Graphs, Graph Algorithms, and Optimization
Problem set 8

W 24/25

- **GitHub** https://github.com/JanPastorek/1-AIN-413-22-Graphs

---

Today's exercises are based on:

- Stanoyevitch, A. (2011). *Discrete structures with contemporary applications*. CRC Press, Taylor & Francis Group.
- West, D. B. (2001). *Introduction to graph theory* (2nd ed). Prentice Hall.

Recall:

- **Matching** is a set of non-loop edges with no shared endpoints. vertices incident to the edges of a matching are **saturated**. others are **unsaturated**. **perfect matching** is when all vertices are saturated
- A graph G is **bipartite** if vertex set V can be partitioned into two subsets $V = U \cup W$ such that each edge of G has one endpoint in U and one in W.
- **maximal matching**: matching that cannot be enlarged by adding another edge
- **maximum matching:** -II- that is of maximum size among all other matchings
- Maximal $\neq$ Maximum

3.1.5. Example. *Maximal* $\neq$ *maximum.* The smallest graph having a maximal matching that is not a maximum matching is P_4. If we take the middle edge, then we can add no other, but the two end edges form a larger matching. Below we show this phenomenon in P_4 and in P_6. ■

  –

- **M-alternating path** is a path that alternates between edges in M and edges not in M
- **M-augmenting path** is a path whose (both) endpoints are unsaturated by M
- Given a matching $M; x, y$ form a **rouge couple**, if they prefer each other to their mates in M
- A matching is **stable**, if there are no rouge couples

Maximum Matching Algorithm

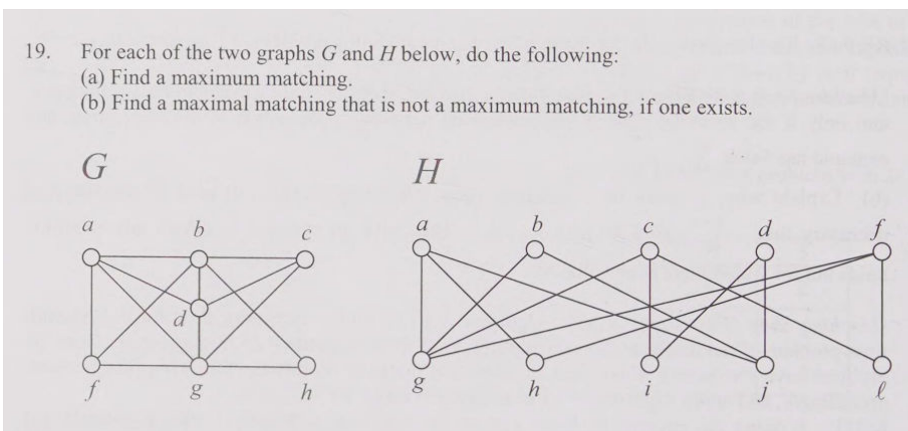
```
# Create vertices s and t; connect them respectively with the set X and Y
# Give new edges capacity 1
# Original edges have capacity capacity greater than |X|
# A complete matching exists if and only if it uses all edges from vertex s
# and the size of the maximum flow is |X|

# For Max-Flow use FORD-FULKERSON / EDMONDS-KARP
flow = 0
for each edge (u, v) in G:
```

```
    flow(u, v) = 0
while there is a path, p, from s -> t in residual network G_f:
# use DFS / BFS to find p: s -> t
# residual network is reconstructed from original graph
# by $c_f(u,v)=c(u,v)-f(u,v)$.
    residual_capacity(p) = min([residual_capacity(u, v) for (u, v) in p])
    # the min is the bottleneck value
    flow = flow + residual_capacity(p)
    for each edge (u, v) in p:
        if (u, v) is a forward edge:
            flow(u, v) = flow(u, v) + residual_capacity(p)
        else:
            flow(u, v) = flow(u, v) - residual_capacity(p)
return flow
```

Problem 0. [Any questions?]

Is there anything unclear from the lectures or about the home project?

Problem 1. [Simulating algorithms]



19. For each of the two graphs G and H below, do the following:
(a) Find a maximum matching.
(b) Find a maximal matching that is not a maximum matching, if one exists.

Problem 2. [Matching properties of classes of graphs]

1. **3.1.2.** (−) Determine the minimum size of a maximal matching in the cycle C_n.

2. **3.1.8.** (!) Prove or disprove: Every tree has at most one perfect matching.

3. Use the algortihm for matching on the following graph, start from the bottom so that it is more interesting. Write a final matching. Is the found matching X-saturated (X is the left part of the graph bipartite graph)?

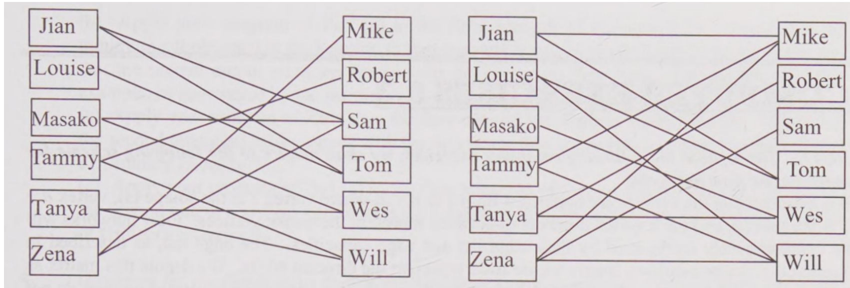


FIGURE 9.69: Two small marriage problems (a) (left), (b) (right).

4.

3.1.1. (−) Find a maximum matching in each graph below. Prove that it is a maximum matching by exhibiting an optimal solution to the dual problem (minimum vertex cover). Explain why this proves that the matching is optimal.

6. Prove the following:

3.1.13. Corollary. **For $k > 0$, every k-regular bipartite graph has a perfect matching.**

7. Suppose you are given an oracle that, given a graph G, tells you wether G has a perfect matching or not. Show how to use this oracle to determine the maximum cardinality matching of a graph $G(V, E)$. The total number of calls should be at most $|V| + |E|$. Hint: modify the graph at each call of the oracle.

Problem 3. [Stable marriage]

Gale-Shapley algorithm

```
Initialize all men (X) and women (Y) to free
while there exist a free man m who still has a woman w to propose to
    w = # highest ranked woman to whom m has not yet proposed
    if w is free:
       (m, w) become engaged
    else some pair (m_, w) already exists
       if w prefers m to m_:
          (m, w) become engaged
           m_ becomes free
       else:
          (m_, w) remain engaged
```

0. Think of an example where each men and women will not get his/her first choice.
1. what is the time complexity of Gale-Shapley algorithm algorithm?
2. The following are the tables of preferences of men (left) for women, and women for men (right). Each row corresponds to one person, the order of columns corresponds to the order of preferences. Solve the stable marriage problem for these inputs:

	1st	2nd	3rd	4th	5th
V	A	B	C	D	E
W	B	C	D	A	E
X	C	D	A	B	E
Y	D	A	B	C	E
Z	A	B	C	D	E

	1st	2nd	3rd	4th	5th
A	W	X	Y	Z	V
B	X	Y	Z	V	W
C	Y	Z	V	W	X
D	Z	V	W	X	Y
E	V	W	X	Y	Z