When does Weighted A* Fail?

Christopher Wilt

University of New Hampshire

Weighted Search

- Weighting the heuristic is one of the most important techniques in heuristic search.

Overview

- Weighting the heuristic is one of the most important techniques in heuristic search.
- When A* fails to find a solution in the required time, next we typically try Weighted A*

- Weighted A* expands nodes in $f'(n) = w \cdot h(n) + g(n)$ order.

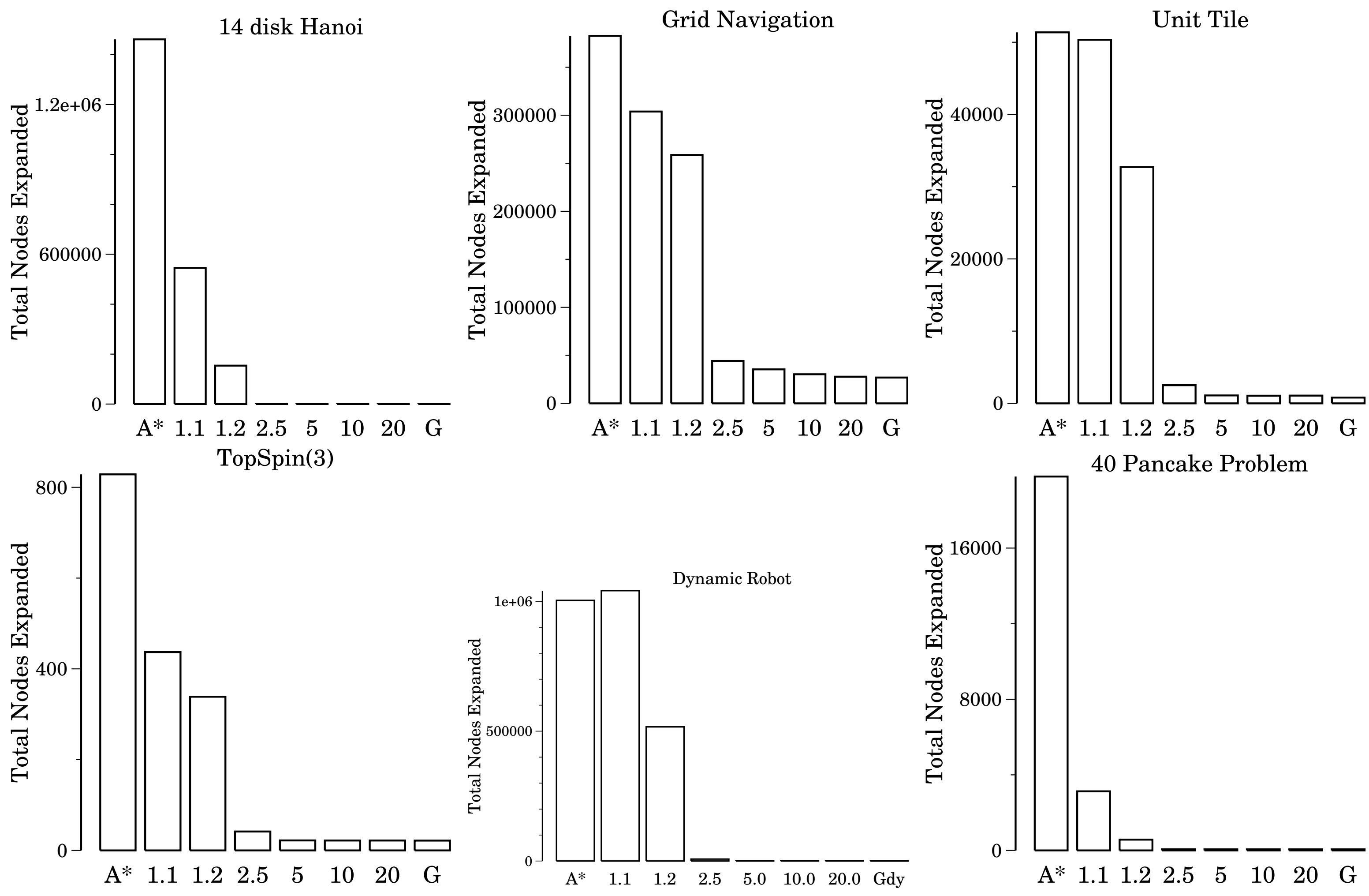
Terminology

- Weighted A* expands nodes in $f'(n) = w \cdot h(n) + g(n)$ order.
- A weight of 1 is equivalent to A*.

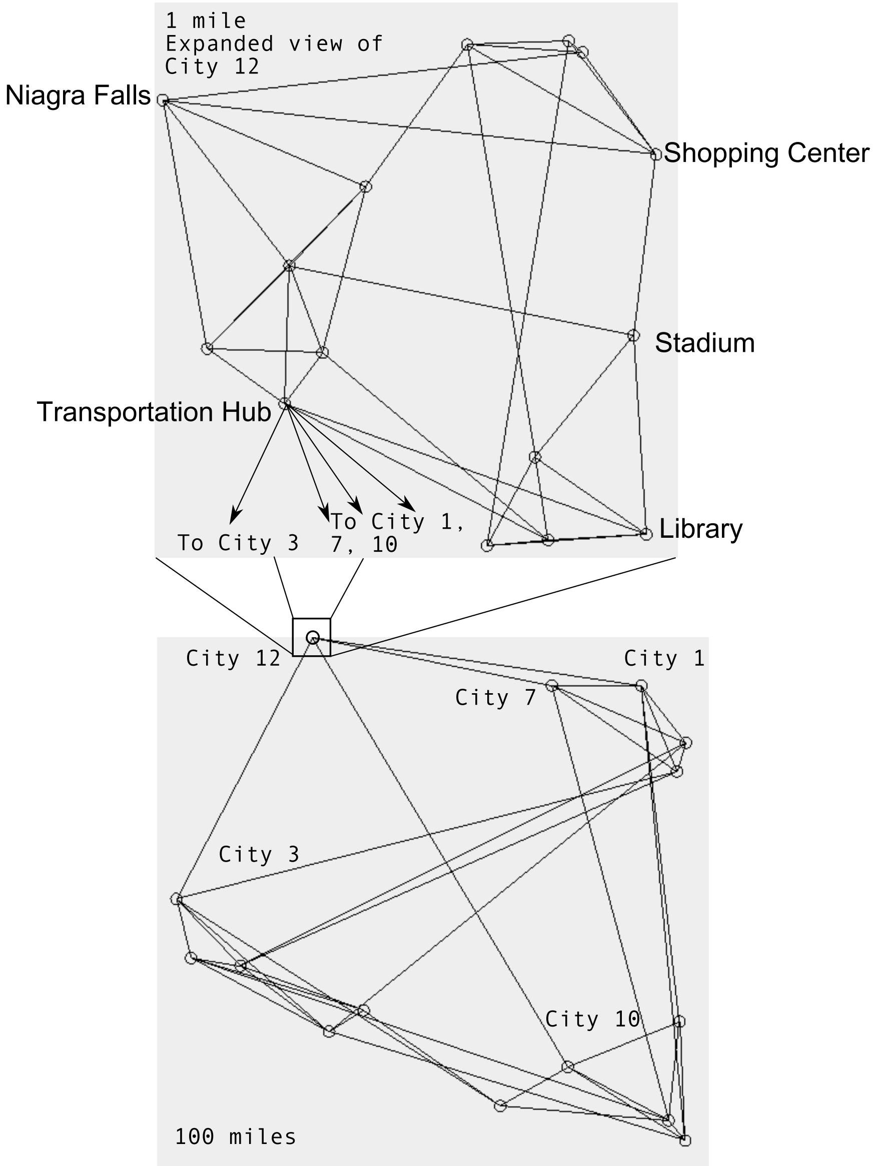
Terminology

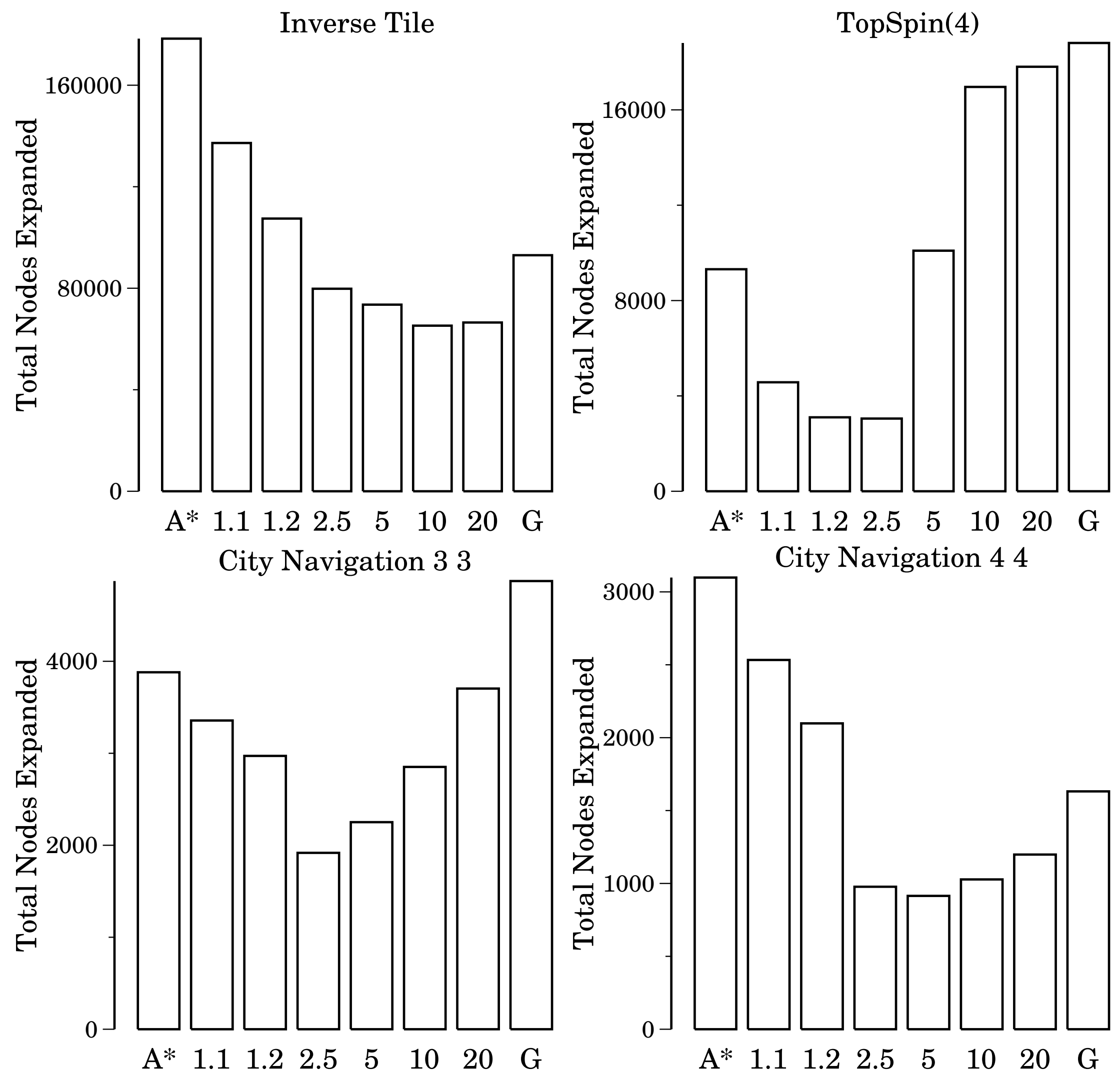
- Weighted A* expands nodes in $f'(n) = w \cdot h(n) + g(n)$ order.
- A weight of 1 is equivalent to A*.
- A weight of ∞ is equivalent to greedy search.

What happens when w increases?



City Navigation




Inverse Tile
Total Nodes Expanded
0
80000
160000
A* 1.1 1.2 2.5 5 10 20 G
TopSpin(4)
Total Nodes Expanded
0
8000
16000
A* 1.1 1.2 2.5 5 10 20 G
City Navigation 3 3
Total Nodes Expanded
0
2000
4000
A* 1.1 1.2 2.5 5 10 20 G
City Navigation 4 4
Total Nodes Expanded
0
1000
2000
3000
A* 1.1 1.2 2.5 5 10 20 G

Conclusion

- We sometimes weight the heuristic in to try and make best-first searches go faster.

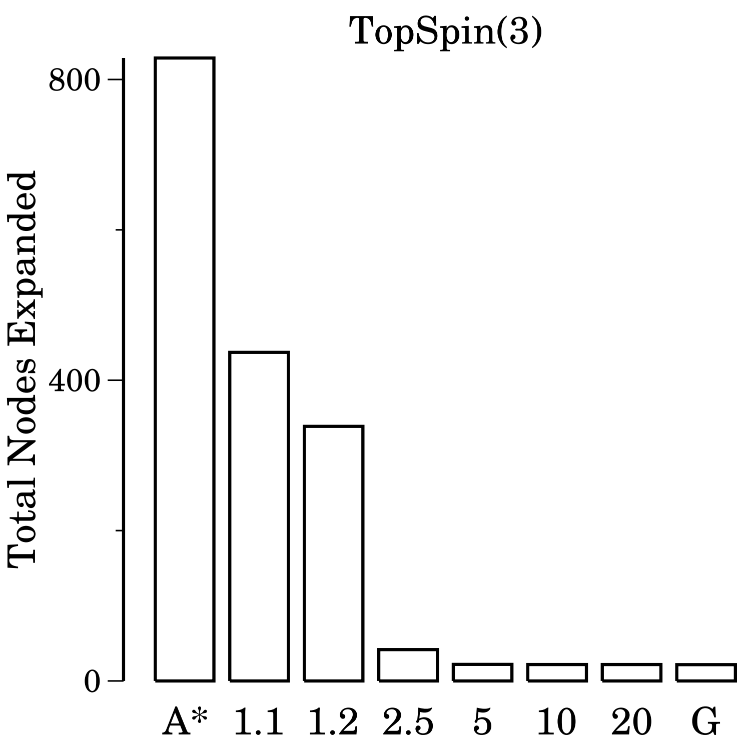
Conclusion

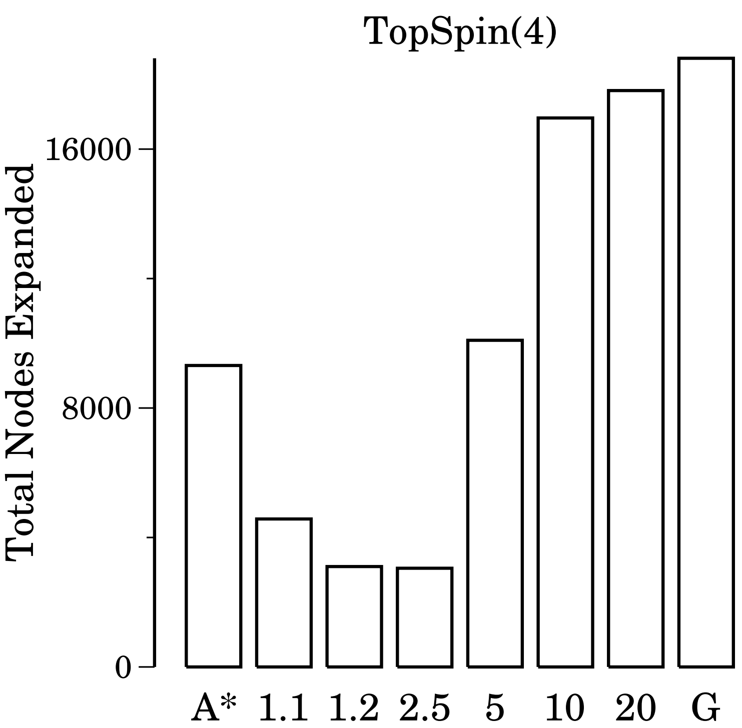
- We sometimes weight the heuristic in to try and make best-first searches go faster.
- This often results in fewer node expansions, but not always.

The Problem

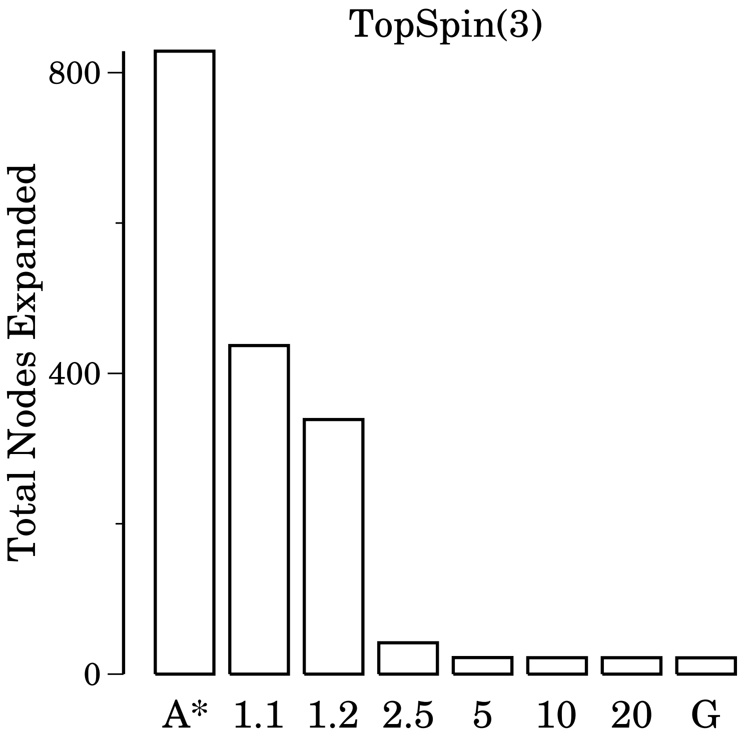
Weighting Fails?

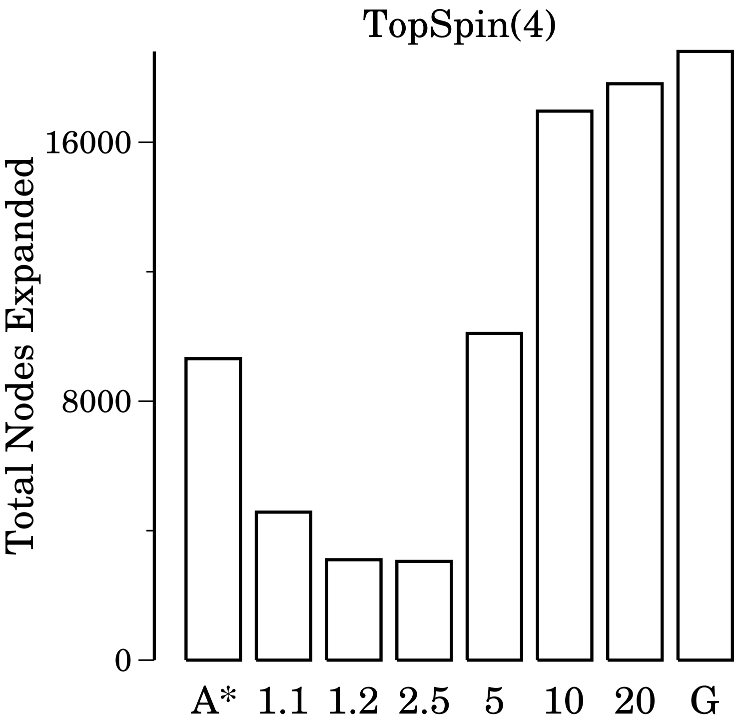
Weighting may or may not help.





How do we differentiate between these?





Hypotheses

Heuristic % Error

- Heuristic % Error is $\frac{h^*(n) - h(n)}{h^*(n)}$

Heuristic % Error

- Heuristic % Error is $\frac{h^*(n)-h(n)}{h^*(n)}$
-

Domain	Error
Citynav 4 4	37.41
Unit Tiles	33.37
Inverse Tiles	29.49
Hanoi	29.47
TopSpin(4)	20.25
Grid	12.78

Heuristic % Error

- Heuristic % Error is $\frac{h^*(n)-h(n)}{h^*(n)}$

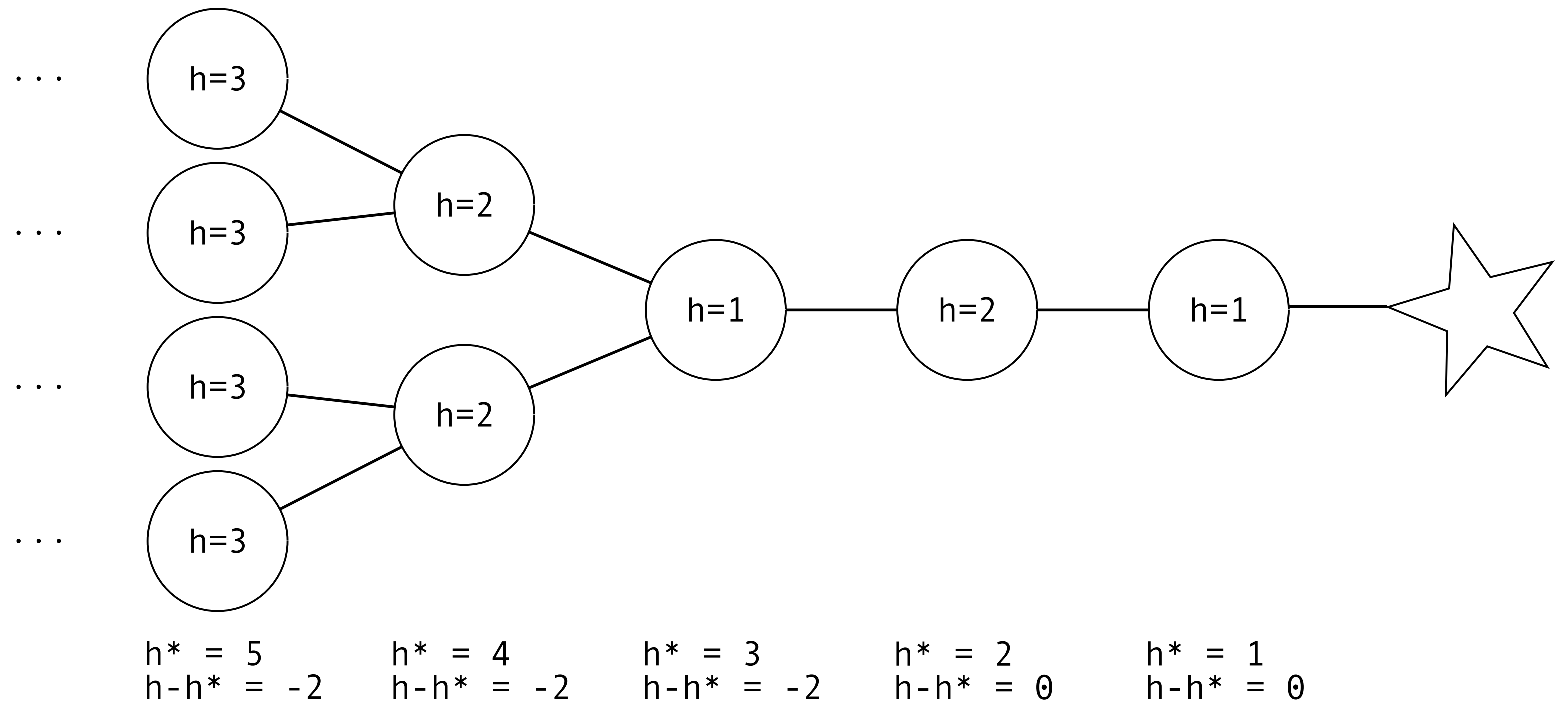
Domain	Error
Citynav 4 4	37.41
Unit Tiles	33.37
Inverse Tiles	29.49
Hanoi	29.47
TopSpin(4)	20.25
Grid	12.78

- Knowing % Heuristic Error does not let us predict whether or not greedy search will fail.

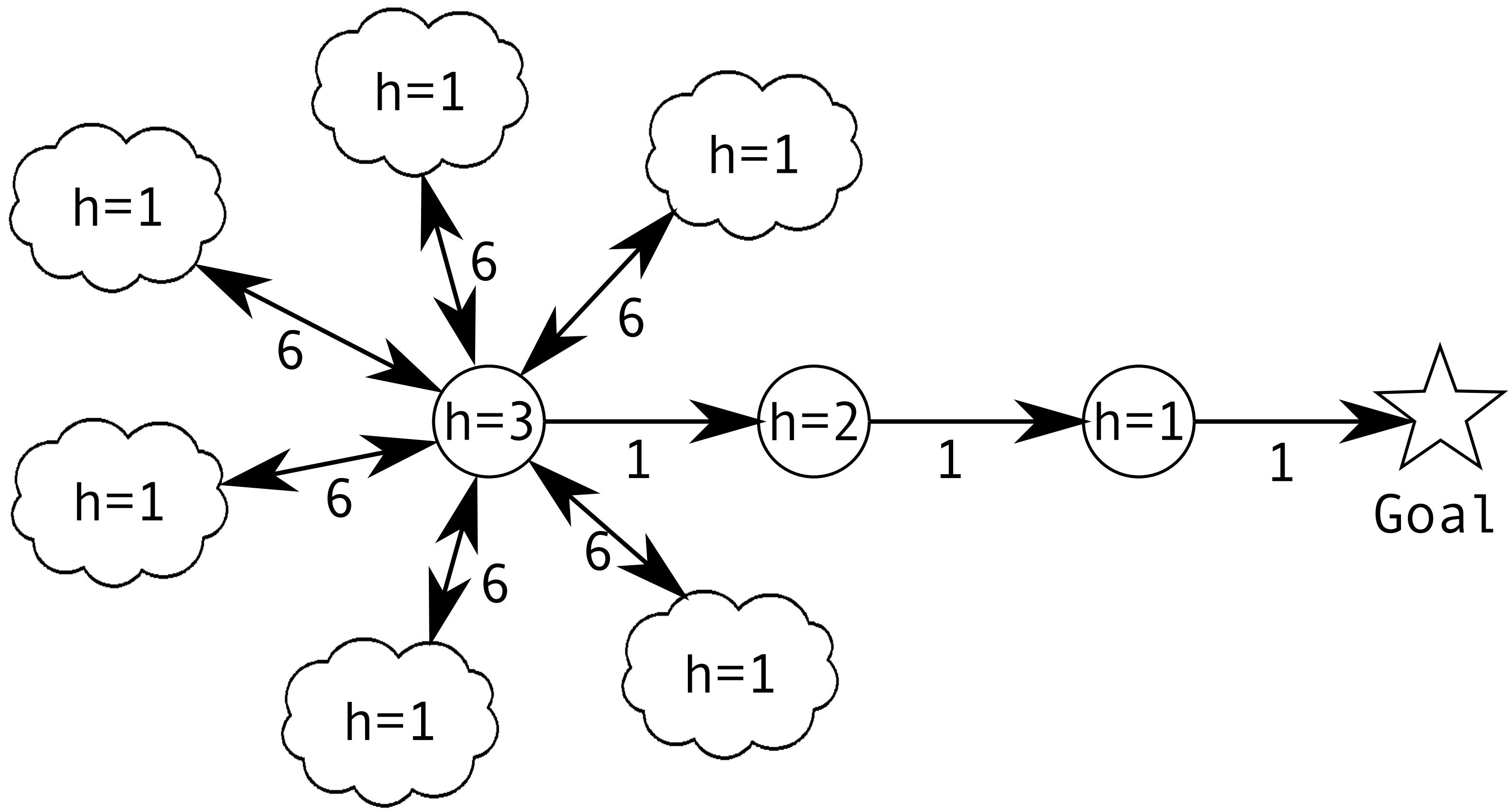
Local Minimum Size

Greedy search might do well in domains where local minima are small.

Local Minimum Size



Local Minimum Size



Local Minimum Size

Greedy search can do well when local minima are large, and poorly when local minima are small.

$h^*(n) - h(n)$ Correlation

- $h^*(n) - h(n)$ Correlation

- $h^*(n) - h(n)$ Correlation

Domain	Correlation
Hanoi	0.9652
Citynav 4 4	0.7077
Unit Tiles	0.7064
Inverse Tiles	0.6722
TopSpin(3)	0.5855
TopSpin(4)	0.2827

- $h^*(n) - h(n)$ Correlation

Domain	Correlation
Hanoi	0.9652
Citynav 4 4	0.7077
Unit Tiles	0.7064
Inverse Tiles	0.6722
TopSpin(3)	0.5855
TopSpin(4)	0.2827

- $h^*(n) - h(n)$ Correlation cannot be used to identify domains where greedy search works poorly.

- $d^*(n) - h(n)$ Correlation

- $d^*(n) - h(n)$ Correlation

Domain	Correlation
Robot	0.9989
Grid	0.9790
Hanoi	0.9652
Pancake	0.9621
Unit Tiles	0.7064
TopSpin(3)	0.5855
Citynav 3 3	0.3670
Citynav 4 4	0.2827
TopSpin(4)	0.0246
Inverse Tiles	0.0853

-

- $d^*(n) - h(n)$ Correlation

Domain	Correlation
Robot	0.9989
Grid	0.9790
Hanoi	0.9652
Pancake	0.9621
Unit Tiles	0.7064
TopSpin(3)	0.5855
Citynav 3 3	0.3670
Citynav 4 4	0.2827
TopSpin(4)	0.0246
Inverse Tiles	0.0853

- $d^*(n) - h(n)$ Correlation **can** be used to identify domains where greedy search works poorly.

Conclusion

Conclusion

Greedy search fails when the correlation between $d^*(n) - h(n)$ is weak.