Homework Assignment #19 Key

For the following 4-city TSP problem assume that the initial *BSSF* is infinite and that the city cost/distance matrix is

∞	7	3	12
3	∞	6	14
5	8	∞	6
9	3	5	∞

This is the same as previous homework and the initial state should start with the same reduced cost matrix.

Question 1: Use the partial path state search approach we discussed in class. This assumes the path starts at city 1, each node represents a city, and a link from a parent to a child node in the search space means a path between the cities. Expanding a node means generating a child state for each node to which the parent node has a path.

The state for the root of the search tree is the answer to question 1. Show the search tree that branch and bound would generate for this problem. For each state show the bound of that state and also show when *BSSF* is updated and use it for proper pruning, etc.

State 1	∞	4	0	8
Cost = 15	0	∞	3	10
BSSF = ∞	0	3	∞	0
	6	0	2	∞

We put this initial state in the Queue = <(S:1,C=15)>
This is the only node on the queue so we expand it. It will have 3 children corresponding to edges from city 1 to cities 2, 3, and 4, which are edges (1,2), (1,3), and (1,4). You could also expand the 4th option of (1,1) which will always be infinity. For the rest of the problem I will assume that we do not expand a child whose "into" node is a node already included in the path from that node to the root. (Except in the case of the very last edge which always has city 1 as it's "into" node). I could have expanded each one of these, and at the next step they would have gotten to an infinite cost anyway. I will also use - in place of ∞.

State (1-2)	-	-	-	-		-	-	-	-
Cost = parent state cost	0	-	3	10		0	-	3	10
+ cost of new edge	0	-	-	0		0	-	-	0
+ cost to reduce matrix	6	-	2	-		4	-	0	-
= 15 + 4 + 2 = 21									

The initial matrix on the left comes from setting the from row to infinity and the to column to infinity. The matrix on the right is the result of reducing the new matrix. reduction of 2 came from that last row which had no zero's after setting column two to infinity.

State (1-3)	-	-	-	-	no reduction necessary
Cost = 15 + 0 + 0 = 15	0	-	-	10	
	0	3	-	0	
	6	0	-	-	

State (1-4)	-	-	-	-	-	-	-	-
Cost = 15 + 8 + 2 = 25	0	-	3	-	0	-	1	-
	0	3	-	-	0	3	-	-
	6	0	2	-	6	0	0	-

These 3 states are put in the queue with lowest bound at 15.
Queue = <(S:1-3,C=15), (S:1-2,C=21), (S:1-4,C=25)>
We will expand the state with the lowest bound (state 1-3) which removes that state from the queue and produces 2 children

State (3-2)	-	-	-	-	-	-	-	-
Cost = 15 + 3 + 10 = 34	0	-	-	10	0	-	-	0
	-	-	-	-	-	-	-	-
	6	-	-	-	0	-	-	-

State (3-4)	-	-	-	-	no reduction necessary
Cost = 15 + 0 + 0 = 15	0	-	-	-	
	-	-	-	-	
	6	0	-	-	

These 2 are put on the queue.
Queue = <(S:3-4,C=15), (S:1-2,C=21), (S:1-4,C=25), (S:3-2,C=34)>
We take the lowest in the queue (state 3-4). The only possible state from there is (4-2), since 1 is already in the path to the root and we are not yet at the last edge.

State (4-2)	-	-	-	-	no reduction necessary
Cost = 15 + 0 + 0 = 15	0	-	-	-	
	-	-	-	-	
	-	-	-	-	

Queue = <(S:4-2,C=15), (S:1-2,C=21), (S:1-4,C=23), (S:3-2,C=28)>
State (4-2) is now expanded. There is just one state left, which will be our last edge (2-1) with no new costs giving a solution with cost 15. We set *BSSF* = 15. We can now prune any state on the queue with bound greater than or equal to 15. That deletes all of them and we are done.

Question 2: This time use the include/exclude state search approach we discussed in class. This assumes particular start city, and at each branch chooses one edge to include/exclude from the solution. At each branch, choose the edge which maximizes bound($S_{excluded}$) – bound($S_{included}$)

Show the search tree that branch and bound would generate for this problem. For each state show the bound of that state and also show when BSSF is updated and use it for proper pruning, etc.

We start with the reduced cost matrix

State 1	∞	4	0	8
Cost = 15	0	∞	3	10
BSSF = ∞	0	3	∞	0
	6	0	2	∞

We put this initial state in the Queue = <(S:1,C=15)>
This is the only node on the queue so we expand it into two children. We now need to select which edge to include/exclude. The only options are those with a 0 in the cost matrix which are edges (1,3), (2,1), (3,1), (3,4), and (4,2). Examining the options suggests that (3,4) gives us a maximum bound($S_{excluded}$) – bound($S_{included}$). I will use - in place of ∞.

State (inc(3-4))	-	4	0	-	no reduction necessary
= 15 + 0 + 0 = 15	0	-	3	-	
	-	-	-	-	
	6	0	-	-	

Note that in the first matrix we also excluded edge (4-3) since that could make a premature cycle

State (exc(3-4))	-	4	0	8		-	4	0	0
= 15 + 0 + 8 = 23	0	-	3	10		0	-	3	2
	0	3	-	-		0	3	-	-
	6	0	2	-		6	0	2	-

Queue = <(S:inc(3-4),C=15), (S:exc(3-4),C=23)>
We now expand State inc(3,4) and consider which edge out of (1,3), (2,2), (4,2) is best. Edge (4,2) looks promising.

State (inc(4-2))	-	-	0	-	no reduction necessary
= 15 + 0 + 0 = 15	0	-	-	-	
	-	-	-	-	
	-	-	-	-	

Note that to avoid premature cycle's we must exclude edge (2-4) [it was already infinity] and we also need to exclude edge (2-3). The reason we need to exclude (2-3) is because the current branch of this tree includes edges (3-4) and (4-2), which means that going from edge 2 to 3 could make a premature cycle of length 3.

State (exc(4-2))	-	4	0	-		-	0	0	-
= 15 + 0 + 10 = 25	0	-	3	-		0	-	3	-
	-	-	-	-		-	-	-	-
	6	-	-	-		0	-	-	-

These 2 states go on the queue.
Queue = <(S:inc(4-2),C=15), (S:exc(3-4),C=23), (S:exc(4-2),C=25)>

We expand state inc(4-2). Note that it only has 0's left in the matrix. It makes no difference which edge we take first, since after two more expansions we will get always end up with edges (1,3) and (2,1) as part of the path. Neither add any cost and so we have a first solution of cost 15. *BSSF* is then set to 15. Since all the states on the queue have bound greater than or equal to 15 they are all pruned and we are done.