Improved Heuristics for Optimal Pathfinding on Game Maps

Samuel Bader

Universität Basel

October 29, 2015
1. Introduction
2. The Decomposition Algorithm
3. The Dead-End Heuristic
4. The Gateway Heuristic
5. Evaluation
We have a map with passable and impassable tiles
We want to find a shortest path between two points
One commonly used algorithm is A*
A* uses the cost function $f(n) = g(n) + h(n)$

- $g(n)$ is the cost from the start to $n$
- $h(n)$, the so called heuristic, is the estimated cost from $n$ to the goal

A* produces an optimal path if the heuristic is admissible

- i.e. if the cost is never overestimated

A commonly used heuristic for game maps is the octile distance
A* with octile distance visits a lot of nodes which are far away from the final path:
The Decomposition Algorithm

- divides map into smaller areas
- uses flood-filling techniques
- needs no additional information
The Decomposition Algorithm

- starts at the top left free tile
- expands the area to the right
  - stops at a non free tile
- repeat for the next line
- stop if walls are not convex
The end result

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
Overview

1. Introduction
2. The Decomposition Algorithm
3. The Dead-End Heuristic
4. The Gateway Heuristic
5. Evaluation
The Dead-End Heuristic

Idea: Ignore dead ends, i.e. areas where the only exit is the entrance

Preprocessing Phase:
- decompose map with the decomposition algorithm
- construct a graph with:
  - areas as nodes
  - connections between areas as edges
- store the graph with the map
Example

Improved Heuristics for Optimal Pathfinding on Game Maps;
Yngvi Björnsson and Kári Halldórsson
Runtime Phase

- search the graph for all possible paths between two areas
  - use simple depth-first search
  - mark used edges to avoid loops
- all areas not on any of these paths are irrelevant
- search the map with A*
  - heuristic returns infinity if a tile is in an irrelevant area
  - use a simple lookup as each tile knows its area
Example

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
Example

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
Overview

1. Introduction
2. The Decomposition Algorithm
3. The Dead-End Heuristic
4. The Gateway Heuristic
5. Evaluation
The Gateway Heuristic

Idea: Pre-calculate the distances between entrances/exits of areas

Preprocessing Phase:
- decompose map with the decomposition algorithm
  - each boundary between two areas is a gateway
- calculate the distances between all gateways
  - for each gateway combination 4 distances are calculated
  - store the distances with the map
Why 4 distances?

[Diagram with two elongated shapes connected by a line]
Use regular A* with the following heuristic:

\[ h^G(n, g) = \min_{i,j}[h^l(n, G_i) + H(G_i, G_j) + h^l(G_j, g)] \]

where

- \( h^l(n, G_i) \) is the distance from tile \( n \) to gateway \( G_i \)
- \( H(G_i, G_j) \) is the pre-calculated distance between two gateways
Example

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
Example

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
## Evaluation

<table>
<thead>
<tr>
<th></th>
<th>Demo map</th>
<th>Octile</th>
<th>Dead-end</th>
<th>Gateway</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>all</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>path cost</td>
<td>7430</td>
<td>7430</td>
<td>7430</td>
<td>7430</td>
</tr>
<tr>
<td>estimate</td>
<td>3940</td>
<td>3940</td>
<td>7241</td>
<td></td>
</tr>
<tr>
<td>nodes</td>
<td>955</td>
<td>579</td>
<td>230</td>
<td></td>
</tr>
<tr>
<td>times (ms.)</td>
<td>18.6</td>
<td>14.7</td>
<td>13.2</td>
<td></td>
</tr>
<tr>
<td><strong>top 10%</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>path cost</td>
<td>14373</td>
<td>14373</td>
<td>14373</td>
<td></td>
</tr>
<tr>
<td>estimate</td>
<td>6605</td>
<td>6605</td>
<td>14179</td>
<td></td>
</tr>
<tr>
<td>nodes</td>
<td>2397</td>
<td>1352</td>
<td>487</td>
<td></td>
</tr>
<tr>
<td>times (ms.)</td>
<td>42.9</td>
<td>30.4</td>
<td>28.0</td>
<td></td>
</tr>
</tbody>
</table>
Game Map Example

Improved Heuristics for Optimal Pathfinding on Game Maps; Yngvi Björnsson and Kári Halldórsson
### Evaluation

<table>
<thead>
<tr>
<th></th>
<th>Game maps</th>
<th>Octile</th>
<th>Dead-end</th>
<th>Gateway</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>all</strong></td>
<td>path cost</td>
<td>10339</td>
<td>10339</td>
<td>10339</td>
</tr>
<tr>
<td></td>
<td>estimate</td>
<td>7788</td>
<td>7788</td>
<td>9884</td>
</tr>
<tr>
<td></td>
<td>nodes</td>
<td>1231</td>
<td>1120</td>
<td>723</td>
</tr>
<tr>
<td></td>
<td>times (ms.)</td>
<td>27.3</td>
<td>24.6</td>
<td>22.6</td>
</tr>
<tr>
<td><strong>top 10%</strong></td>
<td>path cost</td>
<td>20468</td>
<td>20468</td>
<td>20468</td>
</tr>
<tr>
<td></td>
<td>estimate</td>
<td>13290</td>
<td>13290</td>
<td>19731</td>
</tr>
<tr>
<td></td>
<td>nodes</td>
<td>3701</td>
<td>3370</td>
<td>2313</td>
</tr>
<tr>
<td></td>
<td>times (ms.)</td>
<td>69.2</td>
<td>60.7</td>
<td>54.5</td>
</tr>
<tr>
<td><strong>Large map</strong></td>
<td>Octile</td>
<td>30463</td>
<td>30463</td>
<td>30463</td>
</tr>
<tr>
<td></td>
<td>Dead-end</td>
<td>30463</td>
<td>30463</td>
<td>30002</td>
</tr>
<tr>
<td></td>
<td>Gateway</td>
<td>30463</td>
<td>30463</td>
<td>2361</td>
</tr>
<tr>
<td></td>
<td>path cost</td>
<td>30463</td>
<td>30463</td>
<td>30463</td>
</tr>
<tr>
<td></td>
<td>estimate</td>
<td>17201</td>
<td>17201</td>
<td>30002</td>
</tr>
<tr>
<td></td>
<td>nodes</td>
<td>5961</td>
<td>4536</td>
<td>2361</td>
</tr>
<tr>
<td></td>
<td>times (ms.)</td>
<td>110.1</td>
<td>84.0</td>
<td>71.3</td>
</tr>
</tbody>
</table>
Questions?