Algorithm Design and Analysis (Fall 2023)

Assignment 5

Deadline: Jan 2, 2023

1. (30 points) **[Menger's Theorem]** Let $G = (V, E)$ be a connected undirected graph. The *edge connectivity* of $s, t \in V$ is the minimum number of edges whose removal disconnects s and t. The *vertex connectivity* of s, t is the minimum number of graph elements (any vertex in $V \setminus \{s, t\}$ or any edge in E) whose removal disconnects s and t. Menger's Theorem states that the edge connectivity of s, t is exactly the number of edge-disjoint paths between s and t and the vertex connectivity of s, t is exactly the number of *internally* vertex-disjoint paths between s and t. (Obviously, two paths between s and t are not vertex-disjoint since they intersect at s and t; *internally* vertex disjoint means they are disjoint except for s and t.)

   Prove that Menger's Theorem is just a consequence of the max-flow-min-cut theorem. Be sure the network you construct is *directed* and *capacitated*.

2. (30 points) **[Perfect Matching on Bipartite Graph]** A graph is *regular* if every vertex has the same degree. Let $G = (A, B, E)$ be a regular bipartite graph with $|E| > 0$.

   (a) (10 points) Prove that $|A| = |B|$.

   (b) (15 points) Let $n = |A| = |B|$. Can we conclude that G must contain a matching of size n? If so, prove it; if not, provide a counterexample.

3. (40 points) **[König-Egerváry Theorem]** In the class, we have seen that the maximum matching problem can be formulated by the following linear program

$$\begin{aligned} \text{maximize} \quad & \sum_{e \in E} x_e \\ \text{subject to} \quad & \sum_{e:e=(u,v)} x_e \leq 1 && (\forall v \in V) \\ & x_e \geq 0 && (\forall e \in E) \end{aligned}$$

and the minimum vertex cover problem can be formulated by the following linear program

$$\begin{aligned} \text{minimize} \quad & \sum_{u \in V} x_u \\ \text{subject to} \quad & x_u + x_v \geq 1 && (\forall (u,v) \in E) \\ & x_u \geq 0 && (\forall u \in V) \end{aligned}$$

We have also seen that the second linear program is the dual program of the first.

(a) (20 points) Prove that both linear programs have integral optimal solutions if the graph is bipartite.

(b) (10 points) Using the result in the first part, prove König-Egerváry Theorem, which states that the size of the maximum matching equals to the size of the minimum vertex cover in a bipartite graph.

(c) (10 points) Provide a counterexample showing that the claim fails for non-bipartite graphs.

4. (Bonus 60 points) **[Dinic's Algorithm]** In this question, we are going to work out *Dinic's algorithm* for computing a maximum flow. Similar to Edmonds-Karp algorithm, in each iteration of Dinic's algorithm, we update the flow f by increasing its value and then update the residual network G^f. However, in Dinic's algorithm, we push flow along *multiple* s-t paths in the residual network instead of a *single* s-t path as it is in Edmonds-Karp algorithm.

   In each iteration of the algorithm, we find the *level graph* of the residual network G^f. Given a graph G with a source vertex s, its level graph $\overline{G}$ is defined by removing edges from G such that only edges pointing from level i to level $i+1$ are kept, where vertices at level i are those vertices at distance i from the source s. An example of level graph is shown in Fig. 1.

   Next, the algorithm finds a *blocking flow* on the level graph $\overline{G}^f$ of the residual network G^f. A blocking flow f in G is a flow such that each s-t path contains at least one *critical edge*. Recall that an edge e is *critical* if the amount of flow on it reaches its capacity: $f(e) = c(e)$. Fig. 2 gives examples for blocking flow.

   Dinic's algorithm is then described as follows.

   1. Initialize f to be the empty flow, and $G^f = G$.
   2. Do the following until there is no s-t path in G^f:
      - construct the level graph $\overline{G}^f$ of G^f.
      - find a blocking flow on $\overline{G}^f$.
      - Update f by adding the blocking flow to it, and update G^f.

   Complete the analysis of Dinic's algorithm by solving the following questions.

   (a) (15 points) Prove that, after each iteration of Dinic's algorithm, the distance from s to t in G^f is increased by at least 1.

   (b) (15 points) Design an $O(|V| \cdot |E|)$ time algorithm to compute a blocking flow on a level graph.

   (c) (10 points) Show that the overall time complexity for Dinic's algorithm is $O(|V|^2 \cdot |E|)$.

   (d) (20 points) **(challenging)** We have seen in the class that the problem of finding a maximum matching on a bipartite graph can be converted to the maximum flow problem. Show that Dinic's algorithm applied to finding a maximum matching on a bipartite graph only requires time complexity $O(|E| \cdot \sqrt{|V|})$.

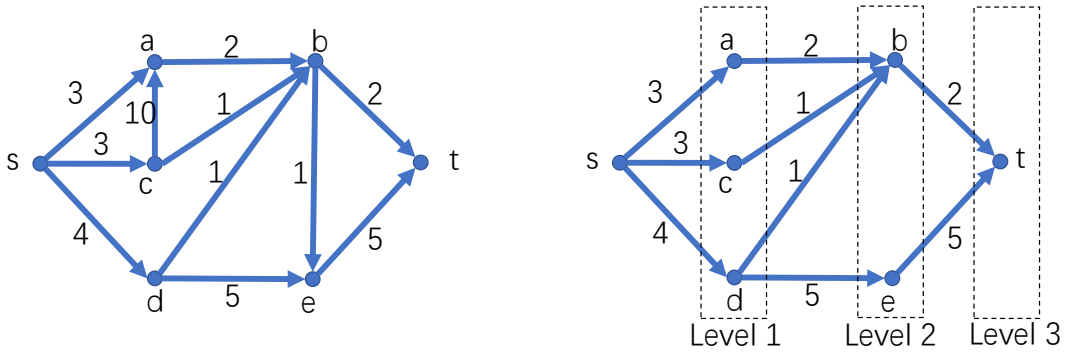

Figure 1: The graph shown on the right-hand side is the level graph of the graph on the left-hand side. Only edges pointing to the next levels are kept. For example, the edges (c, a) and (b, e) are removed, as they point at vertices at the same level. If there were edges pointing at previous levels, they should also be removed.

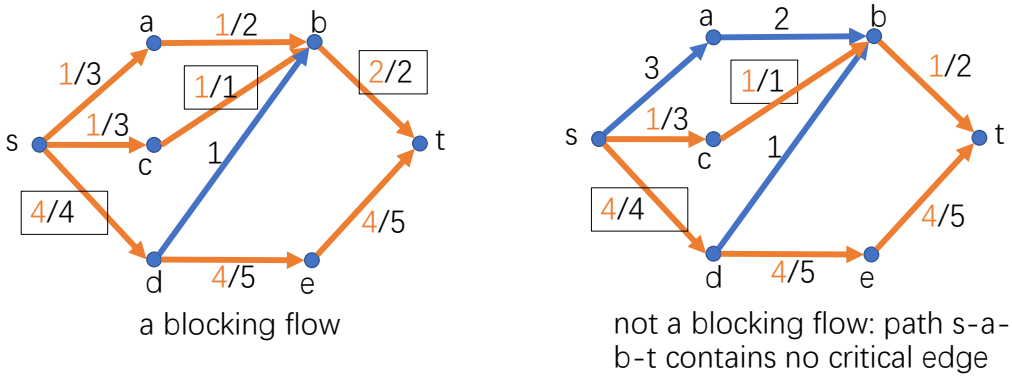

Figure 2: Blocking flow examples

5. How long does it take you to finish the assignment (including thinking and discussion)? Give a score (1,2,3,4,5) to the difficulty. Do you have any collaborators? Please write down their names here.