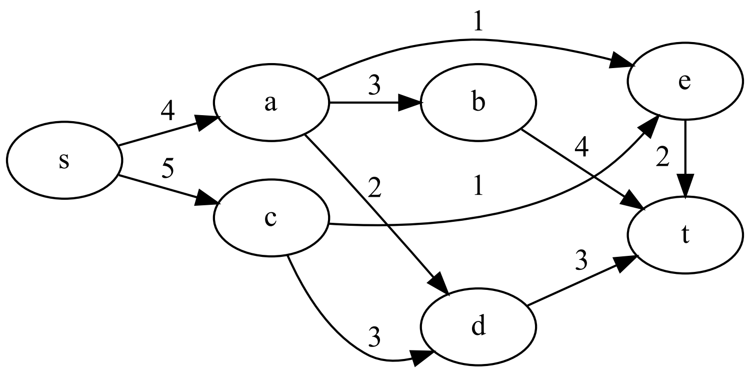
CS41 Lab 8: Network Flow

In typical labs this semester, you'll be working on a number of problems in groups of 3-4 students. You will not be handing in solutions; the primary purpose of these labs is to have a low-pressure space to discuss algorithm design. However, it will be common to have some overlap between lab exercises and homework sets.

The lab this week focuses on network flow. The purpose of this lab is to gain practice thinking about the Ford-Fulkerson algorithm and using FF to solve other problems using **reductions**.

Try to solve your algorithm design problems in lab this week by **reducing to network flow**. You should **use the Ford-Fulkerson algorithm as a subroutine** rather than making an entirely new algorithm of your own.

1. **Maximum flow.** Find the maximum flow from s to t and the minimum cut between s and t in the network below. Show the residual graph at intermediate steps as you build the flow. What is the value of the maximum flow?



2. **Bipartite Matching**

   Recall that a *bipartite graph* $G = (V, E)$ is an undirected graph where the vertices can be partitioned into $X, Y \subset V$, where $X \cap Y = \emptyset$ and $X \cup Y = V$, such that all edges have one end in X and the other in Y, that is $\forall (u, v) \in E$, $u \in X \wedge v \in Y$. A *matching* M in G is a subset of the edges $M \subseteq E$ such that each node appears in at most one edge of M. The *bipartite matching* problem is to find the matching M in bipartite graph G with maximum possible size. Give an efficient algorithm for bipartite matching.

3. **Evacuation Congestion.** (K& T 7.14) When handling natural disasters like hurricanes or wildfires, state and federal agencies need to plan how to evacuate potentially thousands of people. One major challenge is that evacuees might take the same routes to evacuation points. The resulting road congestion can hinder evacuation. In this problem, you will design an algorithm to help agencies minimze this congestion. The input to this problem is as follows;

   - a directed graph $G = (V, E)$.
   - a collection of populated vertices $A \subset V$.
   - a collection of safe vertices $B \subset V$.

Assume A and B are disjoint. In case of an emergency, we want evacuation routes from the populated vertices to the safe vertices. A set of evacuation routes is defined as a set of paths in G so that (i) each node in A is the tail of one path, (ii) the last node in each path is in B, and (iii) the paths do not share any edges. Such a set of paths gives a way for the occupants of the populated vertices to *escape* to B without overly congesting any edge in G.

(a) Given G, A, and B, show how to decide in polynomial time whether such a set of evacuation routes exists.

(b) Suppose we have the same problem as before, but now condition (iii) is that paths do not share any *vertices*. Show how to decide in polynomial time whether such a set of evacuation routes exists.

(c) Provide an example input G, A, B such that the answer is yes to the question in (a) but no to the question in (b).

4. **Advertising contracts** (K& T 7.16)

Back in the euphoric early days of the Web, people liked to claim that much of the enormous potential in a company like Yahoo! was in the "eyeballs"—the simple fact that millions of people look at its pages every day. Further, by convincing people to register personal data with the site, a site like Yahoo! can show each user an extremely targeted advertisement whenever the user visits the site, in a way that TV networks or magazines couldn't hope to match. So if a user has told Yahoo! that she is a 21-year-old computer science major at Swarthmore, the site can present a banner ad for apartments in Philadelphia suburbs; on the other hand, if she is a 50-year-old investment banker from Greenwich, CT, the site can display a banner ad pitching luxury cars instead.

But deciding on which ads to show to which people involves some serious computation behind the scenes. Suppose that the managers of a popular site have identified k distinct *demographic groups* $G_1, G_2, \ldots, G_k$. (Some may overlap.) The site has contracts with m different advertisers to show a certain number of copies of their ads to users of the site. Here's what a contract with the i^{th} advertiser looks like:

- For a subset $X_i \subseteq \{G_1, \ldots, G_k\}$ of the demographic groups, advertiser i wants ads shown only to users who belong to at least one of the groups listed in X_i.
- For a number r_i, advertiser i want its ads shown to at least r_i users each minute.

Consider the problem of designing a good *advertising policy* — a way to show a single ad to each user of the site. (Imagine a world where each user saw only *one* ad per site.) Suppose at a given minute, there are n users visiting the site. Because we have registration about each of the users, we know that user j belongs to a subset U_j of the demographic groups.

The problem is: is there a way to show a single ad to each user so that the site's contracts with each of the m advertisers is satisfied for this minute?

Give an efficient algorithm to decide if this is possible, and if so, to actually choose an ad to show each user.