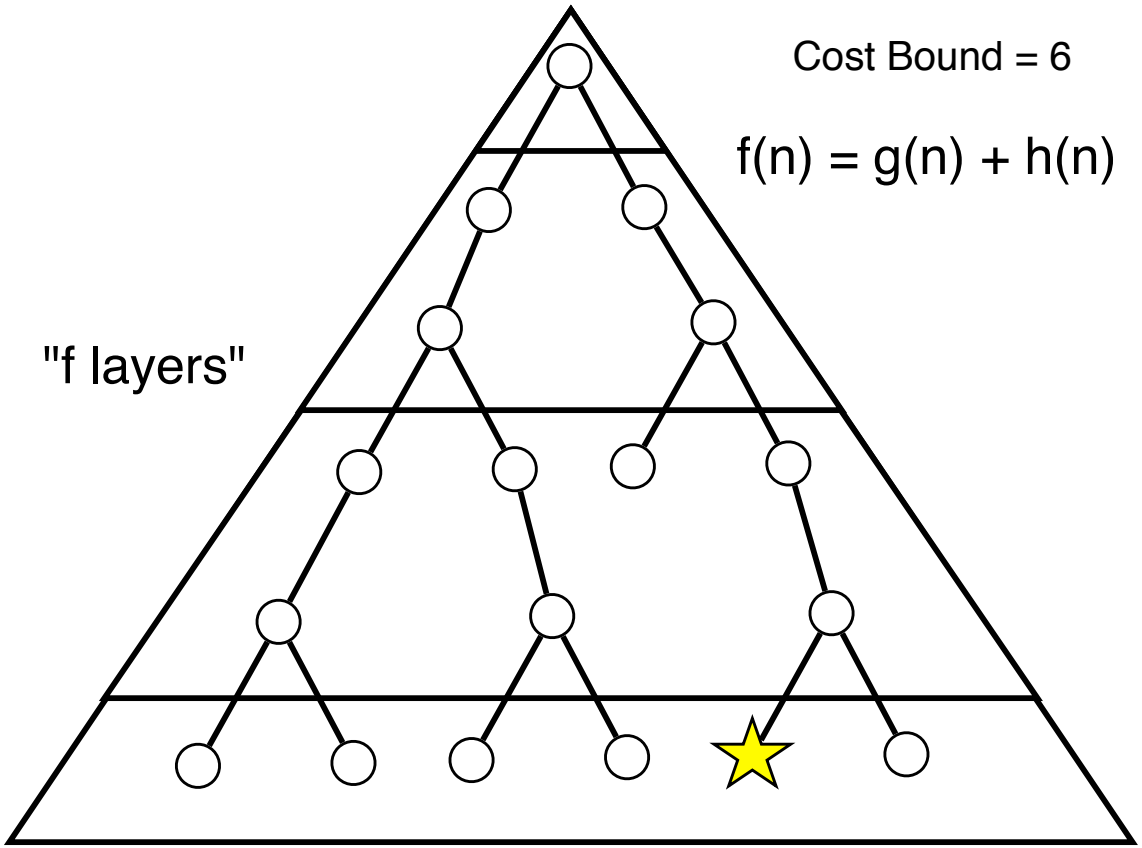
Iterative Deepening Depth-First Search (IDDFS)



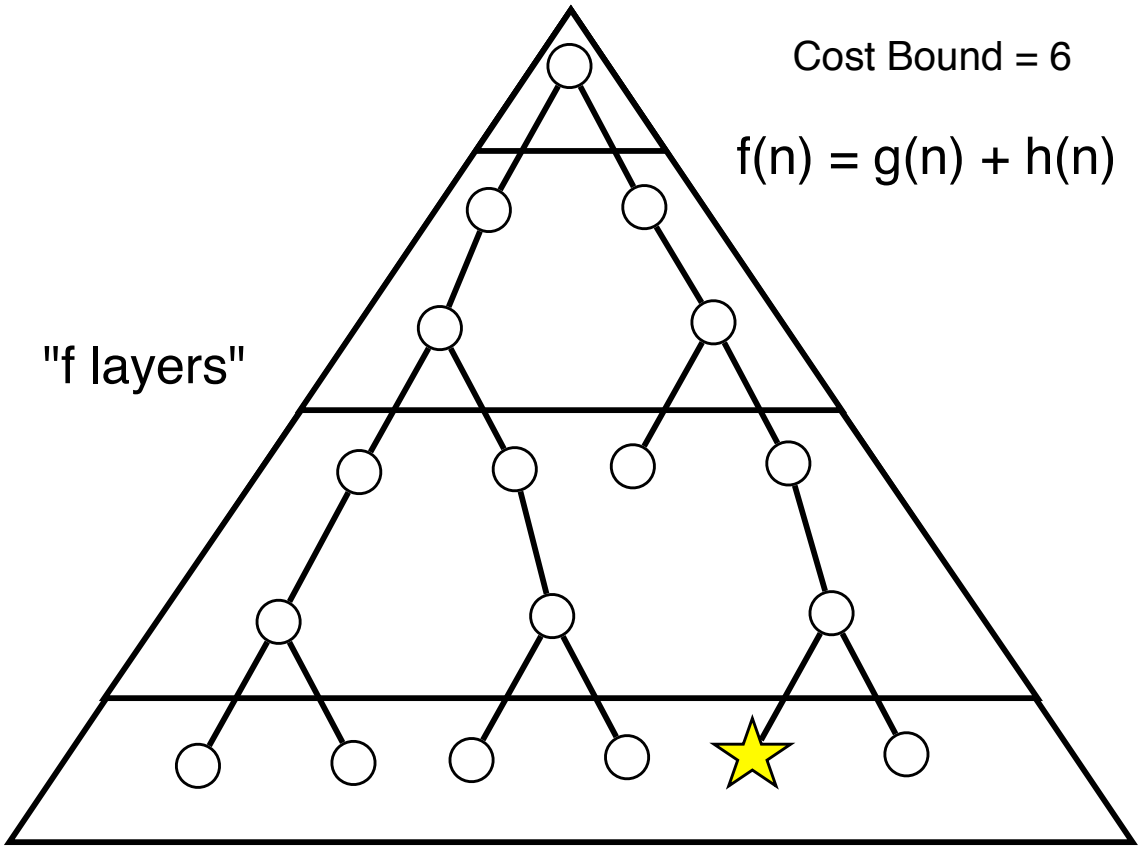
Iterative Deepening Depth-First Search (IDDFS)



Iterative Deepening A* (IDA*)



Iterative Deepening A* (IDA*)

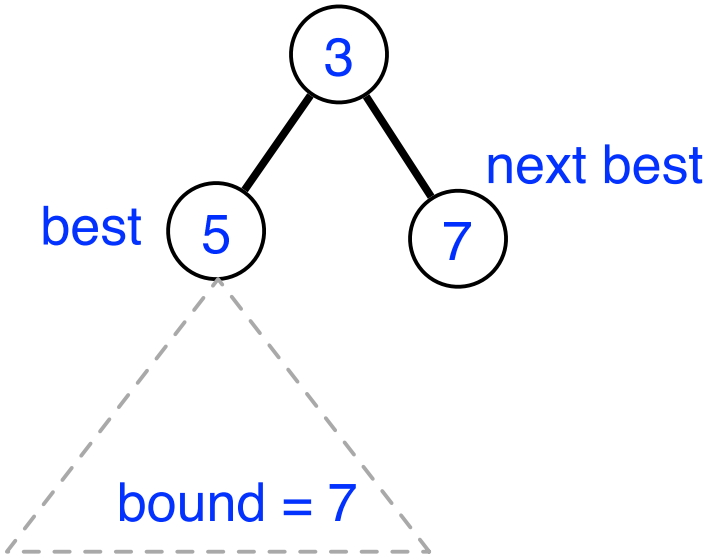


Only best-first if f layers are monotonically increasing!

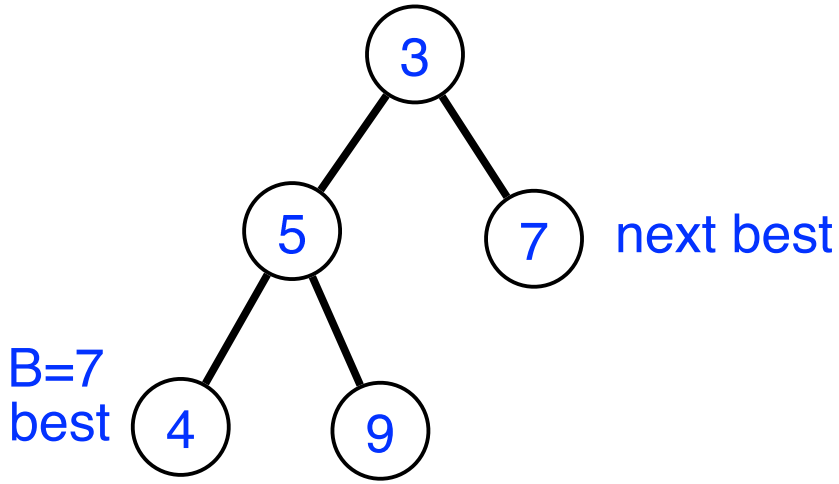
(not the case in, eg, suboptimal variants)

Recursive Best-First Search (RBFS)

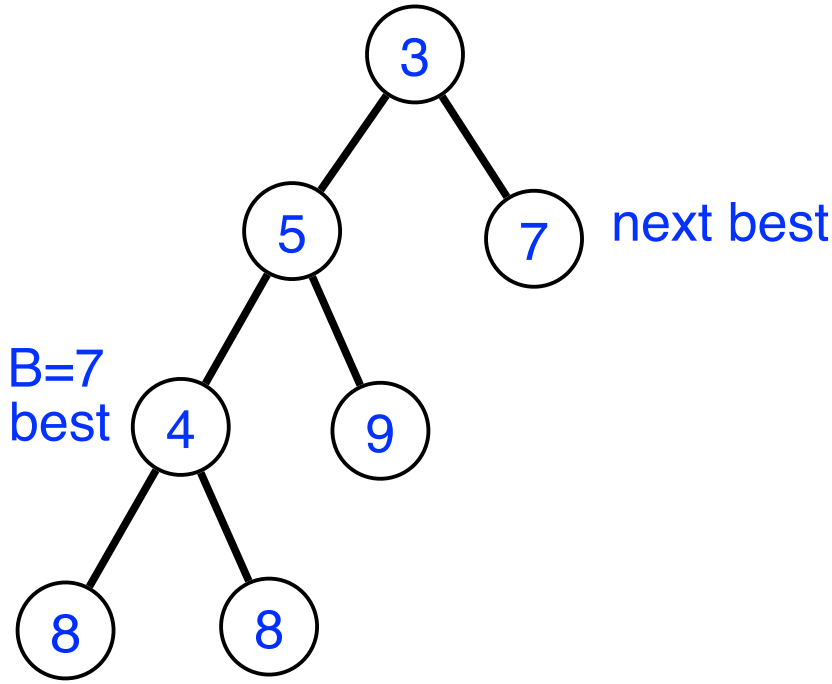
Recursive Best-First Search (RBFS)



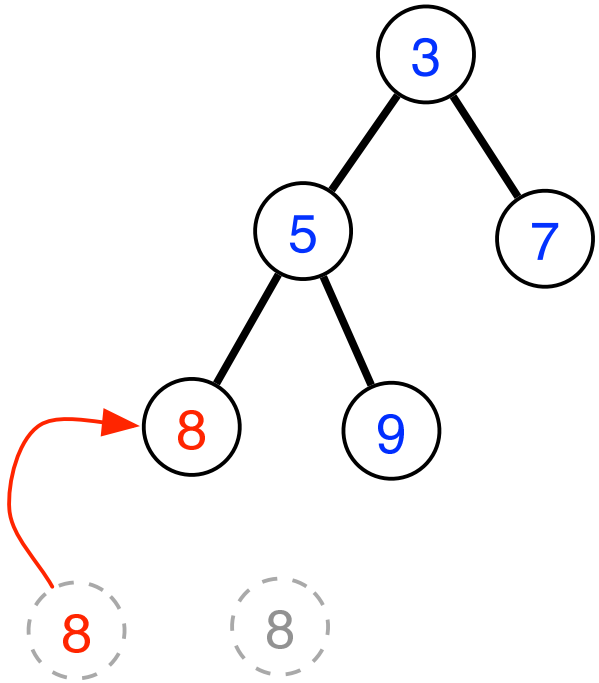
Recursive Best-First Search (RBFS)



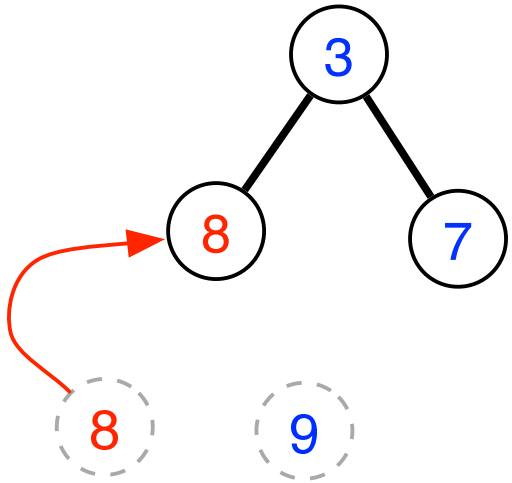
Recursive Best-First Search (RBFS)



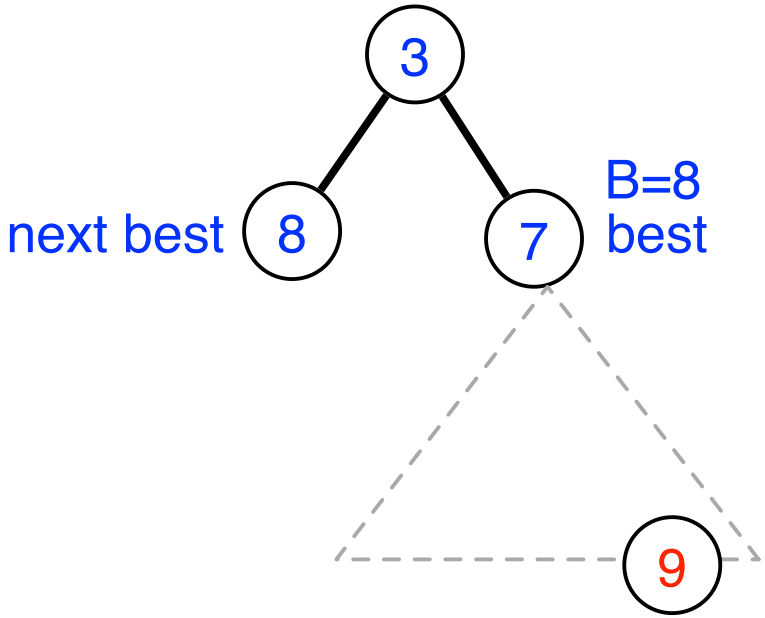
Recursive Best-First Search (RBFS)



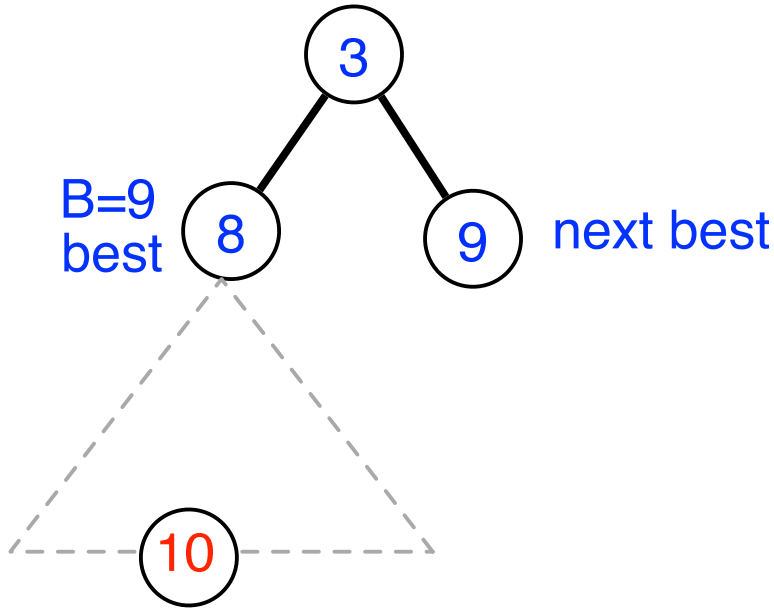
Recursive Best-First Search (RBFS)



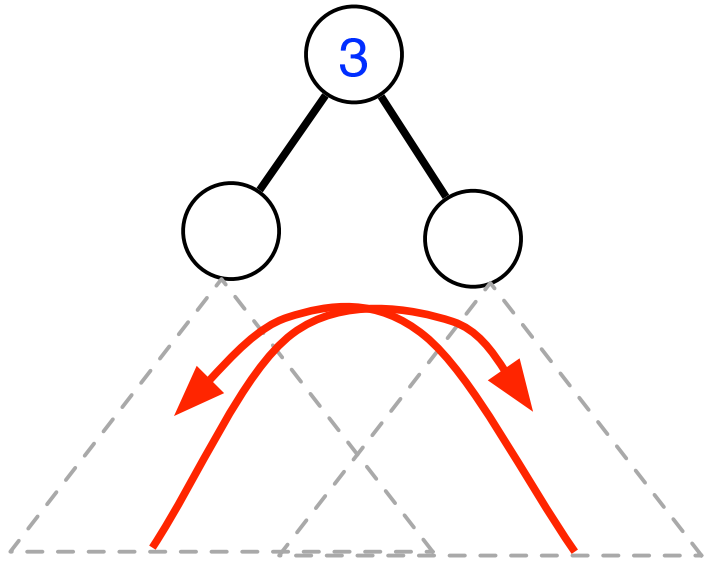
Recursive Best-First Search (RBFS)



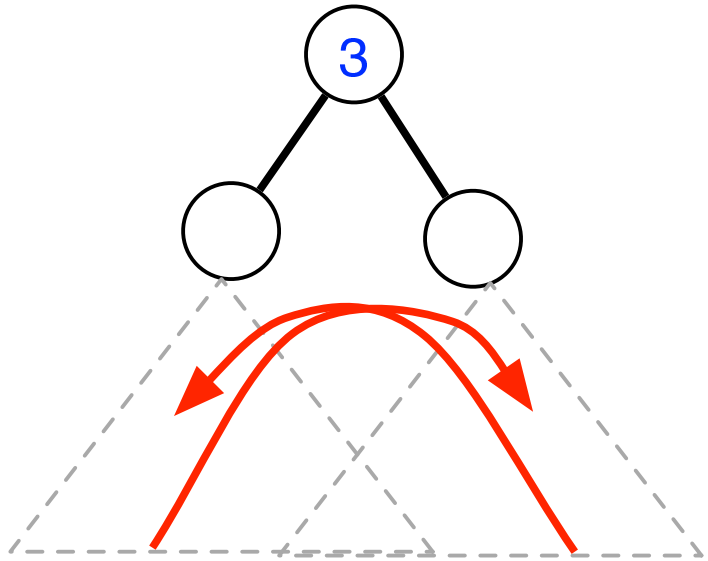
Recursive Best-First Search (RBFS)



Recursive Best-First Search (RBFS)

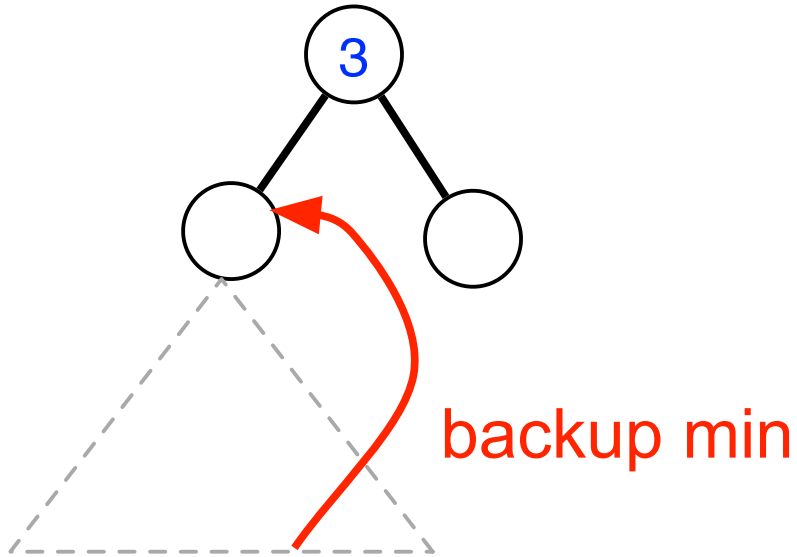


Recursive Best-First Search (RBFS)

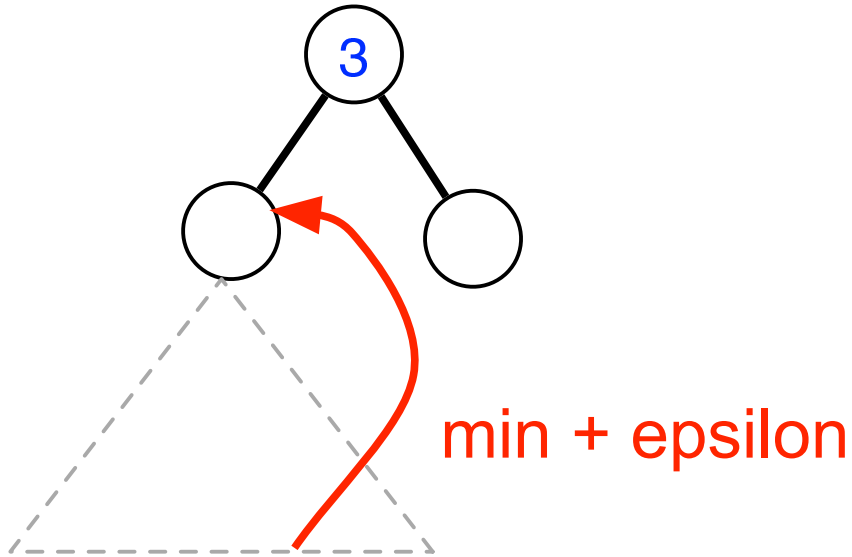


Strict best-first search order causes thrashing!

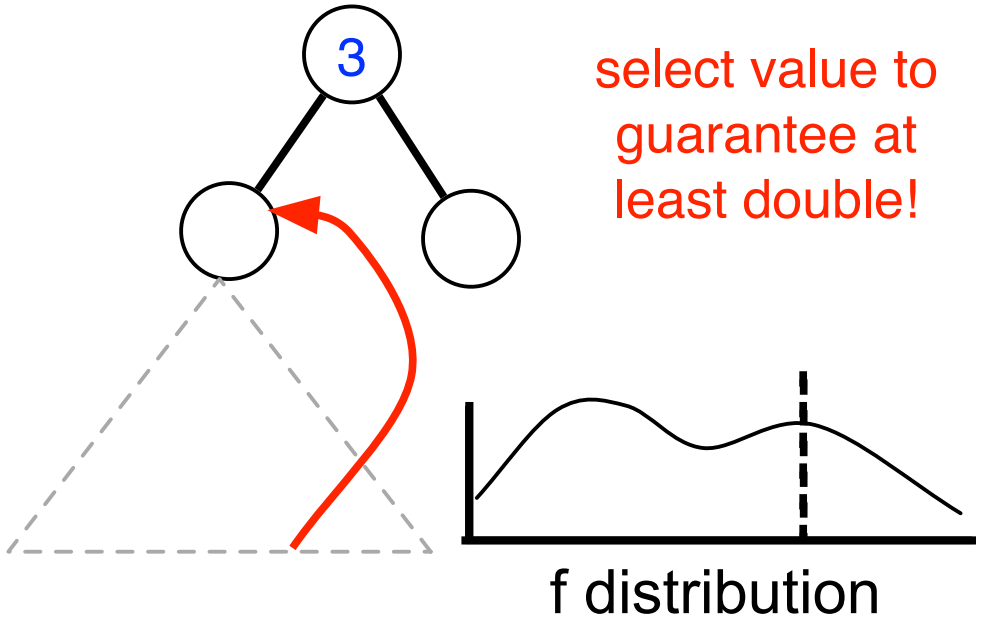
RBFS with Bounded Overhead



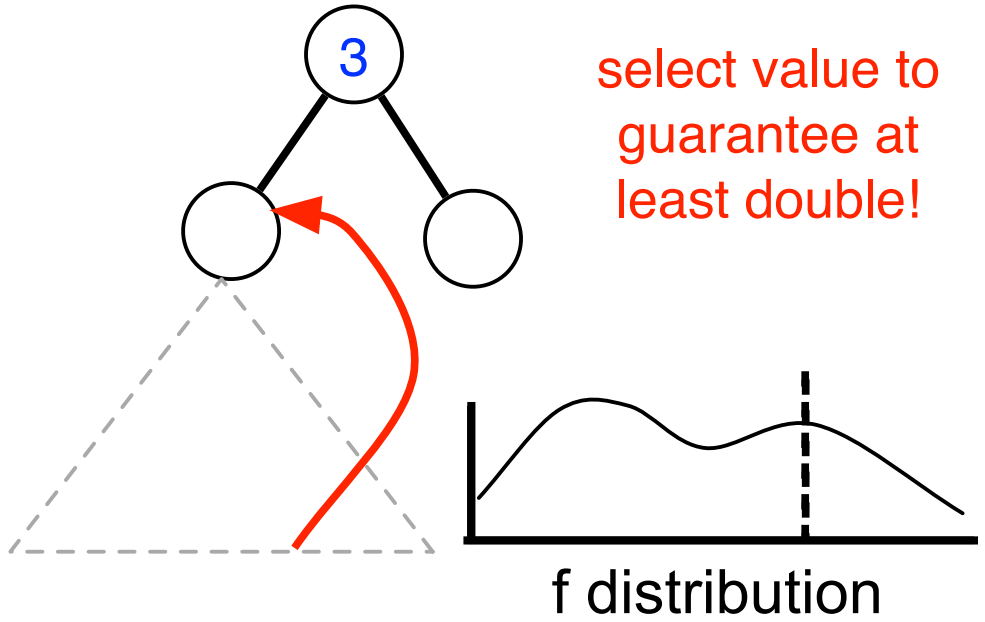
RBFS with Bounded Overhead



RBFS with Bounded Overhead



RBFS with Bounded Overhead

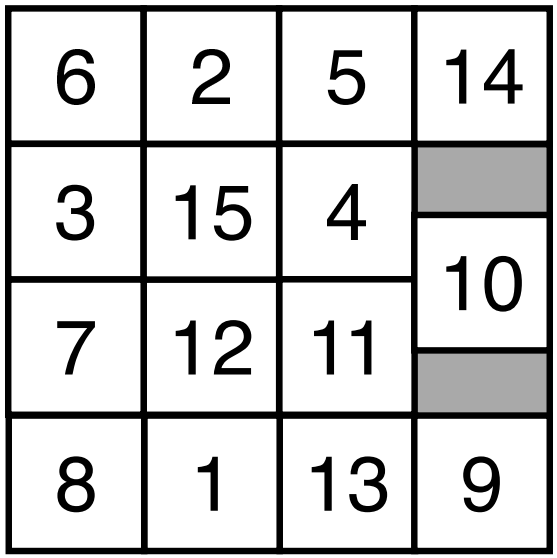


Relaxing best-first search order reduces overhead!

See paper for proof

Sliding Tile Puzzle

- Korf 100 (Korf 1985)
- A* with Manhattan Distance runs out of memory



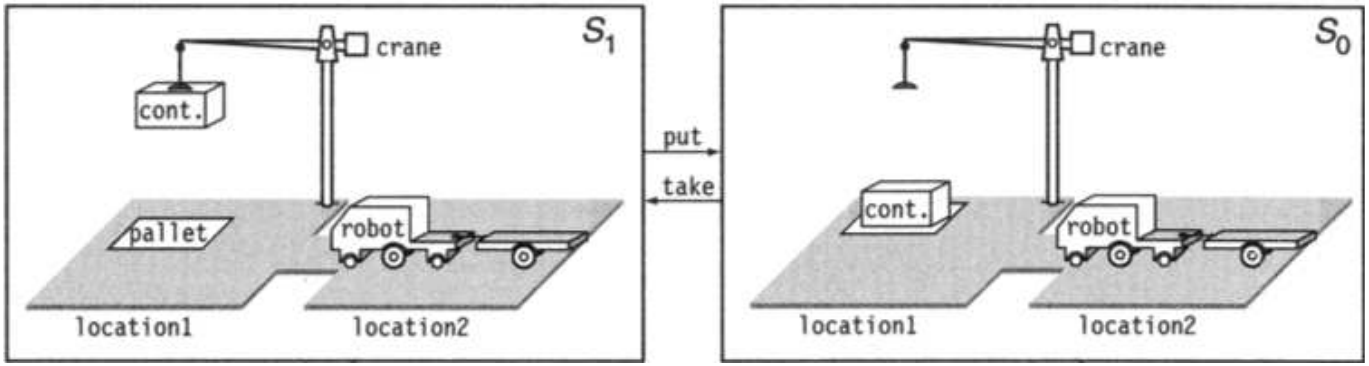
Sliding Tile Puzzle

Korfs 100 15 puzzles (unit cost)

(w=2)	Exp.	Avg. Cost	Time
RBFS	29,253,944	**62**	44
$RBFS_{\epsilon}$	13,078,227	71	30
$RBFS_{CR}$	11,695,743	66	56
IDA*	**11,136,196**	68	**11**

Relaxing best-first search order gives faster solving times
Better solutions but slower than IDA*

Dockyard Robot Planning



- From Ghallab, Nau, Traverso (2004)
- All actions have real costs
- Many duplicate states
- IDA*$_{CR}$ and RBFS$_{CR}$ with transposition tables

Dockyard Robot Planning

5 locations, cranes, piles and 8 containers

	Exp.	Exp./Sec.	Time	Reopened
IDA*$_{CR}$	2,044m	112k	18,394	2,042m
RBFS$_{CR}$	188m	103k	1,824	87m
RBFS$_{\epsilon=1}$	472m	**147k**	3,222	**4m**
RBFS$_{\epsilon=2}$	251m	140k	1,795	5m
RBFS$_{\epsilon=3}$	**154m**	141k	**1,094**	14m

Dockyard Robot Planning

5 locations, cranes, piles and 8 containers

	Exp.	Exp./Sec.	Time	Reopened
IDA*$_{CR}$	2,044m	112k	18,394	2,042m
RBFS$_{CR}$	188m	103k	1,824	87m
RBFS$_{\epsilon=1}$	472m	**147k**	3,222	**4m**
RBFS$_{\epsilon=2}$	251m	140k	1,795	5m
RBFS$_{\epsilon=3}$	**154m**	141k	**1,094**	14m

RBFS$_{CR}$ is faster because it expands 10x fewer nodes

Summary

- IDA* is depth-first, RBFS is always best-first!

Summary

- IDA* is depth-first, RBFS is always best-first!
- Simple modifications reduce overhead (see paper)

Summary

- IDA* is depth-first, RBFS is always best-first!
- Simple modifications reduce overhead (see paper)
- RBFS_{CR} gives first provable bounds for overhead

Summary

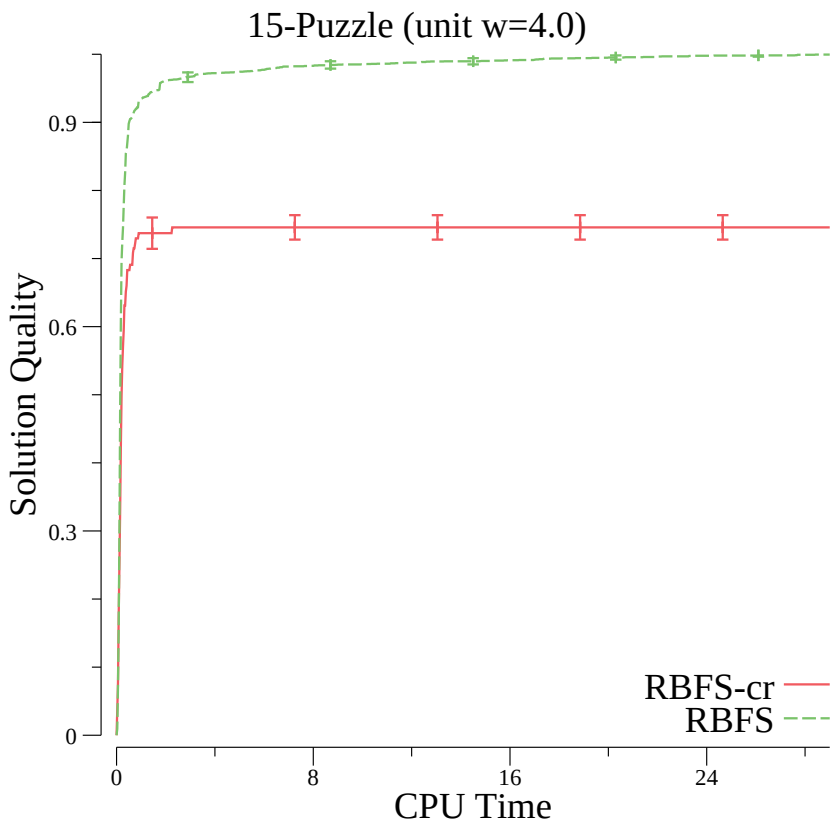
- IDA* is depth-first, RBFS is always best-first!
- Simple modifications reduce overhead (see paper)
- RBFS_{CR} gives first provable bounds for overhead
- Is it time for IDA* to retire?

Summary

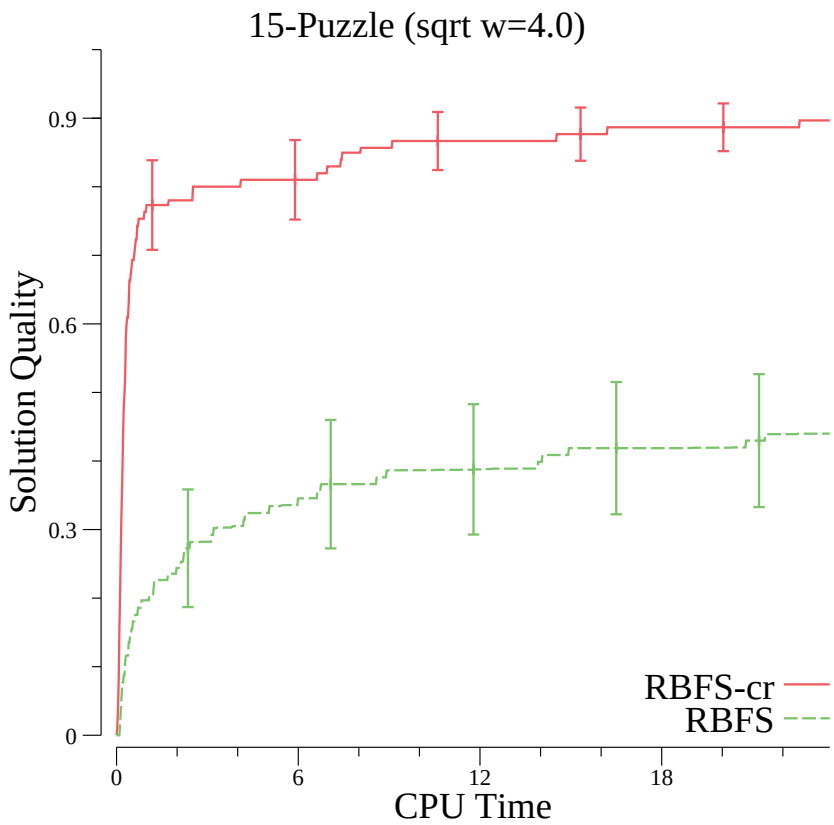
- IDA* is depth-first, RBFS is always best-first!
- Simple modifications reduce overhead (see paper)
- RBFS_{CR} gives first provable bounds for overhead
- Is it time for IDA* to retire?
- RBFS deserves more attention!
  - IDA* citations: 1523
  - RBFS citations: 310

Backup

Anytime Heuristic Search



Anytime Heuristic Search



Anytime Heuristic Search