CSC226 - Assignment 5 Solutions

1. Explain what `Create` is doing. It is guaranteed to generate a maze that always has a unique solution. Why?

`Create(x,y,val)` takes as input the x,y coordinate of a cell and a value val that represents an adjustment to m[x][y] based on a wall at (x,y) being knocked down. That value is initially 0, meaning that no wall is knocked down, after that the value depends on which wall needs knocking down. `Create` is essentially doing a depth-first-search (DFS) traversal of the underlying 4-regular dual graph with the adjacency lists permuted randomly, so that a random looking maze is produced. The walls being knocked down correspond to the tree edges in the DFS. Since the tree edges are a tree, every cell is connected to every other cell, and there is a unique path between every cell (similar sort of reasoning came up in an earlier assignment where you were counting walls). The value of 15 indicates a cell (vertex) that has not been visited before; after visiting it, the value is changed from 15 so Create is called exactly once for each cell.

2. If there are *n* rows and *m* columns, then how many times is `Create` called when making a maze?

`Create` is called exactly once for each cell. Therefore, it is called n*m times when creating a maze.

3. What is the purpose of the `p^2`?

In general `p^2` will flip the next-to-lowest-order bit in word. In our case p is one of 0,1,2,3 and so p^2 swaps 0 <-> 2 and 1 <-> 3, meaning the opposite wall. This is useful for us, because when we knock down a wall in a cell, we need to knock down the opposite wall in an adjacent cell.

4. From the backtracking handout: Question 2 about estimating the size of the tree. Compare with the actual size of the tree.

From handout: Generate an estimate of the number of vertices in the backtracking tree for the 8 by 8 Queens problem. In picking your "random" row positions, simply use the lowest numbered valid row.

Q	x	x	x	x	x	x	x
x	x	x	Q	x	x	x	x
x	Q	x	x	x	x	x	x
x	x	x	x	Q	x	x	x
x	x	Q	x	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x
x	x	x	x	x	x	x	x
8	6	4	3	2	0	0	0

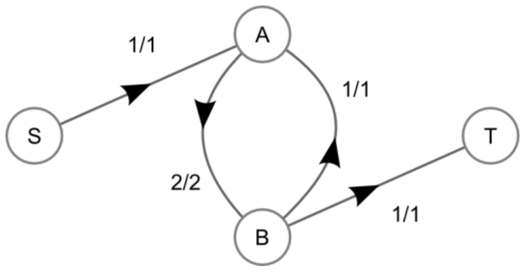
1 + 8 + 8*6 + 8*6*4 + 8*6*4*3 + 8*6*4*3*2 + 0 + 0 + 0 = 1977

5. From the book: Exercise 6.38.

True or false. If true provide a short proof, if false give a counterexample:

a) In any max flow, there is no directed cycle on which every edge carries positive flow.

False.



An excess of one unit of flow is carried around the cycle giving positive flow for every edge.

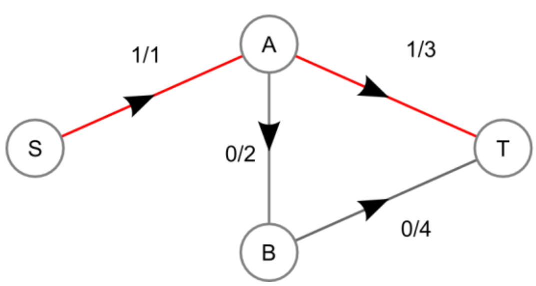
b) There exists a max flow for which there is no directed cycle on which every edge carries positive flow.

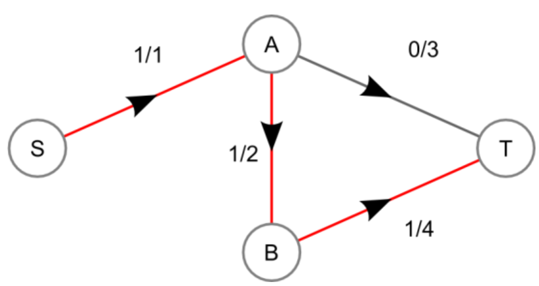
True.

If you reduce the flow around the cycle by the amount of flow through the edge carrying the minimum amount, the overall max flow is not changed. This will give an edge in the cycle with zero flow.

c) If all edge capacities are distinct, the max flow is unique.

False.

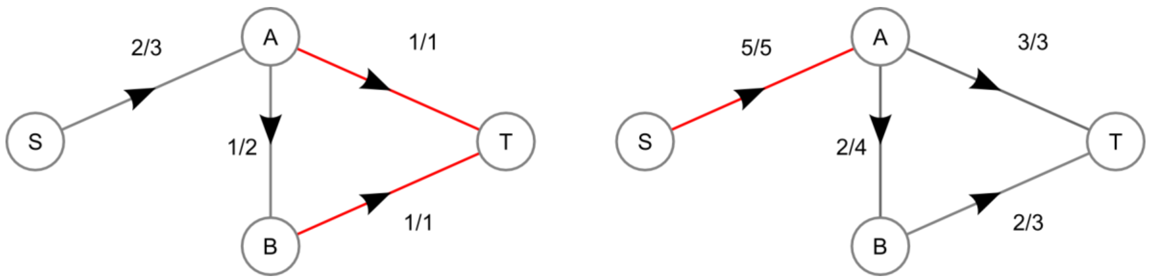




There are two ways to achieve the max-flow of 1 through this network in which each edge has a unique capacity.

d) If all edge capacities are increased by an additive constant, the min cut remains unchanged.

False.

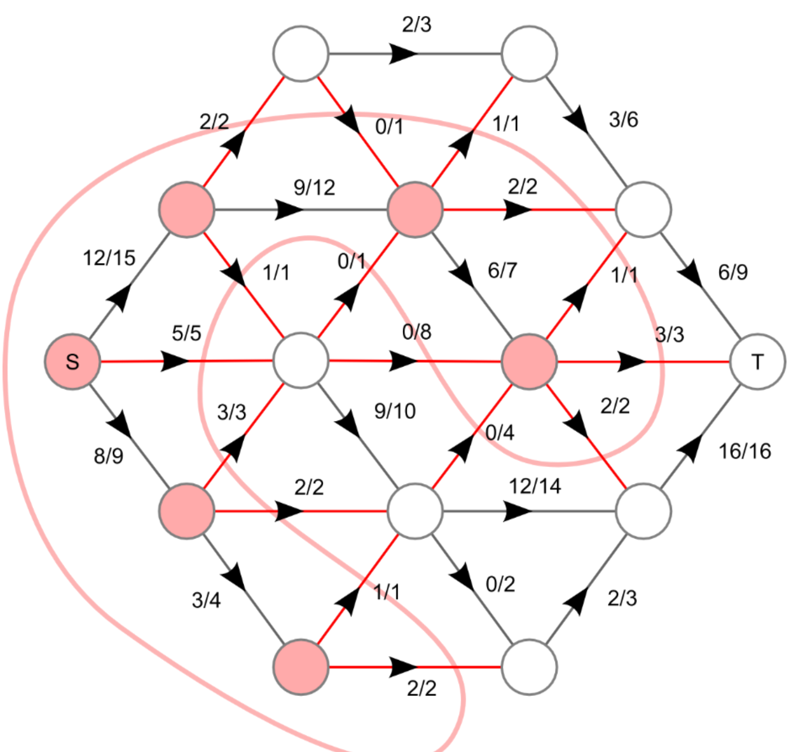


In first case, min-cut divides network into {S,A,B} and {T}. Add 2 to each edge capacity and the min-cut now changes to divide the network into {S} and {A,B,T}.

e) If all edge capacities are multiplied by a positive integer, the min cut remains unchanged.

True. Since all capacities are scaled by the same amount, the relative order of the capacities is unchanged and the min-cut is the same.

6. Find the max-flow and min-cut in the attached network as it would be found by the Ford-Fulkerson algorithm.



Max flow is 25. Edges of min-cut are red. Nodes belonging to S set are coloured red. Nodes belonging to T set are coloured white.