CS221 Fall 2018 Homework [reconstruct]

SUNet ID: [chiwang]
Name: [Chi Wang]

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

Problem 1

1.a:

Answer:
Assume we have this input string: "Adahasagoldfishtailisreadveryspecial"
with below cost functions:
c(Ada,has) = 1.0
c(has,a) = 1.0
c(a,gold) = 1.0
c(a,goldfish) = 10.0
c(goldfish, tail)= 2.0
c(gold,fishtail) = 50.0
c(gold,fish) = 100.0
c(fish,tail) = 1.0
c(tail,is) = 1.0
c(fishtail,is) = 1.0
c(red, very)= 1.0
c(very, special)=1.0

By greedy approach, result is : "Ada has a gold fishtail is red very special", its total cost is 56, but the optimal result sentence would be "Ada has a goldfish tail is red very special", its total cost is 18.

Problem 2

2.a:

Answer:
Assume we have following input string "wrt th prcss s thr smpl stps"
with below possibleFills and bigram cost function:
possibleFills = {

wrt : [write]
th : [the]
prcss : [process]
s : [as] }
thr : [their, three]
smpl : [sample, simple]
stps : [steps]

cost function:
c(write, the) = 1.0
c(the, process) = 1.0
c(process, as) = 1.0
c(as three) = 10.0
c(as their) = 5.0
c(three sample) = 1.0
c(their simple) = 15.0
c(simple steps) = 1.0
c(sample steps) = 1.0

By greedy approach, the result would be "write the process as their simple steps" with total cost 24.0, but the optimal result should be "write the process as three sample steps", its total cost is 15.0.

Problem 3

3.a:

Answer:
state = (last built word, index in input sequence as begin index for next word)
action = word built by inserting vowels and space from select range of chars in the input sequence.
cost = bigram cost of word built in current state and word that's going to be built in next state.
initial state = ('-BEGIN-', 0)
end test condition = index of input sequence for next word == input sequence length

3.c:

Answer:
state = (index in input sequence as begin index for next word)
action = word built by inserting vowels and space from select range of chars in the input sequence.

cost = minimum bigram cost between words in dictionary and word for next state (built after insert vowel and space).
initial state = (0)
end test condition = index of input sequence as begin index for next word == input sequence length

Let's define $u_b(w)$ function for $Cost_{rel}(s, a)$ as $u_b(w) = min_{x \in dict}\{b(x, w)\}$ where x is arbitrary word in training dictionary. Since u_b is minimum bigram cost between every possible word in dictionary and the given word w, we could see $Cost_{rel}(s, a) \leq Cost(s, a)$, so $h(s)$ is constant.

3.d:

Answer:
To answer questions here, let's define couple functions below:
f(n): estimated total cost of path through n to end state.
g(n): a path cost to n from a start state
h(n): a heuristic estimate of cost from n to end state

Is UCS a special case of A*? (**YES**)

A* search: f(n) = g(n) + h(n)
Uniform-cost search: f(n) = g(n)

Seen from above definitions, for h(n) = 0, uniform cost search will produce the same result as A* search, so we could see UCS is a special case of A*.

Is BFS a special case of UCS? (**YES**)
When all step costs are equal (and let's assume cost equal to 1), g(n) is just a multiple of depth n. Thus, breadth-first search and uniform-cost search would behave the same in this case f(n) = g(n) = 1*(depth of n)