MATH 566: Lecture 2 (08/22/2024)

Today:
* Formal definition of MCF
  - SP, MF as cases of MCF
* circulation, transportation, seat-sharing problem

We now specify the mathematical model for the MCF problem. Even though we introduce the model now, we will not solve the problems using this model – we will present efficient algorithms that exploit the network structure better (which the model does not do).

Notation

→ G for "graph"

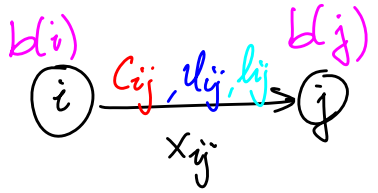
$G = (N, A)$; G: directed network, N: set of nodes, A: set of directed arcs.

$|N| = n$ (# nodes), $|A| = m$ (# arcs).

c_{ij} : unit cost on arc $(i,j) \in A$ (can be $>0, <0$, or $=0$)

u_{ij} : capacity of $(i,j) \in A$ (>0) → upper bound

l_{ij} : lower bound of $(i,j) \in A$ (≥ 0).



By default, $l_{ij} = 0$, unless mentioned otherwise.

$b(i)$: supply/demand at node $i \in N$

$> 0 \Rightarrow$ i is a supply node

$< 0 \Rightarrow$ i is a demand node → demand of $-b(i)$ at node i

$= 0 \Rightarrow$ i is a transshipment node

$b(i), c_{ij}, l_{ij}, u_{ij}$: data (known with certainty). These parameters do not depend on x_{ij}'s, which are variables.

Goal: Find flow x_{ij} $\forall (i,j) \in A$ such that bounds are satisfied, supply/demand is "met" at each node $i \in N$, and overall cost is minimum.

We assume $\sum_{i \in N} b(i) = 0$ by default, i.e., total supply = total demand.

Here is the optimization model for the min-cost flow problem:

$$\min \left\{ \sum_{(i,j) \in A} c_{ij} x_{ij} \right\} \quad \text{(1)}$$

(minimize; total cost)

subject to (s.t.)

$$\sum_{(i,j) \in A} x_{ij} - \sum_{(j,i) \in A} x_{ji} = b(i) \quad \forall\, i \in N \quad \text{(2)}$$

(for all)

outflow − inflow = supply/demand

$$l_{ij} \le x_{ij} \le u_{ij} \quad \forall\, (i,j) \in A \quad \text{(3)}$$

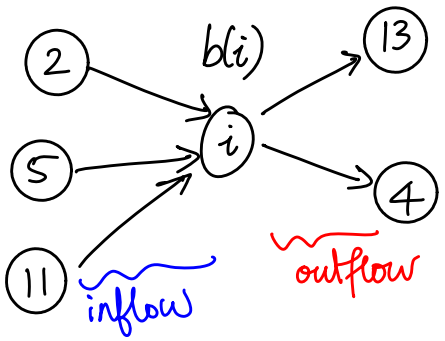
Mathematical notation: $\in$: "element of", $\forall$: "for all", $\sum$: sum.

In words, we want to minimize the total cost (1), subject to meeting the supply/demand constraints at each node (2), and the bounds on flow in each arc (3).

This is a linear optimization problem (or a linear program, LP). But we won't solve these instances the same way we solve LPs. These problems are simpler than the general LPs — the underlying network structure allows one to design efficient algorithms, which run faster than default algos to solve LPs!

(1) models the total cost incurred. On arc (i,j), one unit of flow incurs a cost of c_{ij}. Hence x_{ij} units incur $c_{ij}x_{ij}$. Summing up all such cost terms (over all arcs in the arc set A) gives the total cost. The goal is to find a flow, i.e., all x_{ij} values, that minimizes the total cost.

The flow must satisfy constraints (2) and (3). (2) are the flow balance (or mass balance) constaints, which you could think of as "conservation of flow (or mass)."



$x_{i,13} + x_{i,4}$ } outflow

$x_{2i} + x_{5i} + x_{11,i}$ } inflow

$b(i)$: supply/demand

outflow − inflow = supply/demand

There is no flow "lost" or "appearing out of the blue". In words, "what comes in + what is produced or used = what goes out".

The flow must honor the lower and upper bounds on each arc, as specified by constraints (3). The default value for all l_{ij} is zero. If u_{ij} is not specified, it is taken as $+\infty$ (some large number in practice).

We also assume here that $\sum_{i \in N} b(i) = 0$, i.e., total supply is equal to total demand. There are generalizations where this condition might not hold, when (2) may not be written as '=' for all i.

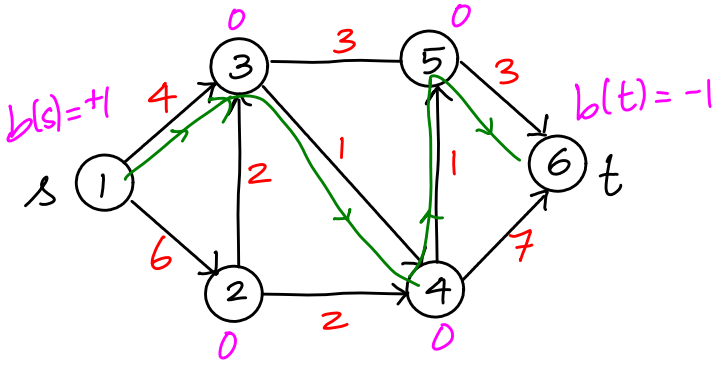
We now show that the shortest path and the max flow problems are special cases of the min cost flow problem.

(*) When "formulating" or "modeling" a problem as a network flow problem, you need to specify the network $G=(N,A)$, i.e., what is the node set, the arc set, and what are the associated parameters $(b(i), l_{ij}, u_{ij}, c_{ij})$. ($M$ some big >0 number in practice)

Default values: $b(i)=0$, $l_{ij}=0$, $c_{ij}=0$, $u_{ij}=+\infty$.

Shortest Path Problem as a min-cost flow Problem

Consider the example for shortest path from Lecture 1.

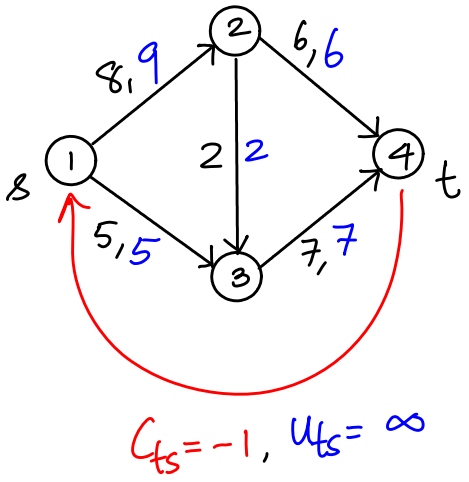


Use same network $G=(N,A)$ as in the shortest path problem. We set $b(s)=+1$, $b(t)=-1$, $b(i)=0$ $\forall i \neq s,t$, $i \in N$. Then we set $l_{ij}=0$, $u_{ij} \geq 1$ $\forall (i,j) \in A$.

Solving the MCF instance here is equivalent to sending one unit of flow from s to t at the least cost. Think of sending one car from the origin to destination. To detail the argument, we can show $x_{ij}=1$ or $x_{ij}=0$ here (there are some deeper assumptions critical here). Then argue that the $(i,j) \in A$ with $x_{ij}=1$ precisely gives the shortest path.

Max Flow as special case of MCF

In the previous instance (shortest path), we did not have to add any extra nodes or arcs. To model the max flow problem as a min cost flow problem, we add a single extra arc.



Add (t,s) to A, with $c_{ts}=-1$, $u_{ts}=+\infty$.

Set $l_{ij}=0 \quad \forall (i,j)\in A$ (including l_{ts})

$c_{ij}=0 \quad \forall (i,j)\in A \neq (t,s)$.

$b(i)=0 \quad \forall i \in N$.

Since $C_{ts}=-1$ (we could set it to any value <0, while keeping all other $C_{ij}=0$), min cost flow will try to send as much flow as possible on arc (t,s). And all flow on (t,s) enters node s. Further, this flow is also equal to the net flow out of node t. In other words, the flow on (t,s) is indeed the maximum flow from s to t. Since $u_{ts}=\infty$, the value of the max flow is determined by the u_{ij} values of the original network, as it should be.

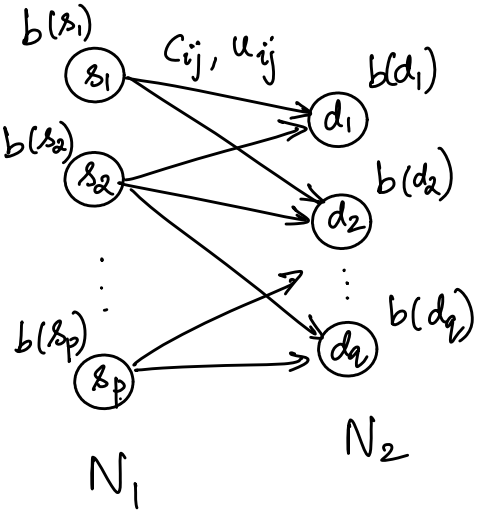
Notice that the flow just circulates around the network here. There is no flow coming in or going out at any of the nodes. This is an instance of the **circulation problem**, which is the min cost flow problem with $b(i)=0$ for all nodes i. This is the fourth network flow problem class (after shortest path, max flow and min cost flow).

→ min cost circulation, to be exact

A typical example of the circulation problem is the scheduling of airplanes operated by a firm, e.g., Alaska Airlines. Each city/destination served is a node, and the total number of planes remains same within the network. If a service is required between, say, Seattle and Spokane, the l_{ij} for that arc is set to 1, ensuring one plane flies from Seattle to Spokane.

(5) Transportation problem

This is a special case of mincost flow where the node set $N = N_1 \cup N_2$, ("union") with N_1 being supply nodes and N_2 being demand nodes. Hence, $b(i) > 0 \ \forall i \in N_1$, and $b(j) < 0 \ \forall j \in N_2$. Further, all arcs $(i,j) \in A$ have $i \in N_1$ and $j \in N_2$. The arcs have c_{ij}, l_{ij}, and u_{ij} specified.



Notice that there are no arcs within N_1 or within N_2.

We assume that $\sum_{i \in N_1} b(i) = -\sum_{j \in N_2} b(j)$,

i.e., total supply = total demand.

Such an instance is a **balanced transportation problem**. In the unbalanced case, there might be penalties for unmet demand or for unused supply.

There are no transshipment nodes in the standard transportation problem. Indeed, if there were a node with zero supply within N_1, we could remove it without changing the main problem. Similar is the case with a zero demand node in N_2.

The default example is that of shipping of a commodity between warehouses and retailers, with the former set forming the supply nodes, and the retailers forming the demand nodes. If there is an option to ship the commodity from warehouse i to retailer j, arc (i,j) is included in the network. For instance, if there is a truck available to ship items from the Seattle warehouse to the Vancouver store; the capacity and cost associated with the truck are modeled as u_{ij} (and l_{ij}, if there is a minimum), and c_{ij}.

Read section 1.3 on modeling applications as various network flow problems. We will discuss one such problem — the seat sharing problem — in detail in the next lecture. A handout on this problem is posted on the course web page.

Seat Sharing Problem

(AMO 1.8, page 21) Several families are planning a shared car trip on scenic drives in the Cascades in Washington. To minimize the possibility of any quarrels, the organizers of the trip want to assign individuals to cars so that **no two members of a family are in the same car**. Formulate this problem as a network flow problem.

Let there p families, with b_i members for $1 \le i \le p$. And let there be q cars, with u_j seats, for $1 \le j \le q$. We start with a node for each family and a node for each car. Let $N_1 = \{f_1, \dots, f_p\}$ be the set of family nodes and $N_2 = \{c_1, \dots, c_q\}$ be the car nodes.

This situation appears to be close to a transportation problem setting. But there is a catch — each car has some set of seats that need not **all** be filled! In that sense, we cannot model the # seats of car c_j as its demand. We need to somehow model it as a capacity of the node.

More in the next class...