**15-451/651 Algorithm Design & Analysis, Fall 2024**

# Recitation #7

**Objectives**

- Understand the faster flow algorithms: Edmonds-Karp and Dinic's.
- Understand and apply Minimum-cost flows.

## Review

### Algorithm Runtimes

1. Ford-Fulkerson (FF): $O(mF)$
2. Edmonds-Karp (EK): $O(m^2n)$
3. Dinic's: $O(mn^2)$

## Recitation Problems

1. **(Entering flow state)**

   (a) True or False?

      i. Ford-Fulkerson runs in polynomial time on unit-capacity graphs.

      ii. Dinic's algorithm is asymptotically faster than Edmonds-Karp for sparse graphs.

      iii. Finding a blocking flow takes $O(n^2)$ time.

   (b) In lecture, the three flow algorithms we presented built on top of each other. Name one main difference between:

      i. Ford-Fulkerson and Edmonds-Karp

      ii. Edmonds-Karp and Dinic's

   (c) What type of search is typically used to solve the following (DFS, BFS, Either)?

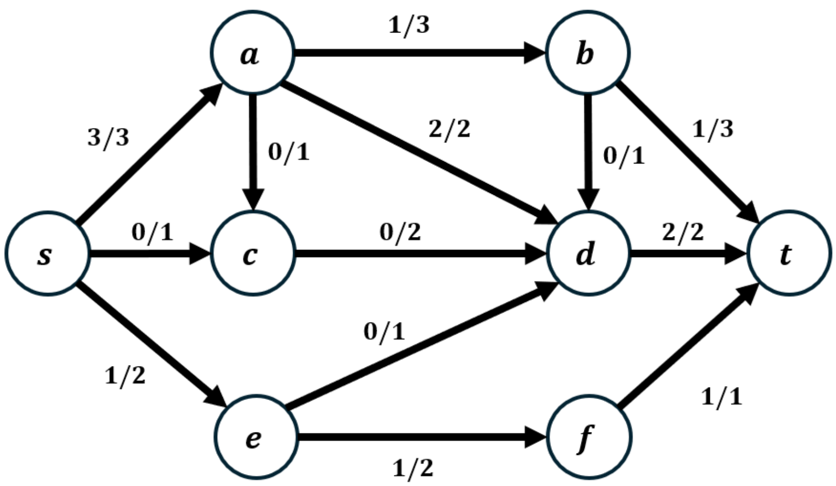
i. An iteration of Ford-Fulkerson

ii. An iteration of Edmonds-Karp

iii. Constructing a layered graph

iv. Finding blocking flow in layered graph in Dinic's algorithm

(d) Consider the following network $G$ with a (non-maximum) feasible flow $f$:




i. Draw the layer/level/admissible graph of the residual graph $G_f$. What is the current $s$-$t$ distance in $G_f$?

ii. Calculate a blocking flow on the layer graph from Part (i) and use it to augment the flow.

2. **(Oral homework scheduling)** You've been hired to help the 451 TAs schedule their oral sessions. There are $n$ TAs, and TA $i$ has $s_i$ slots that they need to book a room for. There are $m$ available room bookings, and each TA $i$ has a list $L_i$ of which room bookings $\{1, 2, \ldots, m\}$ would be suitable for them. Since there is a shortage of room bookings, however, the department has started to sell the bookings for money! The $j^{\text{th}}$ room booking costs $c_j$ dollars. Your job is to find a way to schedule all of the oral sessions for the minimum amount of money, or report that it is not possible.

3. **(Super fast matching - Optional)** In lecture, we saw that Dinic's algorithm runs in $O(n^2 m)$ time on any graph, but on some graphs it runs even faster! For instance, in a *unit-capacity* network, where every edge has capacity one, we proved that Dinic's algorithm runs in time $O(m\sqrt{m})$. In this problem we will take this one step further.

   Suppose our graph has one additional restriction (we still keep the unit-capacity restriction): The net flow across every vertex (except $s$ and $t$) can be at most one. This is equivalent to saying that every vertex other than $s$ and $t$ has either indegree one or outdegree one (but not necessarily both). Such a network is called a "unit network".

   (a) Prove that in a unit network, the number of blocking flows required to find a max flow is at most $O(\sqrt{n})$. (Hint: use a similar argument to the one in lecture for unit-capacity graphs)

   (b) Prove that we can solve the Bipartite Matching problem in $O(m\sqrt{n})$ time.