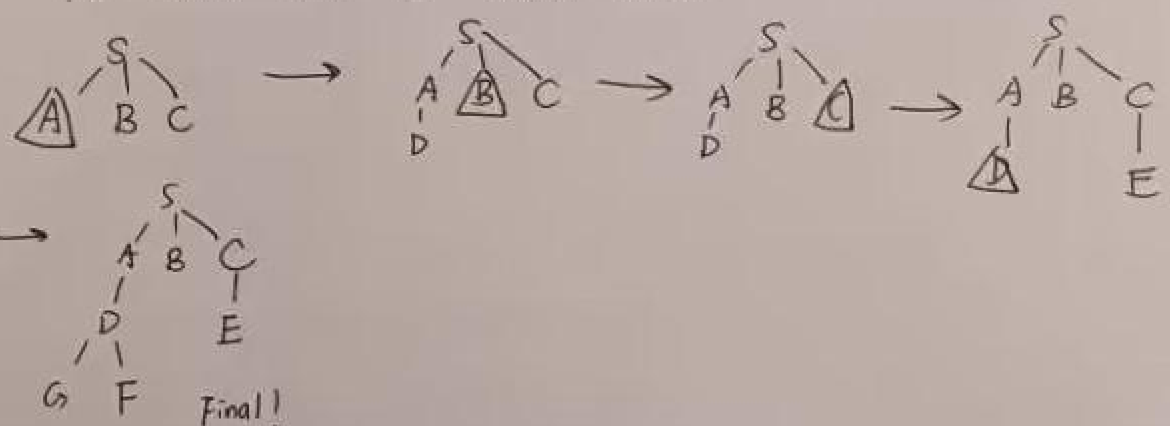
AAI Homework 1

Name: 吉辰卿 SID: 12332152

1.

1.1 solution

The search tree is shown below:



Final!

so, the points to be visited in order using BFS is:

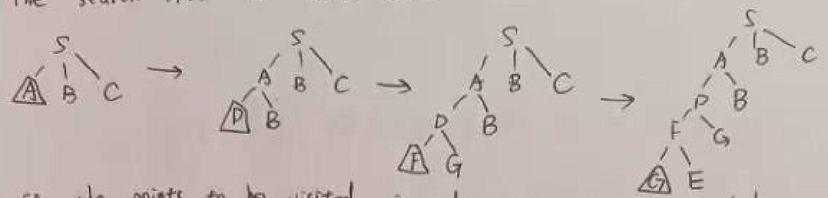
$S \to A \to B \to C \to D$

And the path is: $S \to A \to D \to G$, path cost is: $7+8+30=45$

so, the cost of the solution is 45.

1.2 solution:

The search tree is shown below.



Final !

so, the points to be visited in order using DFS is:

S → A → D → F → G

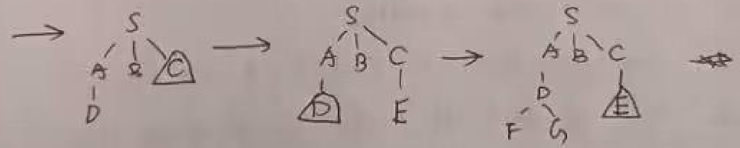
And the path is: S → A → D → F → G , the path cost is: 7+8+12+5 = 32

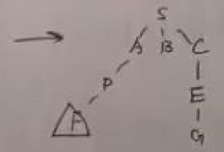
so, the cost of the solution is 32.

1.3 solution

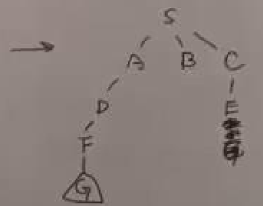
The search tree is shown below.

(the cost S → B → A is equal to S → A , so we don't add)





(because S → C → E → F is not shorter than S → A → D → F so we don't add it ; And S → C → E → G is shorter than S → A → D → G, so we add it).



Final !

(because ~~S →~~ S → A → D → F → E is not shorter than S → C → E , so we don't add it; And S → A → D → F → G is shorter than S → C → E → G , so we add it)

So, the final search tree is:

So, the points to be visited in order using UCS is:

S → B → A → C → D → E → F → G

And the path is: S → A → D → F → G, the path cost is: 7+8+12+5
= 32

So, the cost of the solution is 32.

2.

2.1 Solution

We can use the path representation. It's a natural way to encode TSP tours. That's to say, we can use a sequence of cities to represent a solution.

For example, the path A−D−E−C−B−O−A is a solution to this problem. And to avoid the duplication, we remove the city A in the path representation. So, in this example, the path representation is ADECBO.

For another example, the path representation for two parents in this problem:

parent 1 : AOBDEC   parent 2: AOBCED

The thing we should declare that, in this problem, the distance between the two disconnected points is + inf. So, the path representation such as: ~~ACEOBD~~ ACEOBD is also right. We can set the proper fitness function to die out these paths in the offspring(s)

2.2 solution

In this problem, we know the shorter path is, the better solution is. So, we can select the fitness function is the inverse of the each tour length.

That is: $f = \dfrac{1}{\sum_{i=1} distance_i \text{ (in each connection)}}$

2.3 solution

We can use the Partial - Mapped Crossover in this problem.

Firstly, we randomly select ~~to~~ two cut points on both parents.

Then, we swapping the substring between the two cut points in both parents.

Next, we should do the conflict detection. That is, the inverse replacement is applied outside of the cut points in order to eliminate the duplicates and recover all cities.

Take an example: two parents are: AOBDEC and ADECBO

The cut points are:

Cut point 1 Cut point 2

parent 1 : A O ¦ B D ¦ E C

parent 2 A D ¦ E C ¦ B O

Then, swapping the substering between the two cut points, we get:

offspring (intermediate)

1 A O ¦ E C ¦ E C

2 A D ¦ B D ¦ B O

Next, do conflicts detection. In offspring 1, in the outside of cut points, "EC" is duplicates, which will be replaced by "BD" due inverse replacement

Then, in offspring 2, "D" and "B" in the outside of cut points are duplicated, which will be replaced by "C" and "E" due inverse replacement.

Finally, the two offspring will be:

offspring 1 AOECBD

offspring 2 ACBDEO

From the analysis in 2.1, these two offsprings is right because the distance between the two disconnected points is +inf. so the Figure 3 is a complete graph. Then, we let this algorithm go and the next offspring may probably screen out the paths which distance is +inf. (Due to fitness function)

2.4 solution:

we can randomly swap two cities from ~~both parents~~ a offspring

For example: a offspring: A O B D E C

so, we swap the "D" and "C", we can get the offspring after mutation: A O B C E D, it is right (legal)

3.

3.1 solution For these two tours, we can get the initial edge map:

city 1 has edges to: 2, 6, 7, 9

city 2 has edges to: 1, 3, 5, 8

city 3 has edges to: 2, 4, 5, 7

city 4 has edges to: 3, 5, 8, 9

city 5 has edges to: 2, 3, 4, 6

city 6 has edges to: 1, 5, 7, 9

city 7 has edges to: 1, 3, 6, 8

city 8 has edges to: 2, 4, 7, 9

city 9 has edges to: 1, 4, 6, 8

When city 1 is selected, we update the edge map, that is:

city 1 selected
city 2 has edges to 3. 5. 8
city 3 ---: 2. 4. 5. 7
city 4 ---: 3. 5. 8. 9
city 5 ---: 2. 3. 4. 6
city 6 ---: 5. 7. 9
city 7 ---: 3. 6. 8
city 8 ---: 2. 4. 7. 9
city 9 ---: 4. 6. 8

city 2. 6. 7. 9 can be selected, for the smallest ~~*~~ city value, we select the city 2.

Then, city 2 is selected, we update the edge map, that is:

city 1 selected
city 2 selected
city 3 has edges to: 4. 5. 7
city 4 ---: 3. 5. 8. 9
city 5 ---: 3. 4. 6
city 6 ---: 5. 7. 9
city 7 ---: 3. 6. 8
city 8 ---: 4. 7. 9
city 9 ---: 4. 6. 8

city 3. 5. 8 can be selected, for the smallest city value, we select the city 3.

Then, city 3 is selected, we update the edge map, that is:

city 1 selected
city 2 selected
city 3 selected
city 4 has edges to: 5. 8. 9
city 5 ---: 4. 6
city 6 ---: 5. 7. 9
city 7 ---: 6. 8
city 8 ---: 4. 7. 9
city 9 ---: 4. 6. 8

city 5 and city 7 can be selected (have smallest adjacency edge), for the smallest city value, we select the city 5.

Then, city 5 is selected, we update the edge map, that is:

city 1 selected
city 2 selected
city 3 selected
city 4 has edges to: 8.9
city 5 selected
city 6 ··· : 7.9
city 7 -- : 6. 8
city 8 ---: 4. 7. 9
city 9 -- : 4. 6. 8

city 4. 6 can be selected, for the smallest city value, we select the city 4.

Then, city 4 is selected, we update the edge map, that is:

city 1 selected
city 2 selected
city 3 selected
city 4 selected
city 5 selected
city 6 has edges to: 7. 9
city 7 ···: 6. 8
city 8 --· : 7. 9
city 9 -- : 6. 8

city 8. 9 can be selected, for the smallest city value, we select the city 8.

Then, city 8 is selected, we update the edge map, that is:

city 1 : selected
city 2 : selected
city 3 selected
city 4 : selected
city 5 : selected
city 6 has edges to: 7. 9
city 7 ----- : 6
city 8 selected
city 9 ---- : 6

city 7. 9 can be selected, for the smallest city value, we select the city 7.

Then, city 7 is selected, we update the edge map, that is:

city 1: selected

city 2: selected

city 3: selected

city 4: selected

city 5: selected

city 6 has edges to: 9

city 7: selected

city 8: selected

city 9 has edges to: 6

Only city 6 can be selected, so we selected the city 6.

Then, we selected the last city --- city 9.

So, from the above analysis, the final tour is: 123548769 by using the Edge Recombination crossover (ER).

3.2 solution:

By using the alternate edges crossover in the adjacency representation, we firstly encode the two parents tour, that is:

parents 1: 123456789 $\xrightarrow{\text{after encode}}$ 234567891

parents 2: 173528496 $\xrightarrow{\text{after encode}}$ 785921346

If city 1 is selected as the starting city and alternate from the second path, that is: in parent 2: edge (1,7) is firstly selected. Then edge (7,8) in parents 1 is selected. Then, edge (8,4) in parents 2, (4,5) in parents 1, (5,2) in parent 2, (2,3) in parent 1 are successively selected.

Then, edge (3,5) in parent 2 is selected, but this edge will introduce a cycle. Based on that we should select the city with the smallest value by default, therefore (3,6) is selected.

Then, in parent 1, edge (6,7) is selected, but this edge will also introduce a cycle, so, based on the same principle above, ~~the edge~~ (6,9) is selected.

Finally, the tour is completed with the edge (9,1).

Therefore, the offspring (encoded) is: 7 3 6 5 2 9 8 4 1

So, we decode the offspring and get the decoded offspring: 178452369

Therefore, the final tour is 178452369 and all edges in the offspring are inherited from the parents, apart from the edges (3,6) and (6,9)

3.3 solution

The alternate edge operator introduces many random edges in the offspring when the choices for extending the tour are limited. ~~[illegible]~~

In this method, the offspring may be illegal (due to random edges)

However, the edge recombination crossover (ER) is less likely to require the selection of a random edge, so the introduction of random edges should be minimized. The resulting offspring by this method will be more reliable.