OA 4202, Homework 4

Nedialko B. Dimitrov

1. (AMO)

   6.32. Airline scheduling problem. An airline has p flight legs that it wishes to service by the fewest possible planes. To do so, it must determine the most efficient way to combine these legs into flight schedules. The starting time for flight i is a_i and the finishing time is b_i. The plane requires r_{ij} hours to return from the point of destination of flight i to the point of origin of flight j. Suggest a method for solving this problem.

   An acceptable solution:

   This is exactly the tanker scheduling problem. For each flight leg, say flight leg i, we create two nodes: "Start i" and "End i". We connect "Start i" to "End i" using an arc with lower bound 1, meaning that one plane is required to complete this flight leg.

   We connect two legs, if the same plane can be used for both flights. In other words, we connect "End i" to "Start j" if $b_i + r_{ij} \leq a_j$.

   Finally, we create one super-source node and one super-sink node. We connect the super-source to the start of each leg, and the super-sink to the end of each leg. These arcs indicate putting a plane into service or taking a plane out of service.

   Then, like in the tanker scheduling problem, we can do the following:

   (a) Find a feasible flow/schedule using Producer/Consumer.
   (b) Push as much flow back from the super-sink to the super-source, to find a flow/schedule that uses the minimum number of planes.

---

2. (AMO)

   6.3. A commander is located at one node p in an undirected communication network G and his subordinates are located at nodes denoted by the set S. Let u_{ij} be the effort required to eliminate arc (i, j) from the network. The problem is to determine the minimal effort required to block all communications between the commander and his subordinates. How can you solve this problem in polynomial time?

   An acceptable solution:

   We add a super-source and a super-sink to the graph. We connect the super-source to the commander's node, p, using an arc with infinite capacity. Similarly, we connect each subordinate's node to the super-sink using an arc of infinite capacity.

   The minimum cut between the super-source and the super-sink gives the minimum-effort edges to disconnect the commander from the subordinates. The reason the commander and

the subordinates are on different sides of the cut is because the infinite capacity arcs we added would never cross the cut.

---

3. (AMO)

   6.1. Dining problem. Several families go out to dinner together. To increase their social interaction, they would like to sit at tables so that no two members of the same family are at the same table. Show how to formulate finding a seating arrangement that meets this objective as a maximum flow problem. Assume that the dinner contingent has p families and that the ith family has $a(i)$ members. Also assume that q tables are available and that the jth table has a seating capacity of $b(j)$.

   An acceptable solution:

   Recall that a bipartite graph is a graph $G = (V, E)$ where we are able to split the vertices into two mutually exclusive sets A and B, so that all the edges go between A and B, and no edges are inside of A or inside of B. We saw how to identify if a graph is bipartite with the "professional wrestler" problem a few homeworks ago.

   We create the following bipartite graph. On the "left side" we create one node for each family. On the "right side" we create one node for each table. We connect each family's node to each table's node with an edge of capacity 1. This edge means that only one family member can sit at the table.

   On top of this bipartite graph, we add one "super-source" node and one "super-sink" node. We hook up the super-source to each family... if the family has has size $a(i)$, we give the edge capacity $a(i)$. These edges mean that we have to seat $a(i)$ people in this family.

   Similarly, we hook up each table to the super-sink... if the table has seating capacity $b(j)$, we give the edge capacity $b(j)$. This means that the table can handle seating at most $b(j)$ people.

   We can do a maximum-flow computation from the super-source to the super-sink. If the arc from the super-source to each family's node is saturated, we know we can seat the family under the required condition. The actual flow values give us the seating arrangement. If the arc from a table to the super-sink is saturated, it means that the table is full of people.

---

4. You are working for a group that has been monitoring a terrorist organization. Your group has intercepted enough communications and gathered enough intelligence to reconstruct the terrorist organization's social network... you know which terrorist is friends with which other terrorist. There is some possibility that this terrorist organization may soon split into two organizations. How would you find a likely split?

   For an alternate wording of the same problem, consider:

   6.11. Suppose that we wish to partition an undirected graph into two components with the minimum number of arcs between the components. How would you solve this problem?

   An acceptable solution:

Consider the graph of the terrorist social network, where each node is a terrorist and an edge between two terrorists exists if they are friends. Finding the split of this graph with the minimum number of edges between the two sides is a likely split of the organization.

We can do this by running $n-1$ min-cut problems. Pick a node, lets call it a. We'll run a min-cut problem between a and every other node of the graph... that's $n-1$ min-cut problems. The minimum of these $n-1$ cuts gives us the most likely split.

You can think of these $n-1$ min-cut problems as trying to "guess" one node on each side of most likely split. We know a has to be on one side, but, we don't know exactly which nodes go on the other. That's why we run $n-1$ problems. We know one of these guesses is going to be correct, since some node has to be on the other side from a.

---

5. You own your own temp-agency. Every month you have a pool of temporary workers and a pool of open jobs to be filled with temporary workers. Based on a worker's resume, you are able to discern the jobs the worker is eligible to fill. Specifically, worker i is eligible for the jobs in the set $N(i)$. Your agency receives \$5 for every feasible match between a worker and a job that you make. Of course, you can match each worker to at most one job, and each job can be matched to at most one worker. How would you make the most pairings, to maximize your temp-agency's revenue?

   An acceptable solution:

   Create a bipartite graph. On the left we have workers, on the right we have jobs. Each worker, i, is connected to the jobs he is eligible for, $N(i)$, using an arc with capacity 1.

   We add one super-source and one super-sink. We connect the super-source to each worker using an arc with capacity 1. This arc means that each worker can be matched to at most one job. We connect each job to the super-sink also using an arc with capacity 1. This arc means that each job can be matched to at most one worker.

   The maximum flow between the super-source and the super-sink computes the maximum number of pairings between workers and jobs. The flow values on the edges give the pairings between workers and jobs.

---

6. (AMO)

6.2. Nurse staff scheduling (Khan and Lewis [1987]). To provide adequate medical service to its constituents at a reasonable cost, hospital administrators must constantly seek ways to hold staff levels as low as possible while maintaining sufficient staffing to provide satisfactory levels of health care. An urban hospital has three departments: the emergency room (department 1), the neonatal intensive care nursery (department 2), and the orthopedics (department 3). The hospital has three work shifts, each with different levels of necessary staffing for nurses. The hospital would like to identify the minimum number of nurses required to meet the following three constraints: (1) the hospital must allocate at least 13, 32, and 22 nurses to the three departments (over all shifts); (2) the hospital must assign at least 26, 24, and 19 nurses to the three shifts (over all departments); and (3) the minimum and maximum number of nurses allocated to each department in a specific shift must satisfy the following limits:

		Department 1	Department 2	Department 3
Shift	1	(6, 8)	(11, 12)	(7, 12)
	2	(4, 6)	(11, 12)	(7, 12)
	3	(2, 4)	(10, 12)	(5, 7)

Suggest a method using maximum flows to identify the minimum number of nurses required to satisfy all the constraints.

(Hint: While the way to solve this problem bears some resemblance to the tanker problem, the graph for this problem is *not* like the one for tanker problem we discussed in class. That is because each nurse can work at most one of the three shifts, and cannot simply continue from one shift to the next. If you need a further hint, see AMO Application 6.3.)

An acceptable solution:

We create a bipartite graph: Shifts are on the left, departments are on the right. We connect each shift node to each department node using an arc with the corresponding upper/lower bounds on the flow (these are given in the table in the problem).

To the bipartite graph, we add a super-source and a super-sink. We connect the super-source to each shift using an arc with a lower bound equal to the minimum number of nurses for the shift. We connect each department to the super-sink using an arc with a lower bound equal to the minimum number of nurses that must be allocated to the department.

We can solve the problem from here in a way that is similar to how we solved the tanker problem. First, we use one max-flow computation to find a feasible flow. We do this by first adding an additional ark from the sink back to the source, then applying our Producer/Consumer. Then, we remove the additional temporary arc from the sink back to the source.

We adjust the feasible flow's residual graph in the same way that we adjusted it for the tanker problem... so that the lower bounds remain satisfied. Then, we push as much flow from the sink back to the source as possible to complete the problem solution.

---

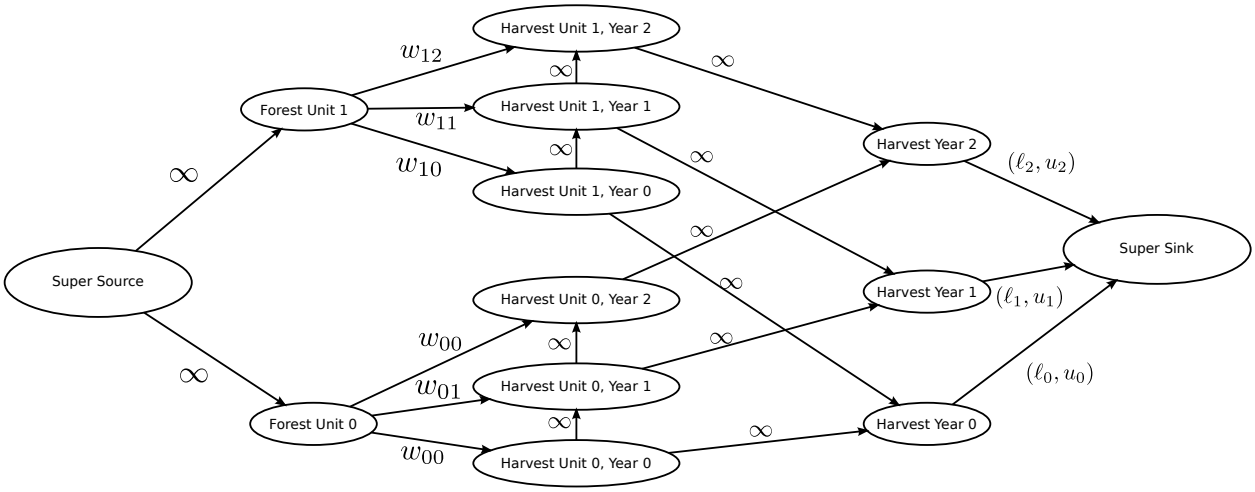
7. (AMO) You are in charge of a wood product company, and you would like to maximize the

total wood yield of the forests owned by your company. Suppose your company controls two forests and wants to identify the best cutting schedule over a planning horizon of three years. Your company only harvests mature trees and studies performed by the company predict that w_ij wood units will mature in forest i on year j. If the wood is available for harvesting one year, but we choose not to harvest it, then it is also available for harvesting in the following years. In other words, trees grown up to harvesting size accrue over time. The company has also performed economic predictions that show it should harvest at least l_j wood units in year j. The economic predictions also show that if the company harvests more than u_j wood units in year j, the market would be flooded and wood prices would bottom out.

Formulate the problem of determining a schedule with maximum wood yield as a network flow problem.

An acceptable solution:

We describe the general graph structure using an example with 2 forest units and a planning horizon of 3 years. All of the numbers on the arcs are upper bound capacities, except the ones on the right, which have both lower and upper bounds.



Each forest unit has its own node. Each combination of (forest unit, year) has its own node. A forest unit node i is connected to a (forest unit, year) node (i, j) using an arc with capacity w_{ij}. This signifies that w_ij tons of wood mature that year and are ready for harvesting.

Each (forest unit, year) node (i, j) is connected to the node $(i, j + 1)$ by an infinite capacity arc. This signifies that unharvested wood accrues and is available for harvesting the following years, if necessary.

Each harvesting year has its own node. Each (forest unit, year) node (i, j) is connected to the node for harvesting year j using an arc of infinite capacity. This signifies that if wood is available for harvesting, we can harvest it.

Each harvesting year node j is connected to the super-sink using an arc with lower bound l_j and upper bound u_j. These signify the upper and lower bounds on the tons of woods that should be harvested that year.

To complete the graph structure, a super-source is connected to each forest unit using an infinite capacity arc.

A feasible flow on this graph gives us a feasible harvesting plan.