Finding Acceptable Solutions Faster Using Inadmissible Information

Jordan Thayer & Wheeler Ruml

University of New Hampshire

Supported by DARPA CSSG program (grant HR0011-09-1-0021) and NSF (grant IIS-0812141)
Grid 4-Way 35% Obstacles

We want speed like this.

We want cost like this.

Motivation

EES

Results

Bounded Suboptimal Heuristic Search

Jordan Thayer (UNH)
<table>
<thead>
<tr>
<th>Motivation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>EES</td>
<td>Guarantee the solution is within a factor $w$ of optimal. Solution is $w$-admissible</td>
</tr>
<tr>
<td>Results</td>
<td>Find solutions as quickly as you can within the bound.</td>
</tr>
</tbody>
</table>
Guarantee the solution is within a factor $w$ of optimal.

Solution is $w$-admissible

Find solutions as quickly as you can within the bound.

- Weighted A*
Pohl, 1970

- Dynamically Weighted A*
Pohl, 1973

- $A^*_\epsilon$
Pearl, 1982

- $A^*_\epsilon$
Ghallad & Allard, 1983

- AlphA*
Reese, 1999

- Clamped Adaptive
Thayer, Ruml, & Bitton
2008

- Optimistic Search
Thayer & Ruml, 2008

- Revised Dynamically wA*
Thayer & Ruml, 2009
Introduce Two Opportunities to Improve Bounded Suboptimal Search

Using Inadmissible Heuristics

Paying attention to differences in cost and distance

Present EES, Which Exploits Them

Show Selected Results
Inadmissible Estimates Outperform Admissible Estimates

Motivation

Summary

Vacuum World: Greedy Search Guidance

Total nodes generated

Admissible h

Inadmissible h

2e+07

1e+07

0
Cost And Distance Are Different

Greedy Search on Cost vs Distance

- **total nodes generated**
  - 100000
  - 50000

- **cost**
- **distance**
We’re Ignoring Useful Information

- Inadmissible estimates of cost provide better guidance.
- Search on distance is faster than search on cost.
Inadmissible estimates of cost provide better guidance. We can’t use these without sacrificing bounds.

Search on distance is faster than search on cost. Previous algorithms haven’t effectively harnessed $d$. 

Motivation
- Inadmissible estimates of cost provide better guidance. We can’t use these without sacrificing bounds.
- Search on distance is faster than search on cost. Previous algorithms haven’t effectively harnessed $d$.

Summary
- EES uses inadmissible estimates for guidance, admissible estimates for bounding takes advantage of cost and distance estimates without brittle behavior of previous approaches
Given:

\( h \) - An admissible estimate of cost to go

\( \hat{h} \) - A potentially inadmissible estimate of cost to go

\( \hat{d} \) - A potentially inadmissible estimate of distance to go

\( \hat{f}(n) = g(n) + \hat{h}(n) \)
Explicit Estimation Search

Given:

\( h \) - An admissible estimate of cost to go

\( \hat{h} \) - A potentially inadmissible estimate of cost to go

\( \hat{d} \) - A potentially inadmissible estimate of distance to go

\( \hat{f}(n) = g(n) + \hat{h}(n) \)

\( f_{min} = \) node with least \( f \)

\( best_{\hat{f}} = \) node with best estimated cost

\( best_{\hat{d}} = \) \( w \)-admissible node nearest to goal
Explicit Estimation Search

Given:

\( h \) - An admissible estimate of cost to go
\( \hat{h} \) - A potentially inadmissible estimate of cost to go
\( \hat{d} \) - A potentially inadmissible estimate of distance to go

\[
\hat{f}(n) = g(n) + \hat{h}(n)
\]

\[
f_{\text{min}} = \text{node with least } f = \arg\min_{n \in \text{open}} f(n) = g(n) + h(n)
\]

\[
\text{best } \hat{f} = \text{node with best estimated cost}
\]

\[
\text{best } \hat{d} = \text{w-admissible node nearest to goal}
\]
Explicit Estimation Search

Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go

\[
\hat{f}(n) = g(n) + \hat{h}(n)
\]

\[
\begin{align*}
  f_{\text{min}} &= \text{node with least } f \\
  &= \arg\min_{n \in \text{open}} f(n) = g(n) + h(n) \\
  \text{best } \hat{f} &= \text{node with best estimated cost} \\
  &= \arg\min_{n \in \text{open}} \hat{f}(n) = g(n) + \hat{h}(n) \\
  \text{best } \hat{d} &= \text{w-admissible node nearest to goal}
\end{align*}
\]
Explicit Estimation Search

Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go

\[
\hat{f}(n) = g(n) + \hat{h}(n)
\]

\[
\begin{align*}
    f_{\text{min}} &= \text{node with least } f \\
                 &= \arg\min_{n \in \text{open}} f(n) = g(n) + h(n) \\

    \text{best } \hat{f} &= \text{node with best estimated cost} \\
                        &= \arg\min_{n \in \text{open}} \hat{f}(n) = g(n) + \hat{h}(n) \\

    \text{best } \hat{d} &= \text{w-admissible node nearest to goal} \\
                        &= \arg\min_{n \in \text{open} \wedge \hat{f}(n) \leq w \cdot \hat{f}(\text{best } \hat{f})} \hat{d}(n)
\end{align*}
\]
Why This Expansion Order?

**Motivation**

- **EES**
  - Nodes
  - Expansion Order
  - Summary

**Results**

Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go

$$f_{\text{min}} = \text{node with least } f$$
$$\text{best} \hat{f} = \text{node with best estimated cost}$$
$$\text{best} \hat{d} = w\text{-admissible node nearest to goal}$$

$$\text{selectNode} = \begin{cases} 
\text{best} \hat{d} & \text{if it is within the bound} \\
\text{best} \hat{f} & \text{if it is within the bound, but } \text{best} \hat{d} \text{ isn't} \\
f_{\text{min}} & \text{otherwise}
\end{cases}$$
Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go

- $f_{\min} =$ node with least $f$
- $best_{\hat{f}} =$ node with best estimated cost
- $best_{\hat{d}} =$ $w$-admissible node nearest to goal

$$selectNode = \begin{cases} 
  best_{\hat{d}} & \text{if it is within the bound} \\
  best_{\hat{f}} & \text{if it is within the bound, but } best_{\hat{d}} \text{ isn't} \\
  f_{\min} & \text{otherwise}
\end{cases}$$

Of all the nodes within the bound, expand the one closest to a goal.
Why This Expansion Order?

Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go

- $f_{\text{min}}$ = node with least $f$
- $\text{best}_{\hat{f}}$ = node with best estimated cost
- $\text{best}_{\hat{d}}$ = $w$-admissible node nearest to goal

$$selectNode = \begin{cases} 
\text{best}_{\hat{d}} & \text{if it is within the bound} \\
\text{best}_{\hat{f}} & \text{if it is within the bound, but } \text{best}_{\hat{d}} \text{ isn't} \\
\text{f}_{\text{min}} & \text{otherwise}
\end{cases}$$

Ensures $\text{best}_{\hat{d}}$ is a high quality node.
Why This Expansion Order?

Given:

\( h \) - An admissible estimate of cost to go
\( \hat{h} \) - A potentially inadmissible estimate of cost to go
\( \hat{d} \) - A potentially inadmissible estimate of distance to go

\[ f_{\text{min}} = \text{node with least } f \]
\[ \text{best}_{\hat{d}} = \text{node with best estimated cost} \]
\[ \text{best}_{\hat{d}} = w\text{-admissible node nearest to goal} \]

\[
\text{selectNode} = \begin{cases} 
    \text{best}_{\hat{d}} & \text{if it is within the bound} \\
    \text{best}_{\hat{f}} & \text{if it is within the bound, but } \text{best}_{\hat{d}} \text{ isn't} \\
    f_{\text{min}} & \text{otherwise} 
\end{cases}
\]

Provides the suboptimality bounds.
Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go
- $\hat{f}(n) = g(n) + \hat{h}(n)$

$$f_{\text{min}} = \text{node with least } f$$
$$\text{best}_\hat{f} = \text{node with best estimated cost}$$
$$\text{best}_\hat{d} = \text{w-admissible node nearest to goal}$$

$$\text{selectNode} = \begin{cases} 
\text{best}_\hat{d} & \text{if it is within the bound} \\
\text{best}_\hat{f} & \text{if it is within the bound, but best}_\hat{d} \text{ isn't} \\
f_{\text{min}} & \text{otherwise}
\end{cases}$$
Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go
- $\hat{f}(n) = g(n) + \hat{h}(n)$

$$f_{\text{min}} = \text{node with least } f$$
$$\text{best } \hat{f} = \text{node with best estimated cost}$$
$$\text{best } \hat{d} = \text{w-admissible node nearest to goal}$$

$\text{selectNode} = \begin{cases} 
\text{best } \hat{d} & \text{if it is within the bound} \\
\text{if } \hat{f}(\text{best } \hat{d}) \leq w \cdot f(f_{\text{min}}) \\
\text{best } \hat{f} & \text{if it is within the bound, but best } \hat{d} \text{ isn't} \\
f_{\text{min}} & \text{otherwise}
\end{cases}$
Given:

- $h$ - An admissible estimate of cost to go
- $\hat{h}$ - A potentially inadmissible estimate of cost to go
- $\hat{d}$ - A potentially inadmissible estimate of distance to go
- $\hat{f}(n) = g(n) + \hat{h}(n)$

\[
\begin{align*}
    f_{\min} & = \text{node with least } f \\
    \text{best } \hat{f} & = \text{node with best estimated cost} \\
    \text{best } \hat{d} & = \text{w-admissible node nearest to goal}
\end{align*}
\]

\[
\text{selectNode} = \begin{cases} 
    \text{best } \hat{d} & \text{if } \hat{f}(\text{best } \hat{d}) \leq w \cdot f(f_{\min}) \\
    \text{best } \hat{f} & \text{if it is within the bound, but best } \hat{d} \text{ isn't} \\
    f_{\min} & \text{otherwise}
\end{cases}
\]
Given:

- \(h\) - An admissible estimate of cost to go
- \(\hat{h}\) - A potentially inadmissible estimate of cost to go
- \(\hat{d}\) - A potentially inadmissible estimate of distance to go
- \(\hat{f}(n) = g(n) + \hat{h}(n)\)

\[f_{\text{min}} = \text{node with least } f\]

\[\text{best}_{\hat{f}} = \text{node with best estimated cost}\]

\[\text{best}_{\hat{d}} = \text{w-admissible node nearest to goal}\]

\[
\text{selectNode} = \begin{cases} 
\text{best}_{\hat{d}} & \text{if it is within the bound} \\
\text{best}_{\hat{f}} & \text{if it is within the bound, but } \text{best}_{\hat{d}} \text{ isn't} \\
f_{\text{min}} & \text{otherwise} \\
\end{cases}
\]

\[
\text{if } \hat{f}(\text{best}_{\hat{f}}) > w \cdot f(f_{\text{min}}) \\
\land \hat{f}(\text{best}_{\hat{d}}) > w \cdot f(f_{\text{min}})
\]
- Inadmissible estimates of cost provide better guidance.
  We can’t use these without sacrificing bounds.
- We can estimate the cost and the distance of a solution.
  Algorithms that use this information perform poorly.
Inadmissible estimates of cost provide better guidance. EES can use these without sacrificing quality bounds.

We can estimate the cost and the distance of a solution. EES avoids the pitfalls of previous approaches.
Vacuums: Inadmissible Heuristics

Motivation

EES

Results

- Vacuums
- Grids
- Aggregate
- Conclusions

Vacuum World

Suboptimality

total nodes generated

wA* optimistic EES

Jordan Thayer (UNH)

Bounded Suboptimal Search – 12 / 18
<table>
<thead>
<tr>
<th>Grids</th>
<th>Motivation</th>
<th>EES</th>
<th>Results</th>
<th>Conclusions</th>
</tr>
</thead>
</table>

**Unit Four-way Grid World**

- **EES**
- **wA***
- **optimistic**

![Graph showing total nodes generated relative to A vs. suboptimality](image_url)

- Total nodes generated relative to A
- Suboptimality

Jordan Thayer (UNH)
### Performance In Aggregate: CPU Relative to EES

<table>
<thead>
<tr>
<th>Bound</th>
<th>1.5</th>
<th>1.75</th>
<th>2.0</th>
<th>3.0</th>
<th>4.0</th>
<th>5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimistic</td>
<td>1.6</td>
<td>1.5</td>
<td>1.6</td>
<td>2.1</td>
<td>2.4</td>
<td>2.1</td>
</tr>
<tr>
<td>wA*</td>
<td>4.1</td>
<td>3.4</td>
<td>2.8</td>
<td>3.7</td>
<td>3.4</td>
<td>2.4</td>
</tr>
<tr>
<td>skeptical</td>
<td>2.6</td>
<td>4.7</td>
<td>4.9</td>
<td>5.1</td>
<td>11.4</td>
<td>13</td>
</tr>
<tr>
<td>$A^*_\epsilon$</td>
<td>50</td>
<td>44</td>
<td>28</td>
<td>1.8</td>
<td>1.1</td>
<td><strong>0.6</strong></td>
</tr>
<tr>
<td>Clamped</td>
<td>8.3</td>
<td>10</td>
<td>11</td>
<td>67</td>
<td>85</td>
<td>85</td>
</tr>
<tr>
<td>AlphA*</td>
<td>120</td>
<td>140</td>
<td>180</td>
<td>280</td>
<td>300</td>
<td>310</td>
</tr>
<tr>
<td>rdwA*</td>
<td>370</td>
<td>310</td>
<td>240</td>
<td>100</td>
<td>84</td>
<td>120</td>
</tr>
<tr>
<td>$A^*_\epsilon$</td>
<td>910</td>
<td>850</td>
<td>680</td>
<td>620</td>
<td>590</td>
<td>610</td>
</tr>
</tbody>
</table>

Numbers are average slowdown per domain, averaged across eight domains: TSP (two variants), Grid Navigation (two variants), Dynamic Robot Path Planning, Vacuum Planning, Sliding Tiles Problem.
## General Performance: Nodes Generated Relative to EES

<table>
<thead>
<tr>
<th>Bound</th>
<th>1.5</th>
<th>1.75</th>
<th>2.0</th>
<th>3.0</th>
<th>4.0</th>
<th>5.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimistic</td>
<td>3.1</td>
<td>2.4</td>
<td>2.5</td>
<td>3.3</td>
<td>3.4</td>
<td>3.2</td>
</tr>
<tr>
<td>wA*</td>
<td>6.6</td>
<td>5.5</td>
<td>4.5</td>
<td>5.5</td>
<td>5.0</td>
<td>4.0</td>
</tr>
<tr>
<td>skeptical</td>
<td>3.2</td>
<td>3.0</td>
<td>2.8</td>
<td>3.8</td>
<td>11.0</td>
<td>15.0</td>
</tr>
<tr>
<td>$A^*_\epsilon$</td>
<td>58</td>
<td>44</td>
<td>17</td>
<td>1.8</td>
<td>1.1</td>
<td>0.8</td>
</tr>
<tr>
<td>Clamped</td>
<td>6.8</td>
<td>5.6</td>
<td>7.1</td>
<td>76.0</td>
<td>95.0</td>
<td>97.0</td>
</tr>
<tr>
<td>AlphA*</td>
<td>1.2</td>
<td>1.5</td>
<td>2.2</td>
<td>4.4</td>
<td>5.6</td>
<td>5.7</td>
</tr>
<tr>
<td>rdwA*</td>
<td>180</td>
<td>170</td>
<td>150</td>
<td>86.0</td>
<td>78.0</td>
<td>160</td>
</tr>
<tr>
<td>$A^*_\epsilon$</td>
<td>1500</td>
<td>1400</td>
<td>1100</td>
<td>990</td>
<td>910</td>
<td>970</td>
</tr>
</tbody>
</table>

Numbers are average increase in nodes generated per domain, averaged across eight domains: TSP (two variants), Grid Navigation (two variants), Dynamic Robot Path Planning, Vacuum Planning, Sliding Tiles Problem
## General Performance: Algorithm Rankings (CPU)

<table>
<thead>
<tr>
<th></th>
<th>1&lt;sup&gt;st&lt;/sup&gt;</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt;</th>
<th>3&lt;sup&gt;rd&lt;/sup&gt;</th>
<th>4&lt;sup&gt;th&lt;/sup&gt;</th>
<th>&gt; 4&lt;sup&gt;th&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>EES</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Optimistic</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Skeptical</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>$A_\varepsilon^*$</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>$wA^*$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>$A_\varepsilon$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>AlphA*</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Clamped</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>rdwA*</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

Rankings by CPU time consumed
### Average Performance: Algorithm Rankings (Nodes)

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>1&lt;sup&gt;st&lt;/sup&gt;</th>
<th>2&lt;sup&gt;nd&lt;/sup&gt;</th>
<th>3&lt;sup&gt;rd&lt;/sup&gt;</th>
<th>4&lt;sup&gt;th&lt;/sup&gt;</th>
<th>&gt; 4&lt;sup&gt;th&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>EES</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Optimistic</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Skeptical</td>
<td>0</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>A&lt;sub&gt;ε&lt;/sub&gt;*</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>wA&lt;sub&gt;ε&lt;/sub&gt;*</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>A&lt;sub&gt;ε&lt;/sub&gt;</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>AlphA&lt;sub&gt;ε&lt;/sub&gt;*</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>Clamped</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>rdwA&lt;sub&gt;ε&lt;/sub&gt;*</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
</tbody>
</table>

Rankings by nodes generated
■ We can finally use inadmissible heuristics.
■ We can benefit from using cost and distance information.
■ EES provides robust behavior on a wide range of benchmarks.
  state of the art performance in several domains.
Additional Slides

- Bounds
- Robots
- Bounding
Assume:
\[ \hat{f}(n) \geq f(n) \text{ and } \hat{h}(\text{goal}) = 0 \]
\[ f(n) = \hat{f}(n) = g(n) \]

\[
\text{selectNode} = \begin{cases} 
  \text{best}_{\hat{d}} & \text{if } \hat{f}(\text{best}_{\hat{d}}) \leq w \cdot f(f_{\text{min}}) \\
  \text{best}_{\hat{f}} & \text{if } \hat{f}(\text{best}_{\hat{f}}) \leq w \cdot f(f_{\text{min}}) \\
  f_{\text{min}} & \text{otherwise}
\end{cases}
\]
Proof of Bounded Suboptimality

Assume:
\[ \hat{f}(n) \geq f(n) \] and \[ \hat{h}(goal) = 0 \]
\[ f(n) = \hat{f}(n) = g(n) \]

\[ \text{selectNode} = \begin{cases} 
\text{best}_{\hat{d}} & \text{if } \hat{f}(\text{best}_{\hat{d}}) \leq w \cdot f(f_{\text{min}}) \\
\text{best}_{\hat{f}} & \text{if } \hat{f}(\text{best}_{\hat{f}}) \leq w \cdot f(f_{\text{min}}) \\
f_{\text{min}} & \text{otherwise}
\end{cases} \]

\[ w \cdot f(\text{opt}) \geq w \cdot f(f_{\text{min}}) \]
\[ w \cdot f(f_{\text{min}}) \geq \hat{d}(\text{best}_{\hat{d}}) \]
\[ \hat{f}(\text{best}_{\hat{d}}) \geq f(\text{best}_{\hat{d}}) \]
\[ f(\text{best}_{\hat{d}}) \geq g(\text{best}_{\hat{d}}) \]
Proof of Bounded Suboptimality

Assume:
\[ \hat{f}(n) \geq f(n) \text{ and } \hat{h}(\text{goal}) = 0 \]
\[ f(n) = \hat{f}(n) = g(n) \]

selectNode = \[
\begin{cases} 
  \text{best}_{\hat{d}} & \text{if } \hat{f}(\text{best}_{\hat{d}}) \leq w \cdot f(f_{\text{min}}) \\
  \text{best}_{\hat{f}} & \text{if } \hat{f}(\text{best}_{\hat{f}}) \leq w \cdot f(f_{\text{min}}) \\
  f_{\text{min}} & \text{otherwise}
\end{cases}
\]

\[ w \cdot f(\text{opt}) \geq w \cdot f(f_{\text{min}}) \]
\[ w \cdot f(f_{\text{min}}) \geq \hat{f}(\text{best}_{\hat{f}}) \]
\[ \hat{f}(\text{best}_{\hat{f}}) \geq f(\text{best}_{\hat{f}}) \]
\[ f(\text{best}_{\hat{f}}) \geq g(\text{best}_{\hat{f}}) \]
Assume:
\[ \hat{f}(n) \geq f(n) \text{ and } \hat{h}(\text{goal}) = 0 \]
\[ f(n) = \hat{f}(n) = g(n) \]

\[
\text{selectNode} = \begin{cases} 
\text{best}_d & \text{if } \hat{f}(\text{best}_d) \leq w \cdot f(f_{\text{min}}) \\
\text{best}_f & \text{if } \hat{f}(\text{best}_f) \leq w \cdot f(f_{\text{min}}) \\
f_{\text{min}} & \text{otherwise}
\end{cases}
\]

\[ w \cdot f(\text{opt}) \geq w \cdot f(f_{\text{min}}) \]
Robot Navigation: Inadmissible Heuristics

Motivation

EES

Results

Additional Slides

- Bounds
- Robots
- Bounding

Dynamic Robot Navigation

Suboptimality

$wA^*$ optimistic EES ah

Total nodes generated vs. suboptimality

$8 \times 10^7$ $7 \times 10^7$ $6 \times 10^7$ $5 \times 10^7$ $4 \times 10^7$ $3 \times 10^7$ $2 \times 10^7$ $1 \times 10^7$ $0$

1 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2

Suboptimality

Jordan Thayer (UNH)
Strict vs. Loose Approaches to Quality Bounds

**Motivation**

- **EES**

**Results**

- **Bounds**
- **Robots**
- **Bounding**

Loose: Optimistic Search

- Run weighted $A^*$ with weight $(\text{bound} - 1) \cdot 2 + 1$
- Expand node with lowest $f$ value after a solution is found.
  
  Continue until $w \cdot f_{\text{min}} > f(sol)$
  
  This 'clean up' guarantees solution quality.

Strict: EES

\[
\text{selectNode} = \begin{cases} 
\text{best}_{\hat{d}} & \text{if } \hat{f}(\text{best}_{\hat{d}}) \leq w \cdot f(f_{\text{min}}) \\
\text{best}_{\hat{f}} & \text{if } \hat{f}(\text{best}_{\hat{f}}) \leq w \cdot f(f_{\text{min}}) \\
f_{\text{min}} & \text{otherwise}
\end{cases}
\]