CS 271 - Introduction to Artificial Intelligence **Fall 2016**

HomeWork 2

Name: *Liangjian Chen*
ID: *#52006933* *October 13, 2016*

Problem 1

Step 1 Zerind

Name	g	h	f
Arad	75	366	441
Oradea	71	380	451

Step 2 Arad

Name	g	h	f
Sibiu	215	366	441
Timisoara	193	329	522
Oradea	71	380	451

Step 3 Sibiu

Name	g	h	f
Fagaras	214	176	390
Rimnicu Vilcea	295	193	488
Timisoara	193	329	522
Oradea	71	380	451

Step 4 Fagaras

Name	g	h	f
Bucharest	425	0	425
Rimnicu Vilcea	295	193	488
Timisoara	193	329	522
Oradea	71	380	451

Step 5 Find Bucharest, the distance is 25.

Problem 2 Assume u expand v and s is the start point, t is the destination.

(a)

$$f(v) = c(s,v) + h(v) = c(s,u) + c(u,v) + h(v) \geq c(s,u) + h(u) = f(u)$$

(b)

$$\begin{aligned}
&h_1(u) \leq c(u,v) + h_1(v) \\
&h_2(u) \leq c(u,v) + h_2(v) \\
&h = \max\{h_1(u), h_2(u)\} \leq \max\{c(u,v) + h_1(v), c(u,v) + h_2(v)\} \\
&= c(u,v) + \max\{h_1(v), h_2(v)\} = c(u,v) + h(v)
\end{aligned}$$

(c) prove it is admissible is proving the optimal path would always be found. Were it not he case, there would be another node n' in the frontier on the optimal path from s to t. Let's say path 1 is the path find in heuristic search and path 2 is the optimal path go through n' and d_1 ,d_2 are their corresponding length. Thus,
$f(u) = c(s, n') + h(n') \leq c(s, n') + c(n', t) = d_1 \leq d_2 = f(t)$
Then, in the last step, the frontier should pop the n' rather than t. So it contradict with the assumption. Thus the statement holds.

(d) consider, after we pick u out from the frontier, node v link to u again. From(a), we can obtain that $g(u) + h(u) \leq g(v) + h(v)$ and $h(u) \geq h(v)$. The new weight is $f_{new}(u) = g(v) + c(v, s) + h(u) \geq g(u) + h(u)$. Thus, it would never update the $f(u)$ and would never push u into frontier again.

(e) Intuitively, $h(u)$ is an underestimate of $c(u, t)$, so if $h_1(u) \geq h_2(u)$, $h_1(u)$ is more accurate than $h_2(u)$. Thus, A_1^* would expand **equal or less** than A_2^*.

For formal proof, let's consider picking node u in the frontier by h_2, but there is another node u' whose f-value is smaller than u by heuristic function h_1 which leads u. Then A_1^* will choose u'. If it leads to an optimal path without u, A_1^* does not need to expand u anymore.

However if no case mentioned above happened, the number of expand node should be same.

Problem 3 Solution:

Completeness: For any w, algorithm will expand all the node and find the answer.

Optimal:

$$
\begin{aligned}
&f(n) = (2 - w)g(n) + wh(n) \\
&f(n) = (2 - w)(g(n) + \frac{w}{2 - w}h(n)) \\
&\text{the coefficient of } h(n) \text{ should less or equal 1} \\
&\frac{w}{2 - w} \leq 1 \\
&w \leq 1
\end{aligned}
$$

$w = 0$: Uniform-cost search

$w = 1$: Best-first search

$w = 2$: Greedy best search

Problem 4 Yes, we can use this result to cut some brunch.

If a node u's estimated distance $f(u)$ is larger than f_U, we can just discard it , rather than putting it into frontier.

Problem 5 For example, a graph $G(V, E)$, $V = \{s, t, a, b\}$, $E = \{e(s, a, 0), e(s, b, 0), e(a, t, 100), e(b, t, 101)\}$. Heuristic function $h(a) = 90, h(b) = 10$. So we would expand b before a and get goal node t, at this time $f(t) = 101$, but the optimal path is $s \to a \to t$ which just cost 100. So terminating as soon as goal node find is not correct.