Planning Algorithms: When Optimal Is Just Not Good Enough

Wheeler Ruml

UNIVERSITY of NEW HAMPSHIRE

Department of Computer Science

Joint work with the UNH AI Group. Support from NSF and DARPA.

Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – $1\ /\ 34$

Robot Time!

Introduction

Robot Time!An Agent

Roles

Planning as Search

Optimal Planning

Bounded Suboptimal

Conclusion

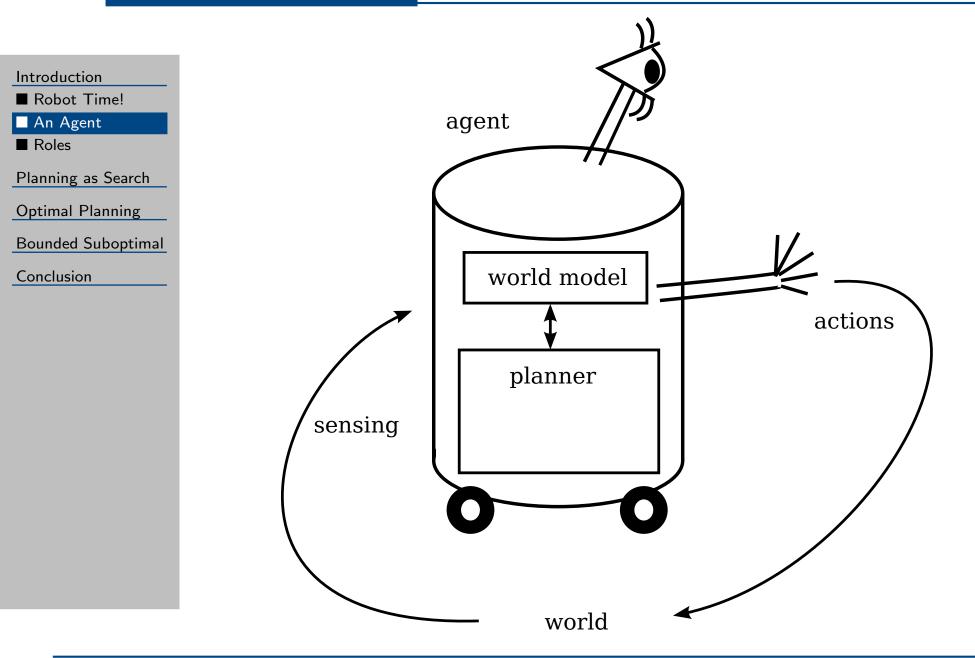




Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 2 / 34

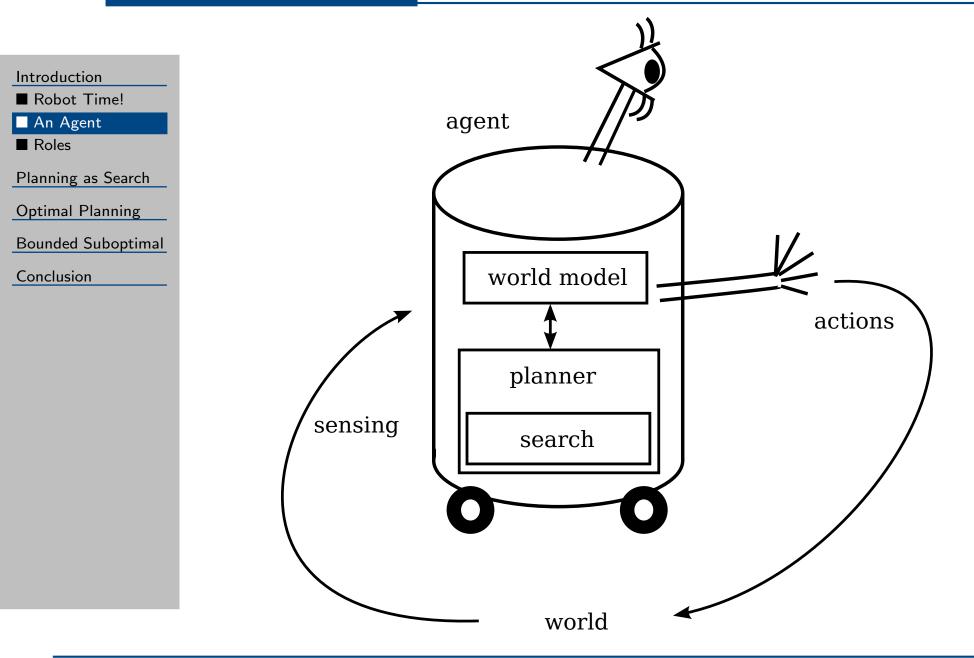
The Role of Planning



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 3 / 34

The Role of Planning



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 3 / 34

The Roles of Planning

Introduction

■ Robot Time!

An Agent

Roles

Planning as Search

Optimal Planning

Bounded Suboptimal

Conclusion

autonomy

- exploration
- transportation
- manufacturing
- autonomic systems
- decision support
 - operations management
 - personal health
 - eldercare
 - education





Introduction

Planning as Search

- Planning
- Graph Search
- Example
- 3 Algorithms

Optimal Planning

Bounded Suboptimal

Conclusion

Planning as Heuristic Graph Search

Planning

Introduction

- Planning as Search
- Planning
- Graph Search
- Example
- 3 Algorithms
- Optimal Planning
- Bounded Suboptimal

Conclusion

Given:

- current state of the world
- models of available actions preconditions, effects, costs
- desired state of the world (partially specified?)

Find:

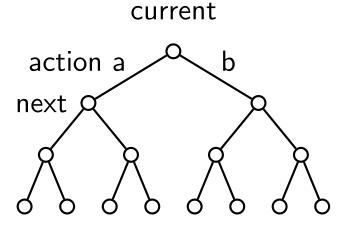
cheapest plan

Graph Search

Introduction	Given:	
 Planning as Search Planning Graph Search Example 3 Algorithms Optimal Planning 	 start stat expand f lazily g goal test 	
Bounded Suboptimal Conclusion	Find:	

- te: an explicit node
- function:
 - generate children and their costs
- t: predicate on nodes

cheapest path to a goal node



Wheeler Ruml (UNH)

Graph Search

1 . 1	
Introdu	lction

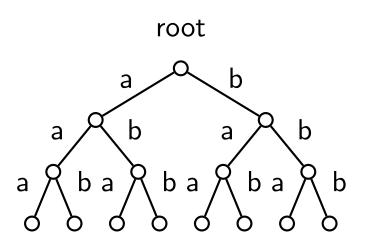
- Planning as Search
- Planning
- Graph Search
- Example
- 3 Algorithms
- **Optimal Planning**
- Bounded Suboptimal
- Conclusion

Given:

- start state: an explicit node
- I expand function:
 - lazily generate children and their costs
- goal test: predicate on nodes

Find:

cheapest path to a goal node



Graph Search

Introduction	Introduction	
--------------	--------------	--

- Planning as Search
- Planning
- Graph Search
- Example
- 3 Algorithms
- **Optimal Planning**
- Bounded Suboptimal

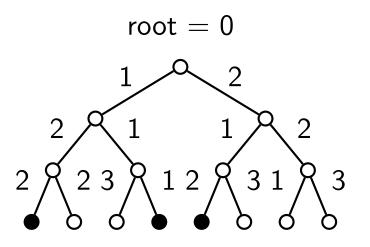
Conclusion

Given:

- start state: an explicit node
- I expand function:
 - lazily generate children and their costs
- goal test: predicate on nodes

Find:

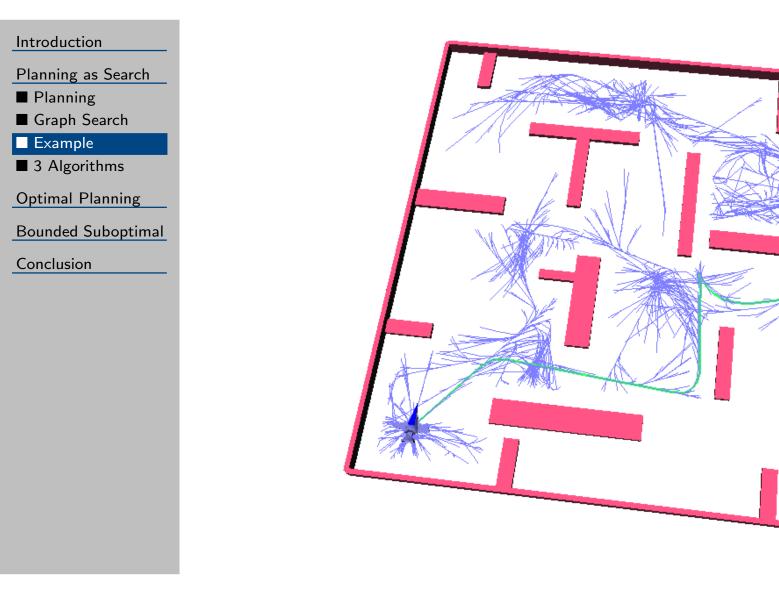
cheapest path to a goal node



When Optimal Is Just Not Good Enough – 7 / 34

Wheeler Ruml (UNH)

Example: Motion Planning



(S. LaValle)

When Optimal Is Just Not Good Enough – 8 / 34

Wheeler Ruml (UNH)

Heuristic Search: From Basic to State-of-the-Art

Introduction

Planning as Search

- Planning
- Graph Search
- Example
- 3 Algorithms

Optimal Planning

Bounded Suboptimal

Conclusion

- 1. Uniform Cost Search (Dijkstra, 1959)
- 2. A* Search (Hart, Nilsson, and Raphael, 1968)
- 3. Explicit Estimation Search (Thayer and Ruml, 2011)

Introduction

Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

Conclusion

Optimal Planning

Introduction

Planning as Search

Optimal Planning

Uniform Search

■ Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound

■ Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

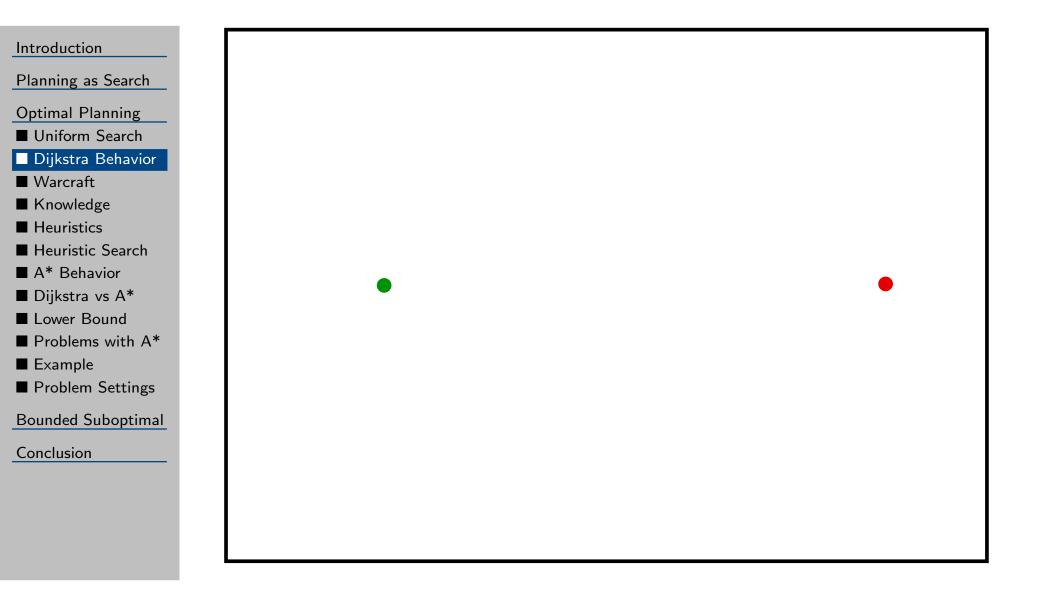
Uniform Cost Search (Dijkstra, 1959)

Introduction	Explore nodes in increasing order of cost-so-far $(g(n))$:
Planning as Search	
Optimal Planning	
Uniform Search	
Dijkstra Behavior	
Warcraft	
■ Knowledge	
Heuristics	
Heuristic Search	
A* Behavior	
Dijkstra vs A*	
■ Lower Bound	
Problems with A*	
Example	
Problem Settings	
Bounded Suboptimal	
Conclusion	

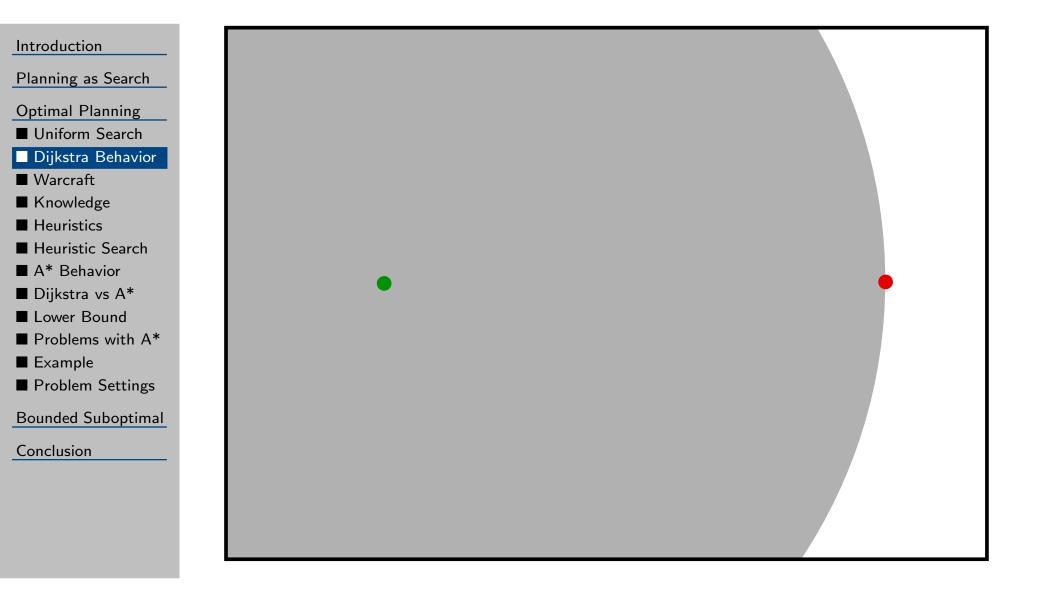
Uniform Cost Search (Dijkstra, 1959)

Introduction	Explore nodes in increasing order of cost-so-far $(g(n))$:
 Planning as Search Optimal Planning Uniform Search Dijkstra Behavior Warcraft Knowledge Heuristics Heuristic Search A* Behavior Dijkstra vs A* Lower Bound Problems with A* Example 	$open \leftarrow$ ordered list containing the initial state Loop If open is empty, return failure Node \leftarrow pop cheapest node off open If Node is a goal, return it (or path to it) Children \leftarrow Expand(Node). Merge Children into open, keeping sorted by $g(n)$.
Problem Settings Bounded Suboptimal Conclusion	$\begin{array}{c} 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \\ 3 \\ 1 \\ 2 \\ 3 \\ 1 \\ 2 \\ 3 \\ 1 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3 \\ 3$

Dijkstra Behavior



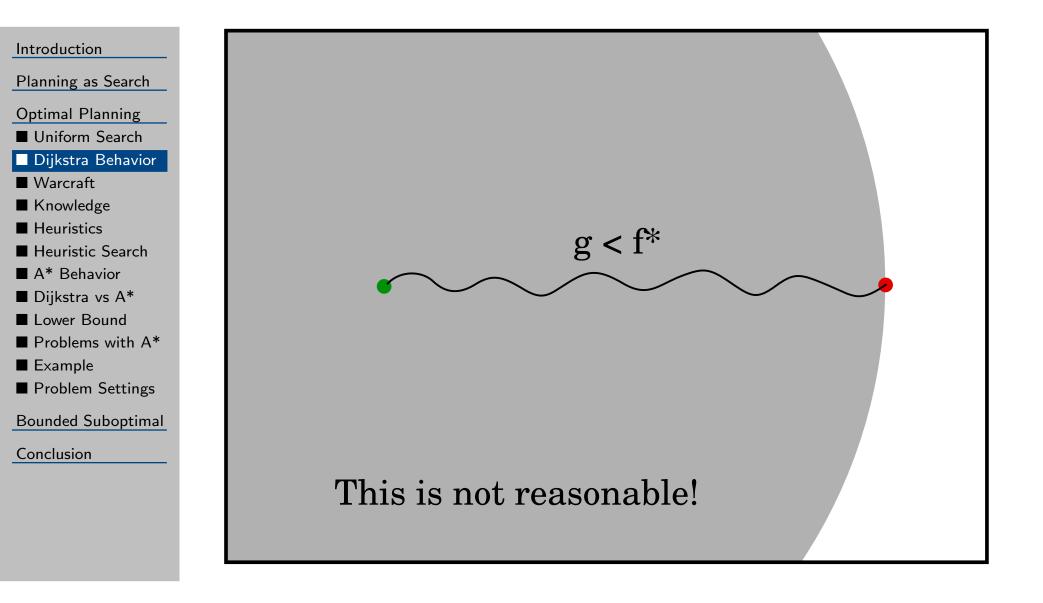
Dijkstra Behavior



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 12 / 34

Dijkstra Behavior



Dijkstra: Pathfinding in Warcraft

Introduction

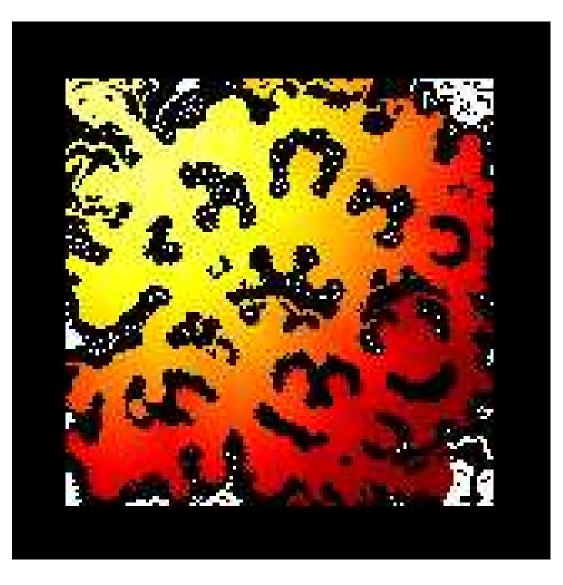
Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- \blacksquare Problems with A*
- Example
- Problem Settings

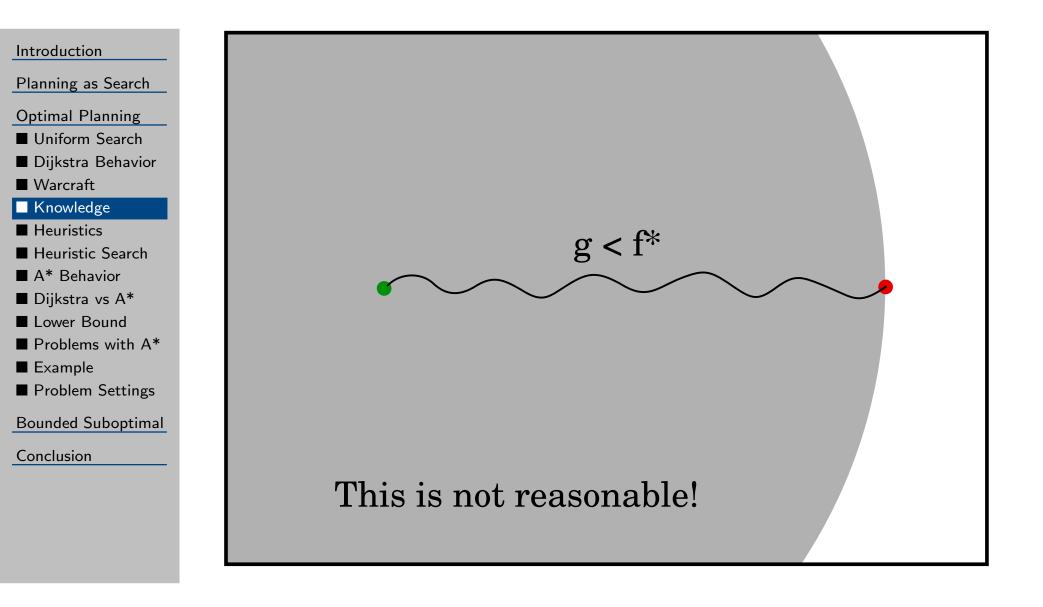
Bounded Suboptimal

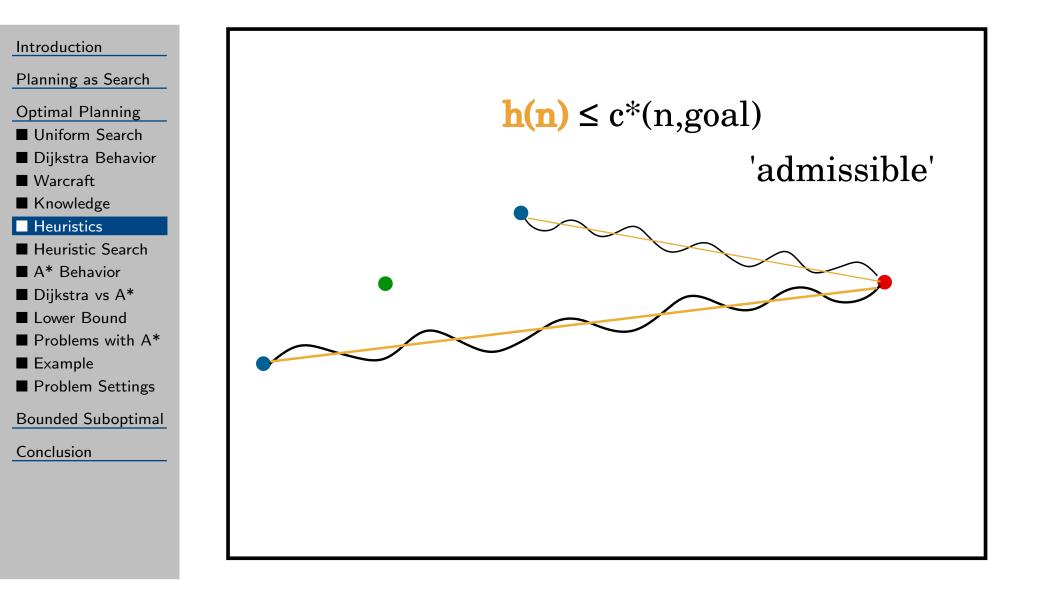
Conclusion



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 13 / 34





Introduction

Planning as Search

Optimal Planning

■ Uniform Search

Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

Heuristic Search

■ A* Behavior

■ Dijkstra vs A*

Lower Bound

 \blacksquare Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

Heuristic Search: A* (Hart, Nilsson, and Raphael, 1968)

Introduction

Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

Conclusion

Cost should include both cost-so-far and cost-to-go:

g(n) = cost incurred so far

h(n) = lower bound on cost to goal

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

Heuristic Search: A* (Hart, Nilsson, and Raphael, 1968)

Introduction

Planning as Search

Optimal Planning

■ Uniform Search

Dijkstra Behavior

■ Warcraft

Knowledge

Heuristics

```
Heuristic Search
```

■ A* Behavior

Dijkstra vs A*

Lower Bound

Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

Cost should include both cost-so-far and cost-to-go:

```
g(n) = \text{cost incurred so far}

h(n) = \text{lower bound on cost to goal}

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

 $open \leftarrow ordered list containing the initial state Loop$

```
If open is empty, return failure

Node \leftarrow pop cheapest node off open

If Node is a goal, return it (or path to it)

Children \leftarrow Expand(Node)

Merge Children into open, keeping sorted by f(n)
```

Heuristic Search: A* (Hart, Nilsson, and Raphael, 1968)

Introduction Planning as Search **Optimal Planning** ■ Uniform Search ■ Dijkstra Behavior ■ Warcraft ■ Knowledge Heuristics Heuristic Search ■ A* Behavior Loop ■ Dijkstra vs A* Lower Bound Problems with A* Example Problem Settings Bounded Suboptimal Conclusion

Cost should include both cost-so-far and cost-to-go:

```
g(n) = \text{cost incurred so far}
h(n) = \text{lower bound on cost to goal}
f(n) = g(n) + h(n)
```

 $open \leftarrow ordered \ list \ containing \ the \ initial \ state$ Loop

```
If open is empty, return failure

Node \leftarrow pop cheapest node off open

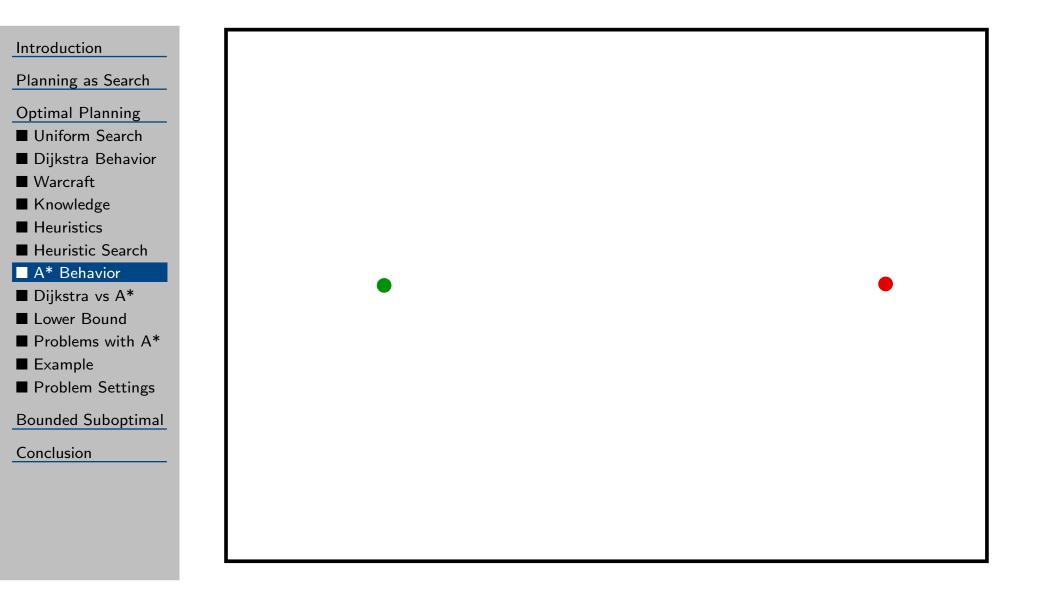
If Node is a goal, return it (or path to it)

Children \leftarrow Expand(Node)

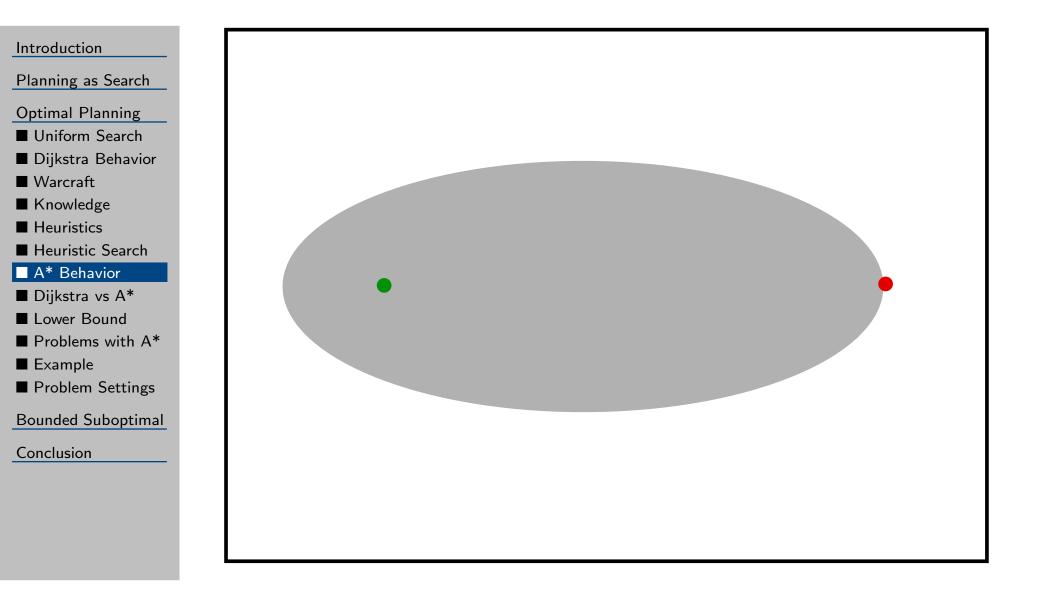
Merge Children into open, keeping sorted by f(n)
```

finds optimal solution if heuristic is admissible

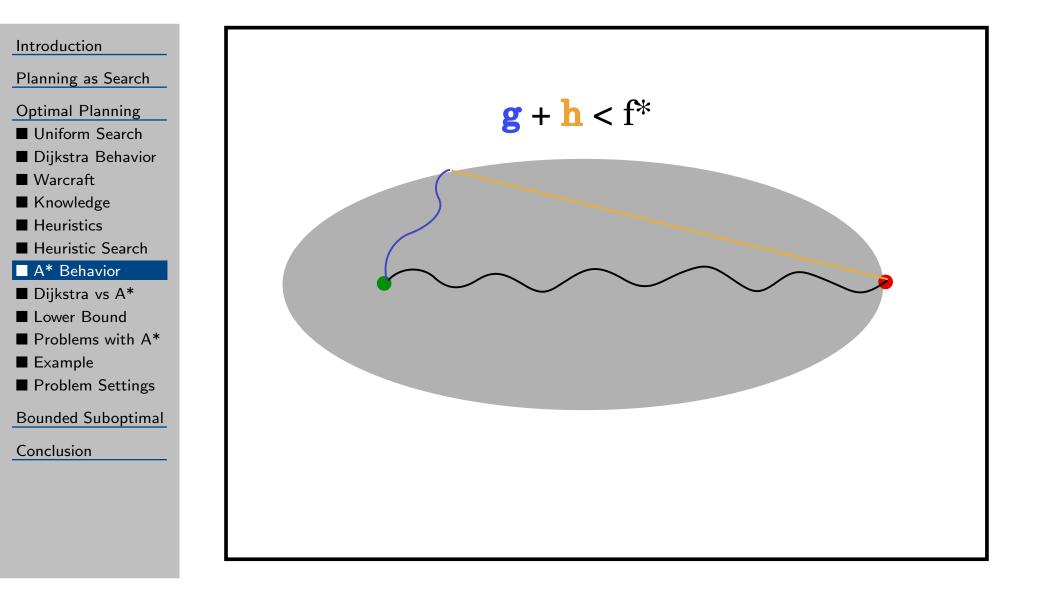
A* Behavior



A* Behavior



A* Behavior



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 17 / 34

Dijkstra vs A*: Pathfinding in Warcraft

Introduction

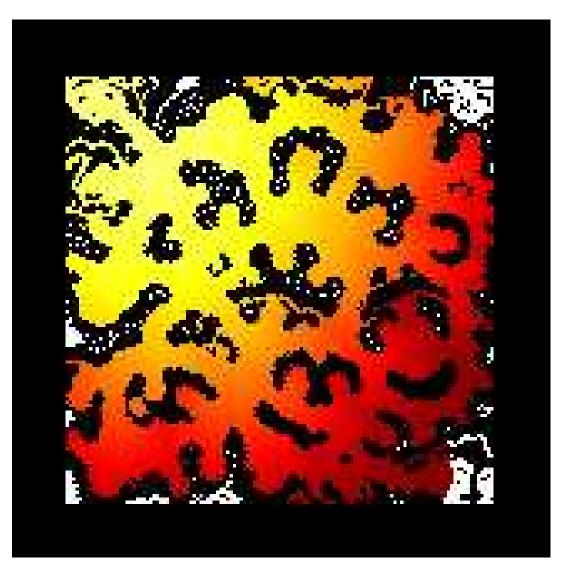
Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- \blacksquare Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

Conclusion



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 18 / 34

Dijkstra vs A*: Pathfinding in Warcraft

Introduction

Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

Heuristic Search

■ A* Behavior

Dijkstra vs A*

■ Lower Bound

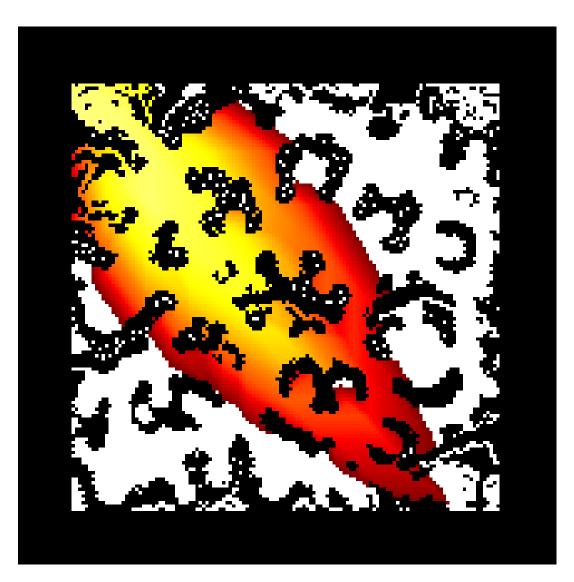
 \blacksquare Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion



When Optimal Is Just Not Good Enough – 18 / 34

Quick Note: A Lower Bound on Solution Cost

Introduction

Planning as Search

Optimal Planning

Uniform Search

Dijkstra Behavior

■ Warcraft

Knowledge

Heuristics

Heuristic Search

■ A* Behavior

■ Dijkstra vs A*

Lower Bound

 \blacksquare Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

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

g(n) is actual cost-so-far, so when h(n) is a lower bound on cost-to-go, f(n) is a lower bound on cost of plan through n

Quick Note: A Lower Bound on Solution Cost

Introduction

Planning as Search

- **Optimal Planning**
- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

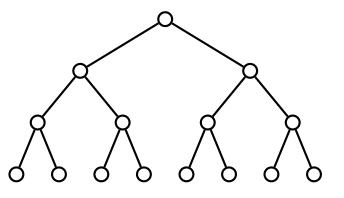
Conclusion

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

g(n) is actual cost-so-far, so when h(n) is a lower bound on cost-to-go, f(n) is a lower bound on cost of plan through n

lowest f(n) on frontier gives lower bound for entire problem!

$$best_f = \operatorname*{argmin}_{n \in open} f(n)$$



Problems with A*

Introduction

Planning as Search

Optimal Planning

■ Uniform Search

■ Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

Heuristic Search

■ A* Behavior

■ Dijkstra vs A*

Lower Bound

Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

Problems with A*

Introduction

Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

Conclusion

A* takes exponential memory

Problems with A*

Introduction

Planning as Search

Optimal Planning

- Uniform Search
- Dijkstra Behavior
- Warcraft
- Knowledge
- Heuristics
- Heuristic Search
- A* Behavior
- Dijkstra vs A*
- Lower Bound
- Problems with A*
- Example
- Problem Settings

Bounded Suboptimal

Conclusion

A* takes exponential memory

Can sometimes be fixed: see 'iterative deepening'

Problems with A*

Introduction

Planning as Search

Optimal Planning

Uniform Search

Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

Heuristic Search

■ A* Behavior

■ Dijkstra vs A*

■ Lower Bound

Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

A* takes exponential memory

Can sometimes be fixed: see 'iterative deepening'

A* takes exponential time

Helmert and Röger, "How Good is Almost Perfect?" AAAI-08

Problems with A*

Introduction

Planning as Search

Optimal Planning

Uniform Search

Dijkstra Behavior

■ Warcraft

■ Knowledge

Heuristics

Heuristic Search

■ A* Behavior

■ Dijkstra vs A*

■ Lower Bound

Problems with A*

Example

Problem Settings

Bounded Suboptimal

Conclusion

A* takes exponential memory

Can sometimes be fixed: see 'iterative deepening'

A* takes exponential time

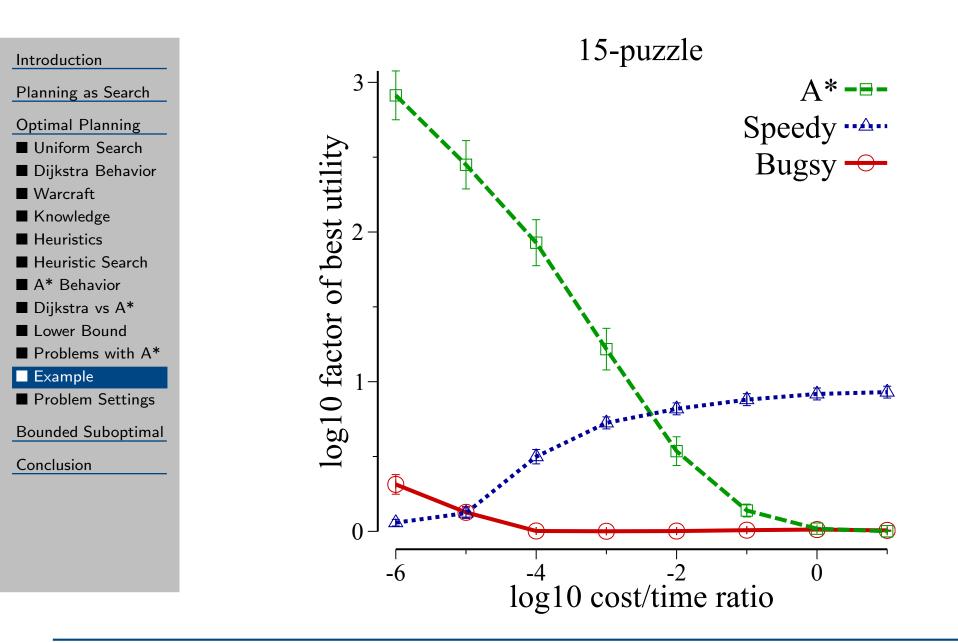
Helmert and Röger, "How Good is Almost Perfect?" AAAI-08

We must trade cost for time.

Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 20 / 34

Optimizing Utility of Cost + Time



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 21 / 34

A New Generation of Problem Settings

Introduction Planning as Search	optimal:	minimize solution cost must expand all with $f(n) < f^*(opt)$
Planning as Search Optimal Planning Uniform Search Dijkstra Behavior Warcraft Knowledge Heuristics Heuristic Search A* Behavior Dijkstra vs A* Lower Bound Problems with A* Example Problem Settings		must expand all with $f(n) < f^*(opt)$
Bounded Suboptimal Conclusion		

A New Generation of Problem Settings

Introduction Planning as Search	optimal: minimize solution cost must expand all with $f(n) < f^*(opt)$
Optimal Planning Uniform Search Dijkstra Behavior Warcraft Knowledge 	greedy: minimize solving time
 Heuristics Heuristic Search A* Behavior 	anytime: incrementally converge to optimal
 Dijkstra vs A* Lower Bound Problems with A* Example 	bounded suboptimal: minimize time subject to relative cost bound (factor of optimal)
Problem Settings Bounded Suboptimal	bounded cost: minimize time subject to absolute cost bound
<u>Conclusion</u>	contract: minimize cost subject to absolute time bound utility function: maximize utility function of cost and time
	eg, goal achievement time = plan makespan + search time

A New Generation of Problem Settings

Introduction Planning as Search	optimal: minimize solution cost must expand all with $f(n) < f^*(opt)$
Optimal Planning Uniform Search Dijkstra Behavior Warcraft Knowledge	greedy: minimize solving time
 Heuristics Heuristic Search A* Behavior Dijkstra vs A* Lower Bound Problems with A* 	anytime: incrementally converge to optimal bounded suboptimal: minimize time subject to relative cost bound (factor of optimal)
 Froblems with A^A Example Problem Settings Bounded Suboptimal 	bounded cost: minimize time subject to absolute cost bound
Conclusion	contract: minimize cost subject to absolute time bound utility function: maximize utility function of cost and time
	eg, goal achievement time = plan makespan + search time

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

■ Three Heuristics

■ Strategy

Expansion Order

Bound

■ EES Performance

Conclusion

Bounded Suboptimal Search

Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 23 / 34

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

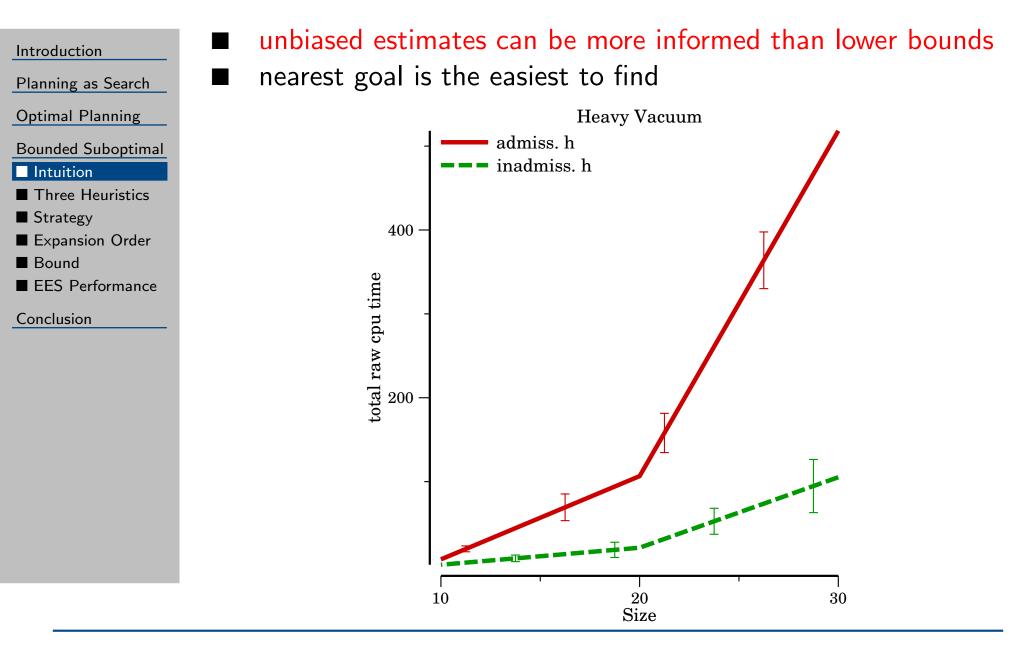
Three Heuristics

Strategy

- Expansion Order
- Bound
- EES Performance

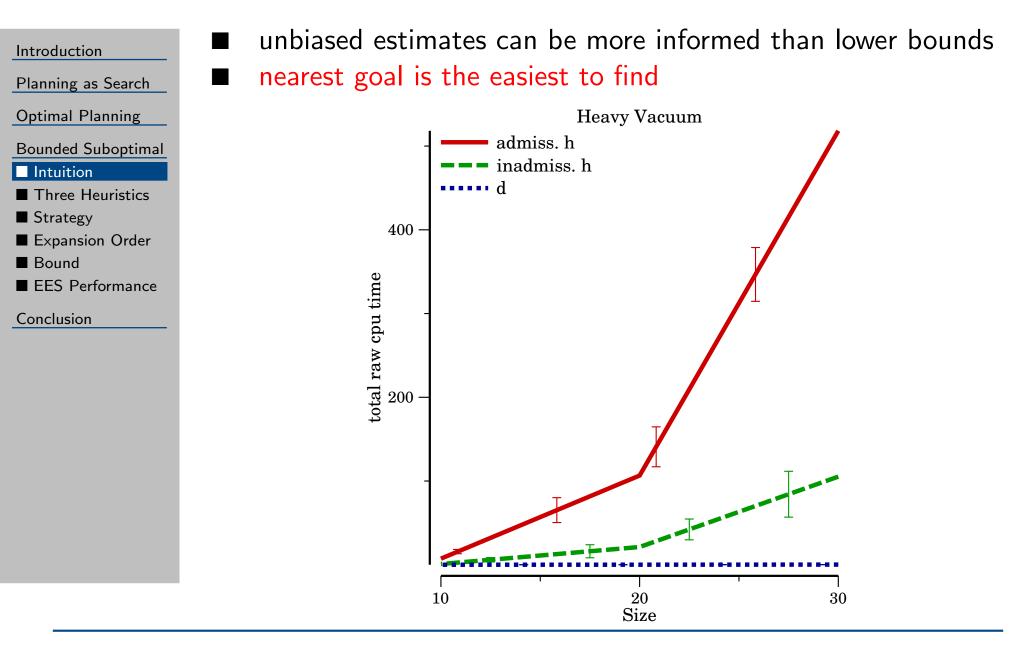
Conclusion

unbiased estimates can be more informed than lower bounds nearest goal is the easiest to find



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 24 / 34



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough -24 / 34

Introduction Planning as Search **Optimal Planning Bounded Suboptimal** Intuition ■ Three Heuristics ■ Strategy Expansion Order Bound ■ EES Performance Conclusion

unbiased estimates can be more informed than lower bounds nearest goal is the easiest to find

minimize solving time subject to cost $\leq w \cdot optimal$:

pursue nearest goal estimated to lie within bound

need more information than just lower bound on cost (h(n))!

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

Three Heuristics

■ Strategy

Expansion Order

Bound

■ EES Performance

Conclusion

1. h: a lower bound on cost-to-go f(n) = g(n) + h(n)the traditional optimal A* lower bound

roduction	1.
nning as Search	
timal Planning	
unded Suboptimal	
ntuition	<u> </u>
Three Heuristics	2.
Strategy	
Expansion Order	
Davind	

Bound
EES Performance

Intr

Pla

Opt

Boı ∎ I

Conclusion

h: a lower bound on cost-to-go f(n) = g(n) + h(n) the traditional optimal A* lower bound

 \widehat{h} : an estimate of cost-to-go unbiased estimates can be more informed $\widehat{f}(n) = g(n) + \widehat{h}(n)$ (Thayer and Ruml, ICAPS-11)

troduction	1.
anning as Search	
ptimal Planning	
ounded Suboptimal	
Intuition	~
Three Heuristics	2.
Strategy	
Expansion Order	
Bound	
EES Performance	

Conclusion

Int

- h: a lower bound on cost-to-go f(n) = g(n) + h(n) the traditional optimal A* lower bound
- 2. \widehat{h} : an estimate of cost-to-go unbiased estimates can be more informed $\widehat{f}(n) = g(n) + \widehat{h}(n)$ (Thayer and Ruml, ICAPS-11)
- 3. \widehat{d} : an estimate of distance-to-go nearest goal is the easiest to find (Pearl and Kim, IEEE PAMI 1982, Thayer et al, ICAPS-09)

Introduction1.Planning as SearchOptimal PlanningBounded SuboptimalIntuition2.Three Heuristics2.StrategyExpansion OrderBoundEES Performance

Conclusion

1. h: a lower bound on cost-to-go f(n) = g(n) + h(n)the traditional optimal A* lower bound

2. \widehat{h} : an estimate of cost-to-go unbiased estimates can be more informed $\widehat{f}(n) = g(n) + \widehat{h}(n)$ (Thayer and Ruml, ICAPS-11)

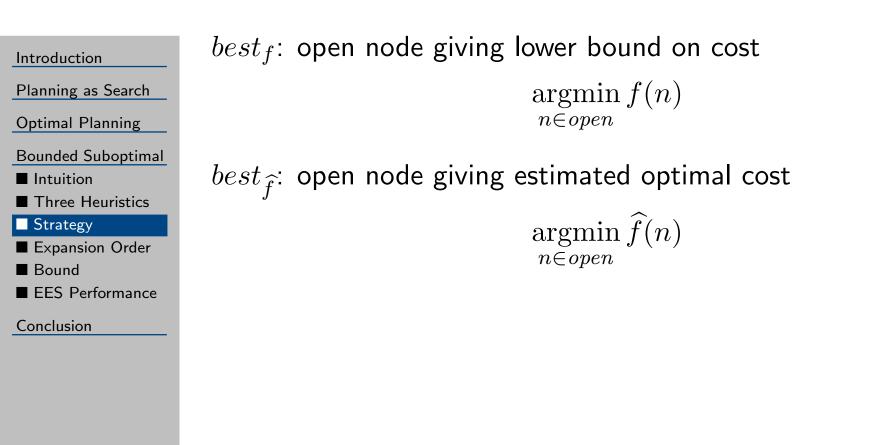
3. \widehat{d} : an estimate of distance-to-go nearest goal is the easiest to find (Pearl and Kim, IEEE PAMI 1982, Thayer et al, ICAPS-09)

pursue nearest goal estimated to lie within bound

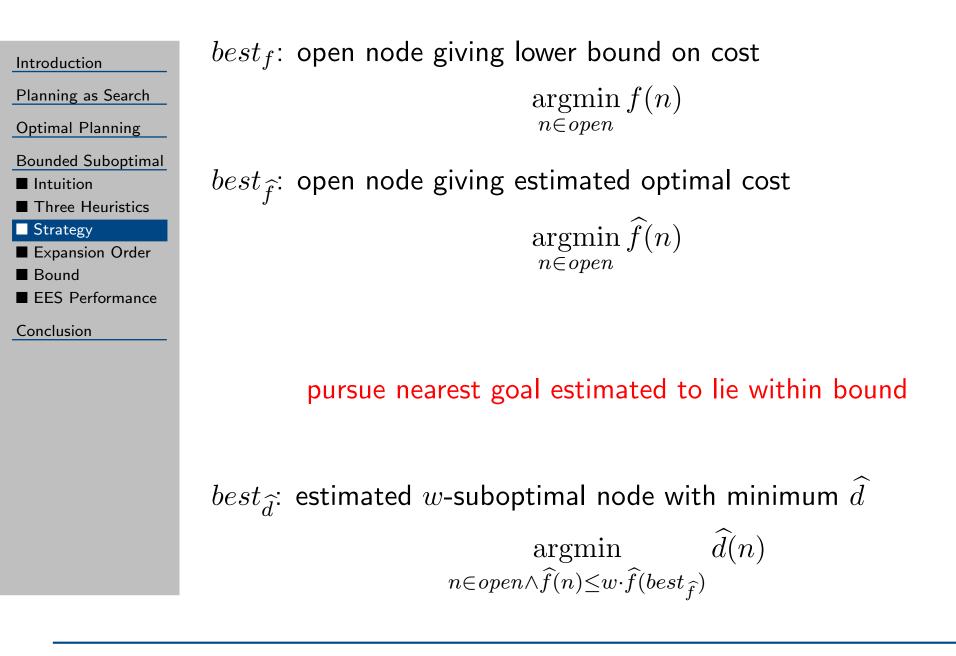
Search Strategy: Which Node to Expand?

Introduction	$best_f$: open node giving lower bound on cost
<u>Planning as Search</u> Optimal Planning	$\operatorname*{argmin}_{n \in open} f(n)$
Bounded Suboptimal Intuition	
 Three Heuristics Strategy Evennesion Order 	
 Expansion Order Bound EES Performance 	
Conclusion	

Search Strategy: Which Node to Expand?



Search Strategy: Which Node to Expand?



Introduction
Planning as Search
Optimal Planning
Bounded Suboptimal
■ Intuition
Three Heuristics
■ Strategy
Expansion Order
Bound
■ EES Performance
Conclusion

 $\begin{array}{l} best_{f}: \text{ open node giving lower bound on cost} \\ best_{\widehat{f}}: \text{ open node giving estimated optimal cost} \\ best_{\widehat{d}}: \text{ estimated } w\text{-suboptimal node with minimum } \widehat{d} \end{array}$

node to expand next:

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

in other words:

1. $best_{\widehat{d}}$

2.

3.

3.

Introduction
Planning as Search
Optimal Planning
Bounded Suboptimal
Intuition
Three Heuristics
■ Strategy
Expansion Order
Bound
■ EES Performance
Conclusion

 $\begin{array}{l} best_{f}: \text{ open node giving lower bound on cost} \\ best_{\widehat{f}}: \text{ open node giving estimated optimal cost} \\ best_{\widehat{d}}: \text{ estimated } w\text{-suboptimal node with minimum } \widehat{d} \end{array}$

node to expand next:

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

in other words:

1. if
$$\widehat{f}(best_{\widehat{d}}) \leq w \cdot f(best_f)$$
 then $best_{\widehat{d}}$

2.

3.

3.

Introduction	be
Planning as Search	be
Optimal Planning	he
Bounded Suboptimal	00
Intuition	
Three Heuristics	
	nc
■ Strategy	nc
	nc 1.
■ Strategy	
StrategyExpansion Order	nc 1. 2.

 $\begin{array}{l} best_{f}: \text{ open node giving lower bound on cost} \\ best_{\widehat{f}}: \text{ open node giving estimated optimal cost} \\ best_{\widehat{d}}: \text{ estimated } w \text{-suboptimal node with minimum } \widehat{d} \end{array}$

node to expand next:

pursue the nearest goal estimated to lie within the bound
 pursue the estimated optimal solution

in other words:

- 1. if $\widehat{f}(best_{\widehat{d}}) \leq w \cdot f(best_f)$ then $best_{\widehat{d}}$
- 2. else if $\widehat{f}(best_{\widehat{f}}) \leq w \cdot f(best_f)$ then $best_{\widehat{f}}$
- 3.

Introduction	
mulouuction	

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

Three Heuristics

- Strategy
- Expansion Order
- Bound
- EES Performance

Conclusion

 $\begin{array}{l} best_f: \text{ open node giving lower bound on cost} \\ best_{\widehat{f}}: \text{ open node giving estimated optimal cost} \\ best_{\widehat{d}}: \text{ estimated } w\text{-suboptimal node with minimum } \widehat{d} \end{array}$

node to expand next:

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

in other words:

- 1. if $\widehat{f}(best_{\widehat{d}}) \leq w \cdot f(best_f)$ then $best_{\widehat{d}}$
- 2. else if $\widehat{f}(best_{\widehat{f}}) \leq w \cdot f(best_f)$ then $best_{\widehat{f}}$
- 3. else $best_f$

see paper for further justification. Note: no magic numbers!

EES Respects the Suboptimality Bound

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

■ Three Heuristics

■ Strategy

Expansion Order

Bound

■ EES Performance

Conclusion

how does $\widehat{f}(n) \leq w \cdot f(best_f)$ ensure the suboptimality bound?

EES Respects the Suboptimality Bound

Introduction Planning as Search Optimal Planning

Bounded Suboptimal

- Intuition
- Three Heuristics
- Strategy
- Expansion Order
- Bound
- EES Performance

Conclusion

how does
$$\widehat{f}(n) \leq w \cdot f(best_f)$$
 ensure the suboptimality bound?

$$\begin{array}{rcl} f(n) & \leq & \widehat{f}(n) & f(n) \text{ is a lower bound for } n \\ \widehat{f}(n) & \leq & w \cdot f(best_f) & \text{expansion criterion} \\ w \cdot f(best_f) & \leq & w \cdot f^*(opt) & \text{because } f(best_f) \text{ is a lower} \\ & & bound \text{ for the entire problem} \\ \hline f(n) & \leq & w \cdot f^*(opt) & \text{suboptimality bound} \end{array}$$

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Intuition

Three Heuristics

Strategy

Expansion Order

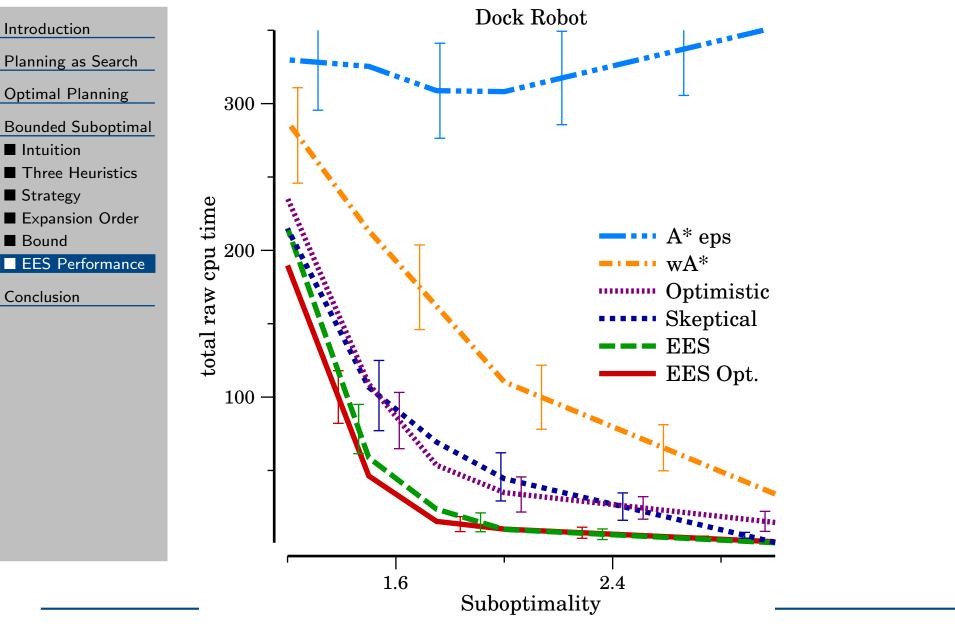
Bound

EES Performance

Conclusion

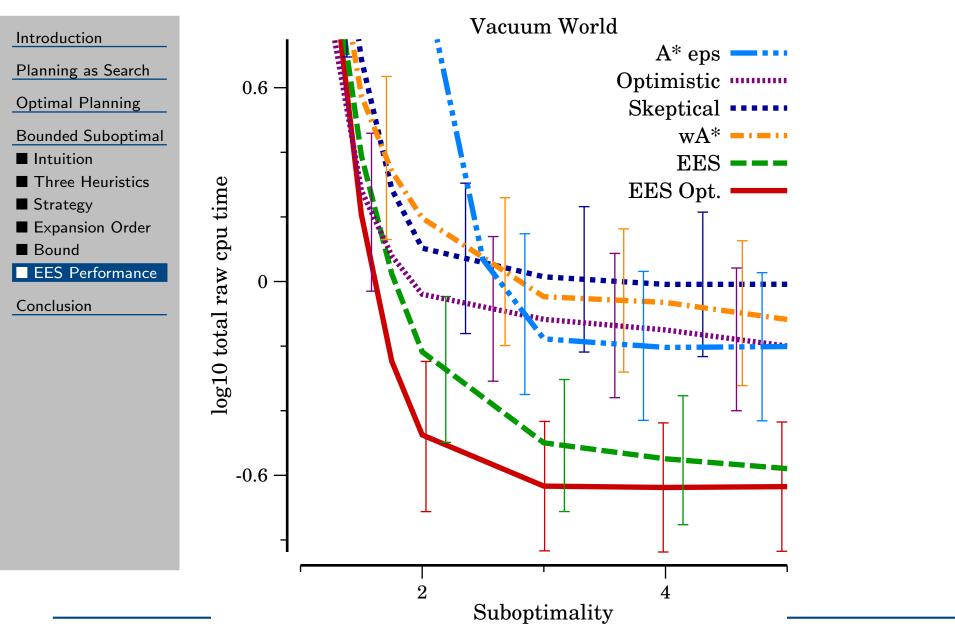
bounded suboptimal search:

minimize time subject to relative cost bound (factor of optimal)



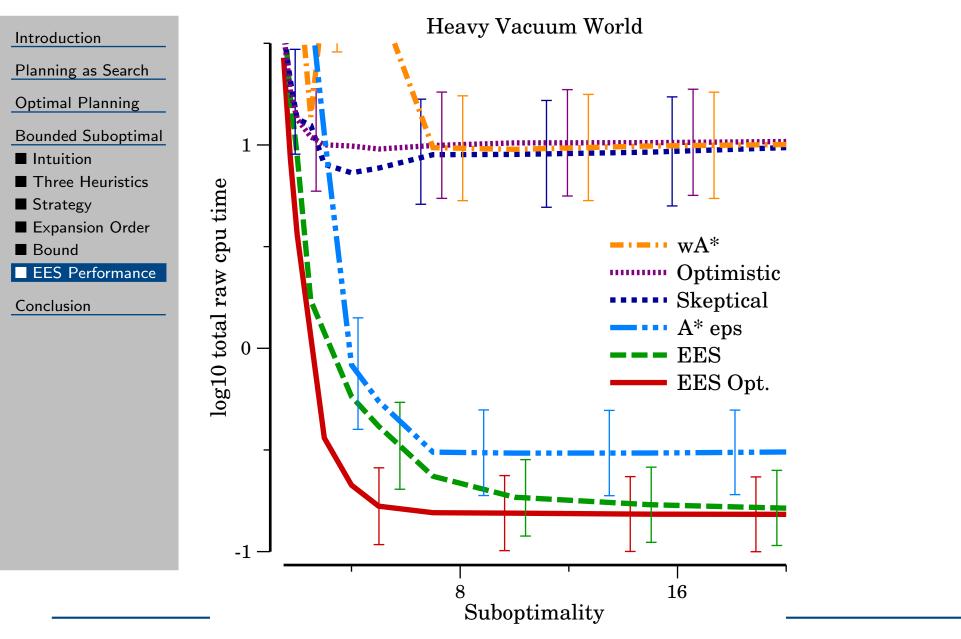
Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 29 / 34



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 29 / 34



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough - 29 / 34

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Conclusion

■ Subopt. Search

■ The AI Vision

■ Algs as Agents

Conclusion

Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 30 / 34

A Science of Suboptimal Search

Introduction

Planning as Search

Optimal Planning

Bounded Suboptimal

Conclusion

Subopt. Search

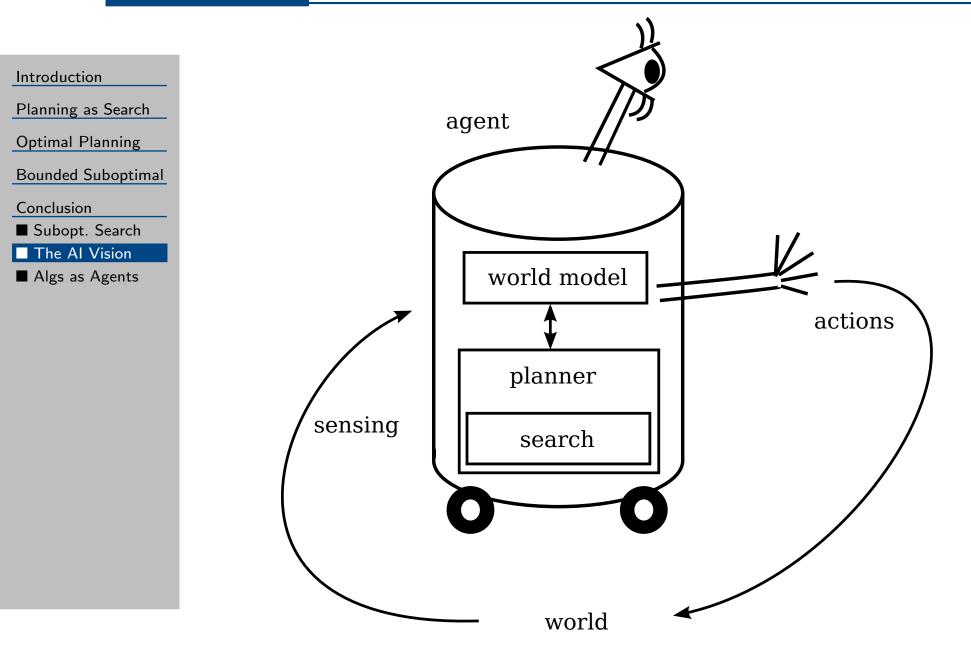
The AI Vision

Algs as Agents

what are the problem settings?

- guarantees beyond optimal search
- eg, bounded suboptimal search
- utility-based optimization
- what are the sources of information?
 - where do inadmissible heuristics come from?
 - estimates instead of lower bounds
 - distance in addition to cost
- how to exploit and combine information?
 - new generation of suboptimal algorithms
 - meta-reasoning

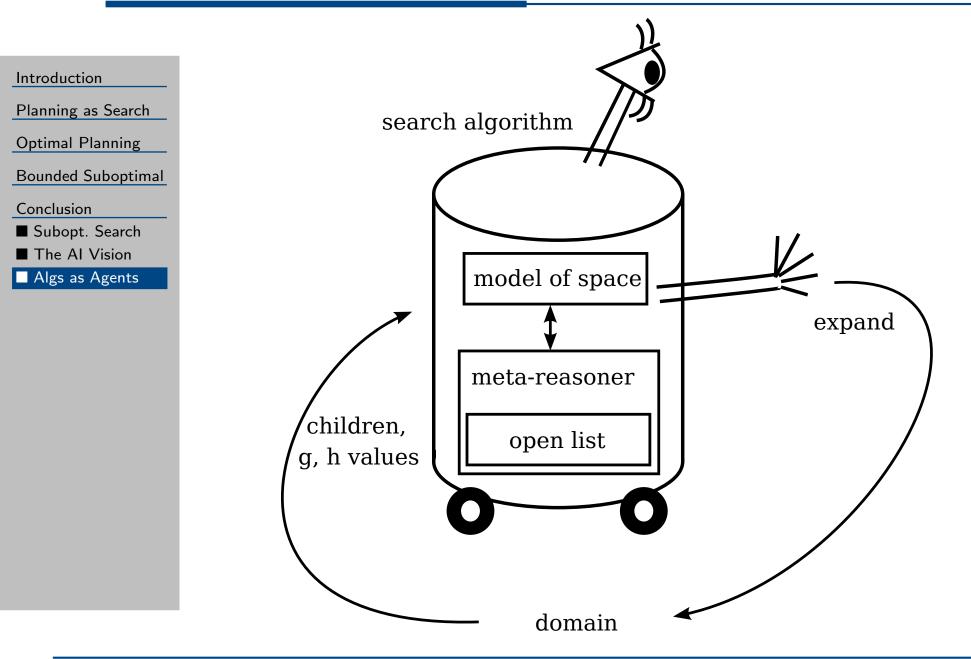
The AI Vision



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough – 32 / 34

Search Algorithms as Agents



Wheeler Ruml (UNH)

When Optimal Is Just Not Good Enough - 33 / 34



The University of New Hampshire