cmput 304 2024 study questions 3 (with solutions)

1. In the box at right,
   for `H = [[0,1,2],[1,0,2],[1,0,2]]`
   and `R = [[1,2,0],[0,1,2],[2,1,0]]`,
   show the output printed by `m=propose_reject(H,R)`.

```
def propose_reject(H,R):
  n = pref_system_size(H,R)
  F,C = [None] * n, [0 for j in range(n)]
  rejection = True
  while rejection:
    rejection = False
    for j in range(n):
      h_choice = H[j][C[j]] # current H proposal
      if F[h_choice] == None: #R has no prop'ls
        F[h_choice] = j
        print(' ',j,' prop ',h_choice,': maybe')
      elif F[h_choice] != j: #R has 2 prop'ls
        r_maybe = F[h_choice] #R's current prop'l
        if prefers(R[h_choice], j, r_maybe):
          r_reject, r_maybe = r_maybe, j
          F[h_choice] = r_maybe
        else:
          r_reject = j
        print(' ',j,'prop',h_choice,
              ':pref',r_maybe,',rej',r_reject)
        C[r_reject] += 1 # H[j_rej.]: next pref
        rejection = True # a prop'l was rejected
  P = [H[j][C[j]] for j in range(n)]
  print('\nj  P  C  F')
  [print(j, P[j], C[j], F[j]) for j in range(n)]
  return P
```

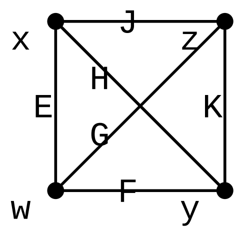
Show your rough work here.

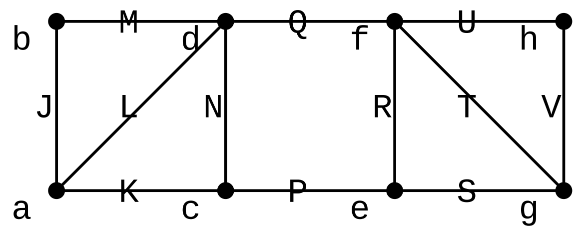
2. Give a matching preference system with size 3 for which the propose-reject algorithm always finds a stable matching, or explain why this is not possible.

3.

```
def myunion(x,y,P):
  rootx = findGP(x,P)
  rooty = findGP(y,P)
  P[rootx] = rooty

def findGP(x, P):
  px = P[x]
  if x==px: return x
  gx = P[px] #grandparent
  while px != gx:
    P[x] = gx
    x = px
    px = gx
    gx = P[gx]
  return px
```

```
Here P represents 8 components of size 1:

j     0  1  2  3  4  5  6  7
P[j]  0  1  2  3  4  5  6  7

Show P after myunion(1,2,P):

P[j] __ __ __ __ __ __ __ __

... and then after myunion(2,3,P):

P[j] __ __ __ __ __ __ __ __

... and then after myunion(3,4,P):

P[j] __ __ __ __ __ __ __ __

... and then after myunion(4,5,P):

P[j] __ __ __ __ __ __ __ __
```

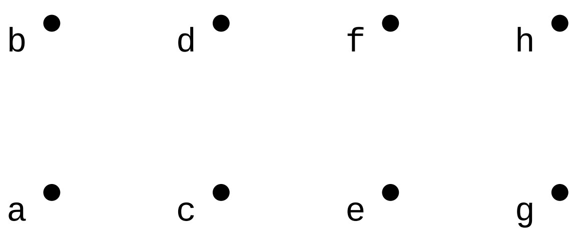




Recall: a *cut* of a graph is a partition of the node set into two non-empty subsets. E.g. on the small graph (above left), {{w,x}, {y,z}} is a cut with cross-edges {F,G,H,J}. RKMC is the randomized Kruskal min cut algorithm: unless otherwise stated, its input is a uniform-random permutation of the edges.

4. For the big graph, give each min cut (partition and cross-edges) ...

5. ...and give the forest (draw on the nodes below) and cut (partition and cross-edges) found by RKMC when edges are input in order QPRKNTMLJUVS.



6. a) By the theorem from class, for the graph above, if we run RKMC one time, we know from the theorem in class that the probability that the cut found is a min cut is at least what?

   b) Let a be probability from part a). Let p be the true probability that, for the graph above, if we run RKMC one time, the cut found is a min cut. Which of these is true: $a < p$, $a = p$, $a > p$? Prove your answer.

7. a) Consider a preference system for the stable matching problem in which hospital h's first choice is resident r and resident r's first choice is hospital h. Prove/disprove: in any stable matching, h is matched with r.

   b) Prove/disprove: for any $n \geq 1$, there is a preference system of size n (n hospitals and n residents) with only one stable matching.

solutions

1. Run class github code `/graphs/stablematch/sm.py`. Check your answer in the quiz 3 solutions from 2023.

2. Check your answer in the quiz 3 solutions from 2023.

3. You can find this code in `/graphs/paths/UF.py`. Check your answer in the quiz 3 solutions from 2023.

4. Check your answer in the quiz 3 solutions from 2023.

5. Check your answer in the quiz 3 solutions from 2023.

6. a) at least $2/(n(n-1)) = 2/(8*7) = 1/28$.

   b) $a < p$. There are three min cuts, with respective cross-edge sets $\{J, M\}$, $\{U, V\}$, $\{P, Q\}$. So we get a min cut if J, M are the last two edges picked, so if J is last and M second-last $(1/12 * 1/11)$ or M is last and J is second-last (same probability), so total $1/66$. We also get a min cut if U, V are the last two edges picked, or if P, Q are the last two edges picked. These three events are independent (they do not interfere with each other), so we get a min cut with probability at least $1/66 + 1/66 + 1/66 = 3/66 = 1/22 > 1/28$. So $a < p$.

   (In fact, with a bit more work we can prove that p is even bigger, e.g. if the last three edges picked are $\{J, M, T\}$: there are many more cases.)

7. a) Proof. Consider any matching in which h and r are not matched. Then h is matched with some $r' \neq r$ and r is matched with some $h' \neq h$. Then h prefers r to its match r' and r prefers h to its match h', so $\{h, r\}$ is an unhappy couple, so the matching is not stable.

   b) Consider the preference system in which, for each index j, hospital h_j's first choice is resident r_j and resident r_j's first choice is hospital h_j. Then by a) we know that in any stable matching, we must have h_j matched with r_j for each j. But there is only 1 such matching.