Assignment 1: A* on Terrain Maps

Kevin Wang and Philip Chen

February 15, 2016

Part 1: Creating Heuristics

Exponential Cost Function

Terminology

D	is the X-Y Chebyshev distance between Point $p1$ and Point $p2$
$G(N)$	is the cost function for some node N
$H(N)$	is the heuristic function for some node N
S	is the slope or average step each edge must take to reach Point $p2$ from Point $p1$
ΔZ	is the absolute Z height difference between Point $p1$ and Point $p2$
Z_1	is the height of Point $p1$
Z_2	is the height of Point $p2$

Cost Function

$G(n) = e^{Z_2 - Z1}$

Heuristic Function

Relax the rule that hikers can only traverse the mountain's surface. With this addition, the hiker can essentially travel a straight and sloped line from his current point to the goal state. The result is an underestimate of the actual cost.

$Z_1 == Z_2 :$

$H(N) = D$

$Z_1 > Z_2 :$

$H(N) = D \times e^{S} | S \geq 1$

$H(N) = \Delta Z \times e^{1} + D - \Delta Z | S < 1$

$Z_1 < Z_2 :$

$H(N) = D \times e^{-S} | S \geq 1$

$H(N) = \Delta Z \times e^{-1} + D - \Delta Z | S < 1$

Proof of Consistency and Admissibility

Consistent heuristics obey the triangle inequality:

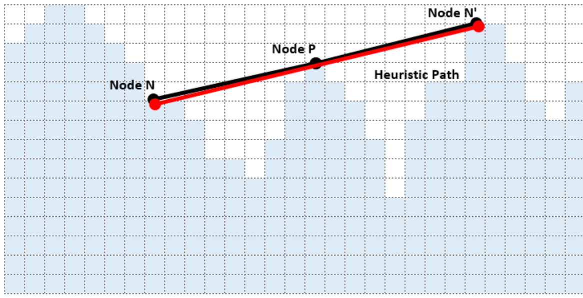
$$H(N) \leq G(N, P) + H(P)$$

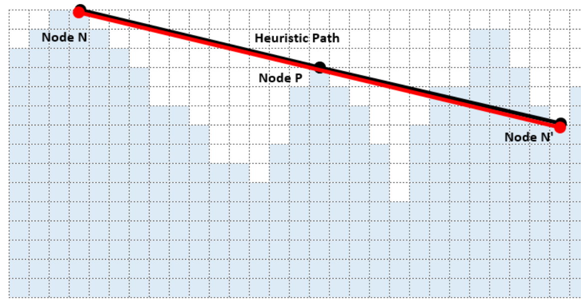
where P is a neighbor of N. The relation can be expanded to any Point p between N and the goal state G. Let n, p, and g be the heights of N, P, and G respectively. There are four cases to consider.

$n < p < g$ or $g < p < n$:

The above inequality is stating that: when trying to reach the goal state, the current node's adjacent neighbors are closer in height to the goal state.

Our heuristic function uses the exponential of the average slope to estimate the cost. It will consistently underestimate the cost, and $Cost(N, P)$ will be greater than the estimated heuristic $H(N)$ always. Thus, our heuristic function obeys the consistency relation.



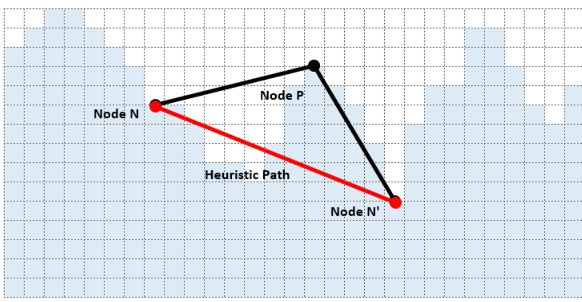


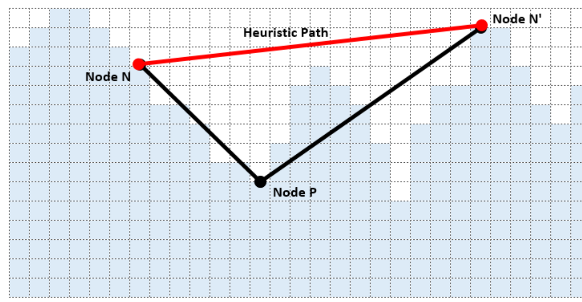
$g < n < p$ or $p < n < g$:

The above inequality is stating that: when trying to reach the goal state, the current node's adjacent neighbors are further in height to the goal state.

Our heuristic function handles for this condition because it factors in the greater distance by increasing the average step S value. The new average step goes into the exponential of e^N as N, resulting in a greater estimated cost for P that will still be less than the $Cost(N, P) + H(P)$.

In these examples, the blue shaded cells are not the actual cost, but just to help with visualization. The actual path is outlined by the bolded lines, N being the current node, N' being the end point, and P being an intermediary node.





Examples and Worse Case

Refer above for examples.

Admissible Exponential Heuristic Code

Listing 1: Code for Exponential Heuristic Function

```
// Exponential Heuristic Function
private double getHeuristic(final TerrainMap map, final Point pt1, final Point pt2)
{
   double z1 = map.getTile(pt1);
   double z2 = map.getTile(pt2);
   double dist = Math.max(Math.abs(pt1.x-pt2.x), Math.abs(pt1.y-pt2.y));
   double zDist = Math.abs(z2 - z1);
   double avgStep = Math.floor(zDist/dist);
   if(z2 == z1) {
      return dist;
   } else if (z2 > z1) {
      if (avgStep >= 1) { // steep slope
         return dist * Math.exp(avgStep);
      } else {
         return zDist * Math.exp(1) + (dist-zDist);
      }
   } else { // z1 > z2
      if (avgStep >= 1) { // steep slope
         return dist * Math.exp(-avgStep);
      } else {
         return zDist * Math.exp(-1) + (dist-zDist);
      }
   }
}
```

Division Cost Function

Terminology

C = Chebyshev distance
N = Node height
E = Endpoint height
m = min(N, E)

Cost Function

$G(n) = \frac{Z_2}{Z_1+1}$

Heuristic Function

Different than the other cost function, the value of the cost function is based on the height values, not just the difference in heights. The idea is to relax the "rule" that one can only travel above land. This heuristic calculates the cost if one walked in a straight line, with each step the same height as the previous, until the end point. Here is the heuristic:

$$h(n) = \frac{m}{m+1} \times C$$

Proof of Consistency and Admissibility

A heuristic is consistent if and only if the following is true:

$$h(N) \leq Cost(N, P) + h(P)$$

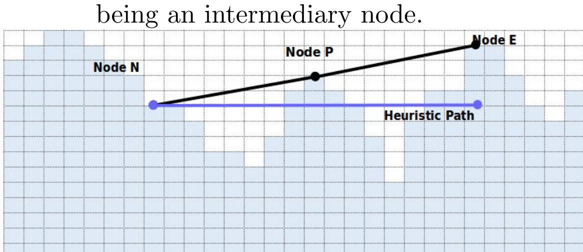
P is a neighbor of N, and in the explanations, can even be expanded to any point between N and the endpoint. There are four cases for this, where n, p, and e are the heights of nodes N, P, and E respectively: $n < p < e$, $p < n < e$, $e < p < n$, $e < n < p$.

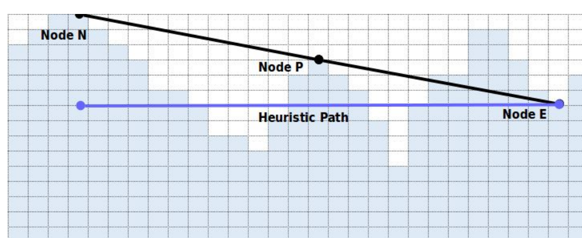
n <p <e and e <p <n:

In these cases, the height used by the heuristic is lower than or equal to all other heights in the path. By the cost AND the heuristic equation, this flat, low height path will also be lower than the actual path cost.

In the examples below, the heuristics use path costs clearly lower than the actual costs. They also satisfies the consistency equation $h(N) \leq Cost(N, P) + h(P)$.

In these examples, the blue shaded cells are not the actual cost, but just to help with visualization. The actual path is outlined by the bolded lines, N being the current node, E being the end point, and P being an intermediary node.

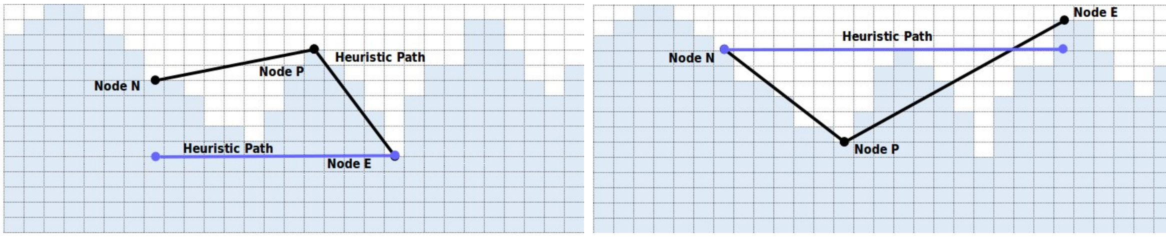




e <n <p and p <n <e:

In these more likely cases, the path to the end is not monotonic, rather it contains a change in direction, or mini hills or valleys. In this case, it may seem that because there is an extra downhill segment, the path could be potentially decreased. But all of the extra uphill that needs to be made up causes other segments to be more steep, thus having a greater cost.

In the examples below, although the paths dip below the heuristic's heights, the added steepness to the other segments more than make up for the slightly lowered costs. In fact, traveling along hills and valleys are more costly than traveling along a consistently sloped path. Although it is not mathematically proven for all cases in this report, the examples, and many like it, satisfies the consistency equation $h(N) \leq Cost(N, P) + h(P)$.



In these examples, the blue shaded cells are not the actual cost, but just to help with visualization. The actual path is outlined by the bolded lines, N being the current node, E being the end point, and P being an intermediary node.

Examples and Worst Case

Refer above for examples.

Admissible Division Heuristic Code

Listing 2: Code for Division Heuristic Function

```java
private double getHeuristic(final TerrainMap map, final Point pt1, final Point pt2)
{
   double z1 = map.getTile(pt1);
   double z2 = map.getTile(pt2);
   double dist = Math.max(Math.abs(pt1.x-pt2.x), Math.abs(pt1.y-pt2.y));
   double min = Math.min(z1, z2);
   return dist*min/(min+1);
}
```

Part 2: Implementing the Heuristics and A* Algorithm

Refer to the submission for AStarExp-[student-ids].java and AStarDiv-[student-ids].java

Part 3: Trying out your Code on a Small Problem

AStarExp

	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5
Path Cost	533.4482191461119	549.5036346739352	510.97825243663607	560.6570436319696	479.5879215923168
Uncovered	71644	81810	74001	66382	67837
Time Taken	1104	1213	1070	1060	996

AStarDiv

	Seed 1	Seed 2	Seed 3	Seed 4	Seed 5
Path Cost	198.59165501141644	198.56141550095256	198.4864317411443	198.70066424826052	198.25499446810397
Uncovered	1013	1004	1005	1006	1020
Time Taken	58	58	54	52	58

Part 4: Climbing Mount Saint Helens During The Eruption

To fill in later.