CSC 520 Homework Assignment 2

Parth Satra - 200062999

October 1, 2014

1. (50 Points) This question concerns route-finding, with comparison of several search algorithms. This time, we're in the U.S. The solution consists of the series of cities a network packet must pass through, each city connected to one or more others by network links of the indicated length. There are no other network links.

   Now perform some experiments.

   (a) (10 points) Experiment with executing your implementation of A* to find various paths, until you understand the meaning of the output. Are there any pairs of cities (A,B) for which the algorithm finds a different path from B to A than from A to B? Are there any pairs of cities (A,B) for which the algorithm expands a different total number of nodes from B to A than from A to B? Output in such cases should consist of the following:

   Answer

   A* always finds the same path from A to B and B to A. This is because A* takes into account both the actual distance and heuristics to calculate the best possible path. So if while going from A to B if C is the city in best possible path then from B to A also C will be the present in the best possible path. But the expanded nodes may vary since A* will now explore nodes in reverse direction and depending on the geography different nodes will be expanded. Example for both the cases is as follows:

   Sacramento to Portland: A*

   i. A comma separated list of expanded nodes (the closed list) : sacramento, reno, stockton, pointReyes, modesto, sanFrancisco, oakland, redding, medford, sanJose, eugene, salem
   ii. The number of nodes expanded : 12
   iii. A comma-separated list of nodes in the solution path : sacramento, pointReyes, redding, medford, eugene, salem, portland
   iv. The number of nodes in the path : 7
   v. The total distance from A to B in the solution path. 755

   Portland to Sacramento: A*

   i. A comma separated list of expanded nodes (the closed list) : portland, salem, eugene, medford, redding, pointReyes
   ii. The number of nodes expanded : 6
   iii. A comma-separated list of nodes in the solution path : portland, salem, eugene, medford, redding, pointReyes, sacramento
   iv. The number of nodes in the path : 7
   v. The total distance from A to B in the solution path : 755

   (b) (10 Points) Change your code so as to implement greedy search, as discussed in the web notes.

   Answer

   Code present in SeachUSA.java and searchType is greedy.

(c) (10 Points) Do enough exploration to find at least one path that is longer using greedy search than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.

Answer

The below examples shows both the instances. A* can be seen to have shorter path and also expanded less number of nodes than Greedy.

Los Angeles to Sacramento: A*

i. A comma separated list of expanded nodes (the closed list) : losAngeles, bakersfield, fresno, modesto, stockton
ii. The number of nodes expanded : 5
iii. A comma-separated list of nodes in the solution path : losAngeles, bakersfield, fresno, modesto, stockton, sacramento
iv. The number of nodes in the path : 6
v. The total distance from A to B in the solution path : 408

Los Angeles to Sacramento: Greedy

i. A comma separated list of expanded nodes (the closed list) : losAngeles, sanLuisObispo, salinas, sanJose, oakland, sanFrancisco
ii. The number of nodes expanded : 6
iii. A comma-separated list of nodes in the solution path : losAngeles, sanLuisObispo, salinas, sanJose, oakland, sanFrancisco, sacramento
iv. The number of nodes in the path : 7
v. The total distance from A to B in the solution path : 495

(d) (10 Points) Change your code so as to implement uniform cost search, as discussed in the web notes.

Answer

The code is present in SearchUSA.java file and the searchType for Uniform Cost search is 'uniform'.

(e) (10 Points) Do enough exploration to find at least one path that is longer using uniform cost than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.

Answer

The below example shows that the uniform cost expands more nodes than A* does as A* is guided by a heuristic function in the right direction. But eventually Uniform cost also finds the same solution path as that of the A* since it also explores the shortest path.

Los Angeles to Sacramento: A*

i. A comma separated list of expanded nodes (the closed list) : losAngeles, bakersfield, fresno, modesto, stockton
ii. The number of nodes expanded : 5
iii. A comma-separated list of nodes in the solution path : losAngeles, bakersfield, fresno, modesto, stockton, sacramento
iv. The number of nodes in the path : 6
v. The total distance from A to B in the solution path : 408

Los Angeles to Sacramento: Uniform Cost Search

i. A comma separated list of expanded nodes (the closed list) : losAngeles, bakersfield, sanDiego, sanLuisObispo, fresno, lasVegas, yuma, salinas, modesto, sanJose, stockton, oakland, sanFrancisco
ii. The number of nodes expanded : 13
iii. A comma-separated list of nodes in the solution path : losAngeles,bakersfield,fresno,modesto,stockton,sacramento
iv. The number of nodes in the path : 6
v. The total distance from A to B in the solution path : 408

(f) As part of your answer, compare the solution paths and explain what happened, especially any weird behavior you might detect.

Answer

In all of the above examples we see different paths or nodes expanded as there are multiple maths to the solution. The main differences in all the three discussed algorithms is

Uniform cost search tries to explore all the shortest path without any additional knowledge about the state space. Hence it explores many short paths before exploring the optimal path which might not be the shortest from the source. Hence uniform cost might end up expanding lots of nodes in search of optimal path.

Greedy search tries to find the state closest to the goal without considering the cost from the source which might turn out to be expensive. Hence greedy search doesnt always give the optimal solution.

A* search tries to search for the optimal cost using a heuristic function which provides it a sense of direction to the goal state. Having said that A* search can also explore states which are not on the optimal path.

2. (20 Points) Tic-tac-toe (also known as Noughts and Crosses) is a two-person, zero-sum game, in which player X and player O alternate placing their symbols in one of the blank spaces in a 3-by-3 grid.

The first player to place three of his symbols in a row – horizontally, vertically, or diagonally – wins.

(a) Beginning from the position specified with X's turn to move, construct by hand the game-tree for the rest of the game. Assume the search horizon is the end of the game on all branches. Hint: You can save a lot of work by taking advantage of symmetry

Answer

The answer to all sub-questions is available in the $min\_max.svg$ or $min\_max.jpg$. Please open the svg file in Google Chrome.

(b) (5 points) Suppose the static evaluation function scores a win for O as +1, a draw as 0, and a loss for O as -1. At each level of your sketch of the game tree, indicate the value of each node based on its children. Indicate the best next move for X.

(c) (5 points) Now use - pruning. At each level of your sketch of the game tree, indicate the value of the nodes and indicate any nodes that would be pruned. Explicitly indicate the final value of the - interval is for each node. Indicate the best next move for X.

3. (12 points) Use Propositional Logic to determine whether or not the following set of requirements is logically consistent. In other words, represent the following sentences in Propositional Logic, convert to Conjunctive Normal Form, and run Resolution until a contradiction is derived, or else show a model of all the expressions showing that no contradiction exists.

   The system is in multiuser state if and only if it is operating normally. If the system is operating normally, the kernel is functioning. Either the kernel is not functioning or the system is in interrupt mode. If the system is not in multiuser state, then it is in interrupt mode. The system is not in interrupt mode. Use the following lexicon: Propositional symbols:

   m – The system is in multiuser state.

   n – The system is operating normally.

   k – The kernel is functioning.

   i – The system is in interrupt mode.

   Answer

   The system is in multiuser state if and only if it is operating normally : $m \leftrightarrow n$

   If the system is operating normally, the kernel is functioning: $n \rightarrow k$

   Either the kernel is not functioning or the system is in interrupt mode: $\neg k \vee i$

   If the system is not in multiuser state, then it is in interrupt mode : $\neg m \rightarrow i$

   The system is not in interrupt mode : $\neg i$

   Conversion to CNF and applying Resolution

   (a) $\neg m \vee n$

   (b) $\neg n \vee m$

   (c) $\neg n \vee k$

   (d) $\neg k \vee i$

   (e) $m \vee i$

   (f) $\neg i$

   (g) $\neg k$ (From : f + d)

   (h) $\neg n$ (From : f + c)

   (i) $\neg m$ (From : h + a)

   (j) $\neg n$ (From : i + b)

   (k) $\neg m$ (From : j + a)

   (l) i (From : k + e) Not taking step b to resolve as it leads to a loop.

   (m) $\square$ (From : l + f)

   Thus the above resolution shows that the system is inconsistent and contradictory.

4. (18 points) Consider the following English sentences.

   All cats are mammals.

   The head of a cat is the head of a mammal.

   (a) (6 points) Convert the sentences into first-order predicate logic. Be extremely careful about quantification; because not everything in the universe is a mammal, or a cat, or a head, you will need both kinds of quantification. Use the following lexicon:

   Predicates:

   cat(X) – X is a cat.

   mammal(X) – X is a mammal.

   headOf(H,X) – H is the head of X.

   Answer

   The sentenses in First order Logic can be represented as

   First formula: $\forall X(cat(X) \rightarrow mammal(X))$

   Second Formula: $\forall H(\exists X(cat(X) \wedge headOf(H,X)) \rightarrow \exists Y(mammal(Y) \wedge headOf(H,Y)))$

   (b) (4 Points) Convert the logic statements into CNF. HINT: With this lexicon, you will need two Skolem constants if you're doing this correctly.

   Answer

   CNF Conversion

   For the first formula:

   $\forall X(cat(X) \rightarrow mammal(X))$

   $\forall X(\neg cat(X) \vee mammal(X))$

   CNF : $\neg cat(X) \vee mammal(X)$

   For the second formula.

   $\forall H(\exists X(cat(X) \wedge headOf(H,X)) \rightarrow \exists Y(mammal(Y) \wedge headOf(H,Y)))$

   $\forall H(\neg(\exists X(cat(X) \wedge headOf(H,X))) \vee \exists Y(mammal(Y) \wedge headOf(H,Y)))$

   $\forall H(\forall X(\neg cat(X) \vee \neg headOf(H,X)) \vee \exists Y(mammal(Y) \wedge headOf(H,Y)))$

   $\forall H \forall X \exists Y((\neg cat(X) \vee \neg headOf(H,X)) \vee (mammal(Y) \wedge headOf(H,Y)))$

   $(\neg cat(X) \vee \neg headOf(H,X)) \vee (mammal(F(X,H)) \wedge headOf(H,F(X,H)))$

   $(\neg cat(X) \vee \neg headOf(H,X)) \vee (mammal(F(X,H)) \wedge headOf(H,F(X,H)))$

   CNF: $\neg cat(X_1) \vee \neg headOf(H_!,X_1) \vee mammal(F(X_1,H_1))$

   CNF: $\neg cat(X_2) \vee \neg headOf(H_2,X_2) \vee headOf(H_2,F(X_2,H_2))$

   (c) (6 Points) Using resolution and the 4-part heuristic presented in class, prove using FOPL resolution that the second of the original sentences follows from the first. Number your clauses, and indicate explicitly step-by-step what resolves together, under what substitution.

   Answer

   CNF for negation of the second formula

   $\neg(\forall H(\exists X(cat(X) \wedge headOf(H,X)) \rightarrow \exists Y(mammal(Y) \wedge headOf(H,Y))))$

   $\neg(\forall H \neg(\exists X(cat(X) \wedge headOf(H,X)) \vee (\exists Y(mammal(Y) \wedge headOf(H,Y)))))$

$\exists H(\exists X(cat(X) \wedge headOf(H,X)) \wedge \neg(\exists Y(mammal(Y) \wedge headOf(H,Y))))$
$\exists H(\exists X(cat(X) \wedge headOf(H,X)) \wedge (\forall Y \neg(mammal(Y) \vee \neg headOf(H,Y))))$
$\exists H \exists X \forall Y((cat(X) \wedge headOf(H,X)) \wedge (\neg(mammal(Y) \vee \neg headOf(H,Y))))$
$((cat(X_0) \wedge headOf(H_0,X_0)) \wedge (\neg(mammal(Y) \vee \neg headOf(H_0,Y))))$

CNF :
$cat(X_0)$
$headOf(H_0,X_0)$
$\neg mammal(Y) \vee \neg headOf(H_0,Y)$

Resolution Proof:

i. $\neg cat(X) \vee mammal(X)$
ii. $cat(X_0)$
iii. $headOf(H_0,X_0)$
iv. $\neg mammal(Y) \vee \neg headOf(H_0,Y)$
v. $\neg cat(X) \vee \neg headOF(H_0,X)$ (From iv + i) (substituting X/Y)
vi. $\neg headOF(H_0,X_0)$ (From v + ii) (Substitution : X_0/X)
vii. $\square$ (From vi + iii) (Substitution : empty)

Thus we can derive the conclusion which is the second statement by substitutions (X/Y, X_0/X, ()).

(d) (2 Points) Show a shorter proof that doesn't use the heuristic, if you can find one.
Answer
The proof above is the shortest possible one.