ECS170 Program1 Part1

Alice Chen and Tyler Welsh

February 2016

Part 1 a):

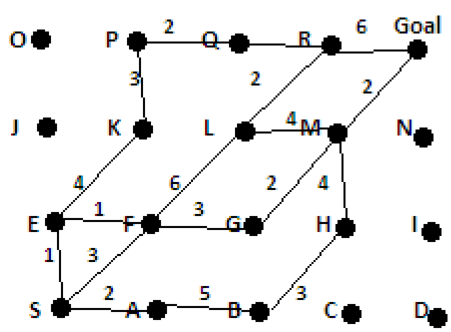
For division cost function:

Use Manhattan Distance with the assumption that each step is 0.9 cost.

$h(n) = (EndPoint.x - CurrentPoint.x) + (EndPoint.y - CurrentPoint.y)$

Examples for Division Cost:

Given a 5x4 grid with the starting point at (0,0) and the end node at (5,5), the following table shows the heuristic costs at each node:



Expanded Node	Open	$f(n)$	$g(n)$	$h(n)$
S	A(7.4), F(7.5), E(6.4)	7.4	2	5.4
		7.5	3	4.5
		6.4	1	5.4
E	K(7.6), F(5.5)	7.6	4	3.6
		5.5	1	4.5
F	L(8.7), G(6.6)	8.7	6	2.7
		6.6	3	3.6
G	M(3.8)	3.8	2	1.8
M	L(6.7), Goal(2.8)	6.7	4	2.7
		2.8	1	1.8
Goal	-	-	-	-

The heuristic values after each expanded node decreases as it moves towards the goal. This corresponds with the definition of a consistent and admissible heuristic.

If the starting node is M where we would only need one move to reach the goal state:

Expanded Node	Open	$f(n)$	$g(n)$	$h(n)$
M	L(6.7), Goal(3.8)	6.7	4	2.7
		3.8	2	1.8
Goal	-	-	-	-

The heuristic still underestimates the true cost between an arbitrary start node and the goal node.

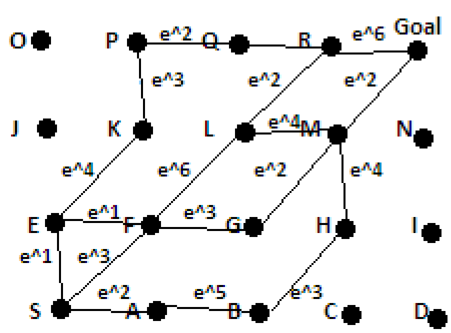
Worse case:
The worse case would be for the algorithm to expand as many nodes as Dijkstra's. But using the heuristic, nodes like P, Q, and H would not be considered for expansion and thus would increase A*'s efficiency.

For exponential cost function:

$h(n) = e^{(EndPoint - getTile(EndPoint - minPoint))}$

where EndPoint is the value after calling getTile and minPoint is the least cost adjacent point to the current node.

Examples for Exponential Cost:
Given same 5x4 grid for Division Cost but weights are now calculated using e^x with end point (4,3):



Expanded Node	Open	$f(n)$	$g(n)$	$h(n)$
S	A(22.5), F(35.24), E(17.87)	22.5	e^2	$e^(e)$
		35.24	e^3	e^e
		17.87	e	e^e
E	K(69.75), F(17.87)	69.75	e^4	e^e
		17.87	e	e^e
F	L(528491714), G(528491331)	528491714	e^6	e^{e^3}
		528491331	e^3	e^{e^3}
G	M(14.778)	14.778	e^2	e^2
M	L(61.98), Goal(2.8)	61.98	e^4	e^2
		10.107	e	e^2
Goal	-	-	-	-

The heuristic value also continuously goes down and underestimates the true cost to the goal node

Worse case:
The worse case would be for the algorithm to expand as many nodes as Dijkstra's. But using the heuristic, nodes like P, Q, and H would not be considered for expansion and thus would increase A*'s efficiency.