Computer Science C73 November 24, 2021
Scarborough Campus University of Toronto

Solutions for Homework Assignment #7

Answer to Question 1. From the given flow network $\mathcal{F} = (G, s, t, c)$ and the given maximum flow f of $\mathcal{F}$ we can construct the residual graph G_f of $\mathcal{F}$ with respect to flow f.

Let L be the set of nodes reachable from s in G_f, and let R be the set of nodes from which t is reachable in G_f. Since f is a maximum flow, there is no $s \to t$ path in G_f (otherwise, we could augment f along that path, contradicting that it is a maximum flow). Thus, L and R are disjoint and non-empty sets. An edge (u, v) of G is a bottleneck edge if and only if adding (u, v) to G_f creates an $s \to t$ path (we need to prove this!) — i.e., if and only if $u \in L$ and $v \in R$. So, our algorithm is as follows:

<u>BOTTLENECKS($\mathcal{F}$, f)</u>
1 $G_f :=$ residual graph of $\mathcal{F}$ with respect to max flow f
2 $L :=$ set of nodes reachable from s in G_f
3 $R :=$ set of nodes from which t is reachable in G_f
4 $B := \varnothing$ ▷ set of bottleneck edges
5 **for each** edge (u, v) of G **do**
6   **if** $u \in L$ and $v \in R$ **then** $B := B \cup \{(u, v)\}$
7 **return** B

Running time. Let n and m be the number of nodes and edges, respectively, of the flow network. Line 1 takes $O(m + n)$ time. Lines 2–3 take $O(m + n)$ time each (by a DFS or BFS of G_f starting from s for line 2 and a DFS or BFS of G_f with the edges reversed starting from t for line 3). Lines 4 and 7 take $O(1)$ time. The loop in lines 5–6 is executed most m times, and each iteration takes $O(1)$ time so the entire loop takes $O(m)$ time. So, the overall running time of the algorithm is $O(m + n)$.

Correctness. We need to prove:

$$(u, v) \in B \text{ at the end of the algorithm if and only if } (u, v) \text{ is a bottleneck.} \tag{1}$$

Fix any edge (u, v) of the given flow network $\mathcal{F} = (G, s, t, c)$, and let G_f be the residual graph of $\mathcal{F}$ w.r.t. the given maximum flow f. Let $\mathcal{F}^+ = (G, s, t, c^+)$ be the flow network obtained from $\mathcal{F}$ by increasing the capacity of edge (u, v) by any positive amount and keeping the capacity of all other edges the same as in $\mathcal{F}$. Clearly f satisfies the capacity and conservation constraints in $\mathcal{F}^+$, so it is a flow of $\mathcal{F}^+$. Let G_f^+ be the residual graph of $\mathcal{F}^+$ with respect to f. Since the only change to obtain $\mathcal{F}^+$ from $\mathcal{F}$ is the increase in the capacity of (u, v), we have:

$$G_f^+ \text{ is the same as } G_f \text{ with forward edge } (u, v) \text{ added (if it is not already in } G_f). \tag{2}$$

["only if" direction of (1)] Suppose $(u, v) \in B$. Since (u, v) is added to B, by line 6 and the definition of L and R (lines 2 and 3), the addition of (u, v) to G_f creates an $s \to t$ path. By (2), the residual graph of $\mathcal{F}^+$ with respect to f has an $s \to t$ path. Therefore f is not a maximum flow in $\mathcal{F}^+$. So, the increase in the capacity of (u, v) increases the value of the maximum flow; i.e., (u, v) is a bottleneck.

["if" direction of (1)] We prove the contrapositive. Suppose $(u, v) \notin B$. Thus, by line 6, either $u \notin L$ or $v \notin R$. By the definition of L and R (lines 2 and 3), this means that the addition of (u, v) to G_f does not create an $s \to t$ path. Therefore, by (2), G_f^+ does not have an $s \to t$ path. This means that f is a maximum flow of $\mathcal{F}^+$. So, increasing the capacity of (u, v) does not increase the value of the maximum flow; i.e., (u, v) is not a bottleneck.

Answer to Question 2. Given the degree-pairs for a set of nodes V, we define a flow network $\mathcal{F} = (G, s, t, c)$ as follows:

- G has a source node s, a target node t, and for each node $v \in V$, G has two nodes, v_{in} and v_{out}.
- For each node $v \in V$, G has an edge (s, v_{out}) with capacity $out(v)$, and an edge (v_{in}, t) with capacity $in(v)$.
- For each pair of nodes $(u, v) \in V \times V$ such that $u \neq v$, G has an edge (u_{out}, v_{in}) with capacity 1.

The intuition behind this construction is as follows: Every edge of capacity 1 from u_{out} to v_{in} represents a *potential* edge (u, v) in the graph that realizes the given degree-pairs. The edge (s, v_{out}) of capacity $out(v)$ represents the goal of creating $out(v)$ edges from v to other nodes. The edge (v_{in}, t) of capacity $in(v)$ represents the goal of creating $in(v)$ edges from other nodes to v. As we will show (see (3) below), if we can "pump" the maximum allowable traffic $out(v)$ on edge (s, v_{out}), for every v, we achieve these two goals, i.e., we realize the given degree-pairs; and the only way to achieve these two goals is by creating a flow that pumps $out(v)$ traffic on the edge (s, v_{out}), for every v. In other words,

$$\begin{gathered}\text{the given degree-pairs are realizable}\\ \text{if and only if the value of the maximum flow of } \mathcal{F} \text{ is } \sum_{v \in V} out(v).\end{gathered} \tag{3}$$

We now prove this statement.

[If] Let f be a maximum flow of $\mathcal{F}$, and suppose that $\mathcal{V}(f) = \sum_{v \in V} out(v)$. By the Integrality Theorem we can assume, without loss of generality, that f an an integral flow. We define the directed graph $R_f = (V, E)$, where $(u, v) \in E$ if and only if $f(u_{out}, v_{in}) = 1$. We now show that R_f realizes the given degree-pairs. For this, it suffices to prove that for every $v \in V$, the out-degree of v in R_f is $out(v)$ and the in-degree of v in R_f is $in(v)$.

Since $\mathcal{V}(f) = \sum_{v \in V} out(v)$ and, for every $v \in V$, the capacity of the edge from s to v_{out} is $out(v)$, this edge is saturated in f, i.e., for every $v \in V$, $f(s, v_{out}) = out(v)$. Since the edge (v_{out}, u_{in}) in the flow network $\mathcal{F}$ has capacity 1 and the flow f is integral, every edge out of v_{out} in F has a flow of either 0 or 1. Since $f(s, v_{out}) = out(v)$, there are exactly $out(v)$ edges out of v_{out} that carry flow in f. By definition of R_f, v has out-degree $out(v)$ in R_f.

By conservation, the total flow out of s must be equal to the total flow into t, and so the value of the maximum flow f is also equal to the total flow into t. That is, $\mathcal{V}(f) = \sum_{v \in V} in(v)$. By reasoning as in the previous paragraph (but now with respect to v_{in} and t instead of s and v_{out}) we can show that for every $v \in V$, v has in-degree $in(v)$ in R_f.

We have therefore shown that if the maximum flow of $\mathcal{F}$ is $\sum_{v \in V} out(v)$ then the given degree-pairs are realizable.

[Only if] Suppose some directed graph $R = (V, E)$ realizes the given degree-pairs. We define a function f_R on the edges of the flow network $\mathcal{F}$ as follows:

$$f_R(x, y) = \begin{cases} out(v), & \text{if } x = s, \text{ and } y = v_{out}, \text{ for some } v \in V \\ in(v), & \text{if } x = v_{in}, \text{ for some } v \in V, \text{ and } y = t \\ 1, & \text{if } x = u_{out} \text{ and } y = v_{in} \text{ from some } u, v \in V \text{ s.t. } (u, v) \in E \\ 0, & \text{if } x = u_{out} \text{ and } y = v_{in} \text{ from some } u, v \in V \text{ s.t. } (u, v) \notin E \end{cases}$$

We now show that the function f_R is a flow. It obviously satisfies the capacity constraint. For the conservation constraint, consider node v_{out} of $\mathcal{F}$. By the definition of f_R, $out(v)$ edges out of v carry a flow of 1 and all other edges out of v carry a flow of 0, so $\sum_{u \in V} f_R(v_{out}, u_{in}) = out(v)$. Also by definition of f_R, $f_R(s, v_{out}) = out(v)$. Therefore, $\sum_{u \in V} f_R(v_{out}, u_{in}) = f_R(s, v_{out})$. Since v_{out} has only one incoming edge, namely that from s, the conservation constraint is satisfied at v_{out}. A similar argument shows that the conservation constraint is also satisfied at v_{in}. Therefore f_R is a flow. It is a maximum flow because, by definition of f_R, every edge out of s is saturated. Finally, $\mathcal{V}(f_R) = \sum_{v \in V} f_R(s, v_{out}) = \sum_{v \in V} out(v)$.

We have therefore shown that if the given pairs are realizable then the value of the maximum flow of $\mathcal{F}$ is $\sum_{v \in V} out(v)$. So, we have the following algorithm to determine if the given degree-pairs for a set of nodes V are realizable:

<u>REALIZABLEDEGPAIRS(V, *in*, *out*)</u>
construct the flow network $\mathcal{F}$ with edge capacities as described above
use the Ford-Fulkerson algorithm to determine an integral maximum flow f of $\mathcal{F}$
if $\mathcal{V}(f) = \sum_{v \in V} out(v)$ **then**
  construct $R = (V, E)$, where $E = \{(u, v) : u, v \in V \text{ and } f(u_{out}, v_{in}) = 1\}$
  return "realizable by graph R"
else
  return "not realizable"

The correctness of this algorithm follows immediately by (3). Its running time is dominated by the time to determine the maximum flow of $\mathcal{F}$. Let n be the number of nodes in V. Since $\mathcal{F}$ has $O(n^2)$ edges and the value of the maximum flow of $\mathcal{F}$ is $O(n^2)$, the running time of the algorithm is $O(n^4)$.

Answer to Question 3. This is a max-this equals min-that kind of result, and we have seen two such results, closely related to each other: max-flow-min-cut for flow networks and max-matching-min-vertex-cover for bipartite graphs.

<u>Irreducible</u> set of tokens means that no two tokens in the set share a row or column; this sounds like matching (no two edges in the matching share a node). <u>Full</u> set of rows or columns means that every token is "touched" by a row or column of the set; this sounds like a vertex cover (the nodes in the vertex cover "touch" every edge). So, the result we are being asked to prove sounds like bipartite graph max-matching-min-vertex-cover in disguise.

Consider an $m \times n$ grid and a set of tokens on the grid, and let $G = \big((X, Y), E\big)$ be the bipartite graph defined as follows:

- X is the set of rows $\{\mathrm{R}1, \ldots, \mathrm{R}m\}$ of the grid.
- Y is the set of columns $\{\mathrm{C}1, \ldots, \mathrm{C}n\}$ of the grid.
- There is an edge $\{\mathrm{R}i, \mathrm{C}j\}$ if and only if there is a token in square (i, j) of the grid.

Thus, a set of tokens is a set of edges of G, and a set of rows or columns is a set of nodes of G. An irreducible set of tokens is a set of edges of G no two of which share a node (row or column); therefore an irreducible set of tokens is a matching of G, and a maximum cardinality irreducible set of tokens is a maximum matching of G. A full set of rows or columns is a set of nodes of G that "touch" all the edges (tokens); therefore a full set of rows or columns is a vertex cover of G, and a minimum cardinality full set of rows or columns is a minimum vertex cover of G.

We have proved that in every bipartite graph G, the cardinality of a maximum matching is equal to the cardinality of a minimum vertex cover. Therefore, the maximum cardinality of an irreducible set of tokens is equal to the cardinality of a minimum full set of rows or columns.

(We can now find a minimum full set of rows or columns by finding a minimum vertex cover of G, and a maximum irreducible set of tokens by finding a maximum matching of G. We have seen how to do both of these in bipartite graphs, such as G, using the maximum flow algorithm.)