Lecture Notes CS:5350 Design and Analysis of Algorithms
Lecture 3: September 3, 2019
Scribe: Anthony Cantor

1 Formulating the LP example in Standard Form

Previously we examined a Linear Programming example from Wikipedia involving parameters determined by farming for wheat and barley. We have choice variables x_w and x_b that represent the amount (in km^2) of wheat and barley planted, respectively. Our objective function depends on the prices S_w and S_b of wheat and barley (measured in $\$/km^2$), respectively. We are limited to the quantity L (in km^2) of land, F (in kg) of fertilizer, and P (in kg) of pesticide. Finally, we have parameters that specify the required amounts of fertilizer and pesticide needed by the crops:

- F_w and F_b of fertilizer (in kg/km^2) is needed for wheat and barley, respectively
- P_w and P_b of pesticide (in kg/km^2) is needed for wheat and barley, respectively

We wanted to maximize our objective function $S_w x_w + S_b x_b$ subject to our constraints

$$\begin{aligned} x_w + x_b &\leq L \\ P_w x_w + P_b x_b &\leq P \\ F_w x_w + F_b x_b &\leq F \\ x_w, x_b &\geq 0 \end{aligned}$$

We can translate this into vector form, as follows. We want to maximize

$$\begin{bmatrix} S_w \\ S_b \end{bmatrix}^T \begin{bmatrix} x_w \\ x_b \end{bmatrix}$$

subject to

$$\begin{bmatrix} 1 & 1 \\ P_w & P_b \\ F_w & F_b \end{bmatrix} \begin{bmatrix} x_w \\ x_b \end{bmatrix} \leq \begin{bmatrix} L \\ P \\ F \end{bmatrix}$$

and

$$\begin{bmatrix} x_w \\ x_b \end{bmatrix} \geq 0$$

This is now in the standard form for an LP problem, which is:

$$\begin{array}{cl} \text{maximize} & \mathbf{c}^T\mathbf{x} \\ \text{subject to} & A\mathbf{x} \leq \mathbf{b} \\ & \text{and } \mathbf{x} \geq 0 \end{array}$$

where $\mathbf{c}$ and $\mathbf{x}$ are n-dimensional vectors, $\mathbf{b}$ is an m-dimensional vector of constraints, and A is an $m \times n$ matrix.

LP could have a minimization objective function. It could also have a $\leq$, $\geq$, or $=$ constraint, and all of these variants can be transformed into the standard form, without loss of generality. The only requirement for an LP is that the objective function and constraints are all linear.

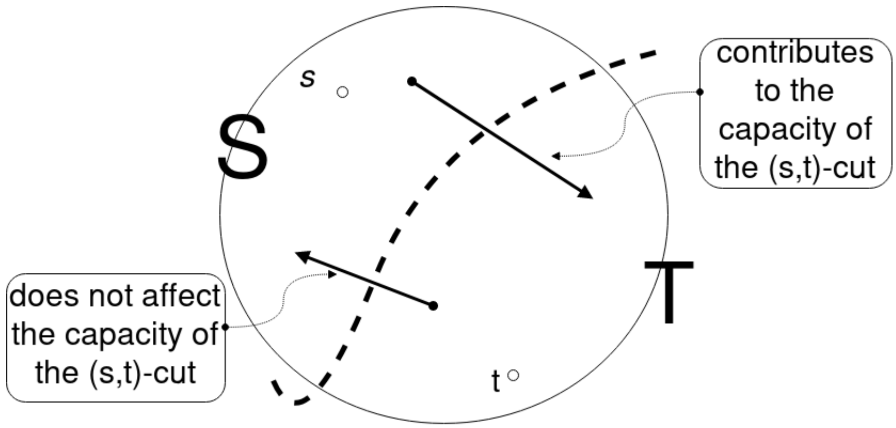

Figure 1: This diagram hints at a visual intuition for the definition of the capacity $||S, T||$ of the (s, t)-cut: the example edges shown (solid black arrows) cross the "cut line" (indicated by the dashed line) determined by the partition elements S and T.

2 LP Formulation of MaxFlow

We formulate MaxFlow as an LP problem as follows. We have choice variables $f_{u \to v}$ for each $u \to v \in E$. We want to maximize $\sum_u f_{s \to u} - \sum_w f_{w \to s}$ subject to

- Flow conservation: $\sum_u f_{v \to u} = \sum_w f_{w \to v}$ for each $v \in V \setminus \{s, t\}$
- Feasibility: $0 \leq f_{u \to v}$ and $f_{u \to v} \leq c_{u \to v}$ for every edge $u \to v \in E$

So in a graph with n vertices and m edges:

- we have $(n - 2) + 2m$ constraints
- we have m variables

This tells us the "size" of the LP instance.

We will consider combinatorial algorithms for the MaxFlow problem despite the fact that MaxFlow can be reduced to LP and we know that LP can be solved in polynomial time. Some of our combinatorial algorithms have better performance than general LP algorithms, and will also provide more insights into the structure of the problem.

3 The MinCut Problem

In order to define the **MinCut** problem we must first define an (s, t)-cut:

Definition 1 *An (s, t)-**cut** is a set partition of V into sets S and T such that $s \in S$ and $t \in T$. The **capacity** of an (s, t)-cut, denoted $||S, T||$, is equal to $\sum_{u \in S, v \in T} c(u \to v)$*

See Figure 1 for a better understanding of how various edges are treated by this definition.

Definition 2 *The **MinCut** problem is defined by its input and output specifications as follows. The **Input** is a directed special capacity graph $G = (V, E)$, vertices $s, t \in V$, and function $c : E \to \mathbb{R}_{\geq 0}$. The **Output** is an (s, t)-cut with minimum capacity.*

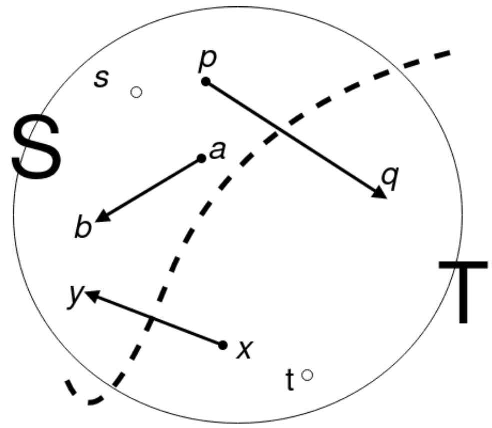

Figure 2: A depiction of the relationship between (S, T) and the edges $a \to b$, $p \to q$, and $x \to y$

4 MaxFlow-MinCut Theorem (weak version)

Now we show an important relationship between feasible flows and cuts.

Theorem 3 *Let f be an arbitrary feasible (s,t)-flow and (S,T) be an arbitrary (s,t)-cut. Then $|f| \le ||S, T||$.*

Proof: The net outflow from s is

$$\begin{aligned}
|f| &= \sum_u f_{s\to u} - \sum_w f_{w\to s} \\
&= (\sum_u f_{s\to u} - \sum_w f_{w\to s}) \\
&+ (\sum_{v\in S\setminus\{s\}} (\sum_u f_{v\to u} - \sum_w f_{w\to v})) \qquad \text{(by flow conservation)} \\
&= \star
\end{aligned}$$

To understand the next step in the equations below, take note of how each particular edge contributes to the sum according to its relationship with the partition elements S and T:

- For all edges $a \to b \in E$ with $a, b \in S$ (as in Figure 2), the net contribution of $a \to b$ to the sum is 0.
- For all edges $p \to q \in E$ with $p \in S$, $q \in T$ (as in Figure 2), we get a positive $f_{p\to q}$ contribution to the sum.
- For all edges $x \to y \in E$, $x \in T$, $y \in S$ (as in Figure 2), we get a negative contribution to the sum.

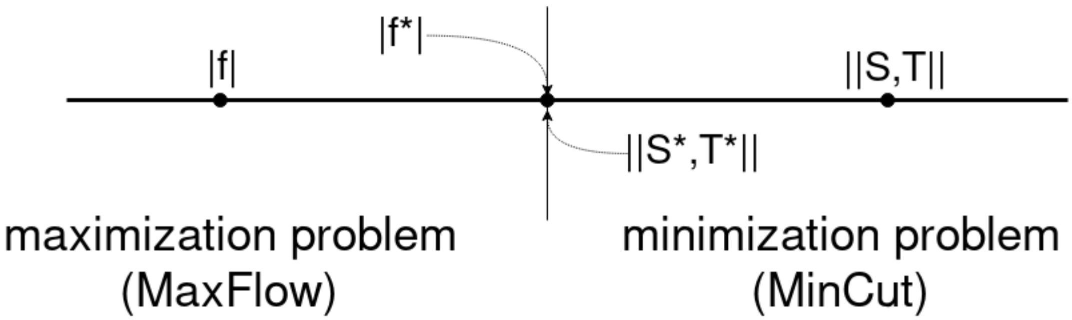

Figure 3: a depiction of the "duality" view obtained from theorems 3 and 4.

Therefore we have:

$$
\begin{aligned}
\star &= \sum_{p\to q\in E, p\in S, q\in T} f_{p\to q} - \sum_{x\to y\in E, x\in T, y\in S} f_{x\to y} \\
&\leq \sum_{p\to q\in E, p\in S, q\in T} f_{p\to q} && (f_{x\to y}\geq 0 \text{ by feasibility of } f) \\
&\leq \sum_{p\to q\in E, p\in S, q\in T} c_{p\to q} && (f_{p\to q}\leq c_{p\to q} \text{ by feasibility of } f)
\end{aligned}
$$

□

5 The MaxFlow-MinCut Theorem

We now know that a MinCut solution is an upper bound of a MaxFlow solution: given a feasible (s,t)-flow f^* with maximum value and an (s,t)-cut (S^*, T^*) with minimum capacity, Theorem 3 implies that we have $|f^*| \leq ||S^*, T^*||$. But is it possible that we have $|f^*| < ||S^*, T^*||$? The strong version of the theorem proves that this is impossible, and that there is a correspondence between MaxFlow solutions and MinCut solutions in the sense that a solution to one induces a solution to the other.

Theorem 4 *Let f^* be a flow with maximum value and (S^*, T^*) be a cut with minimum capacity. Then $|f^*| = ||S^*, T^*||$.*

Figure 4 diagrammatically summarizes the proof strategy that we will use to prove Theorem 4. For any arbitrary (s,t)-flow f, we will construct an object called a **residual graph** that possibly induces the existence of another object called an **augmenting path**. We can use any augmenting path to increase the value of the flow f, and so an augmenting path exists if/only if f is not maximum. Finally, we show that in the case where no augmenting path exists, we can obtain a minimum (s,t)-cut from the residual graph.

This theorem is usually attributed to Ford & Fulkerson (1954), and the proof leads to the Ford Fulkerson algorithm for MaxFlow/MinCut. We will also look at the Edmonds-Karp algorithm, which employs a better method for computing a choice of augmenting path. We will see that these algorithms depend on the capacities of the graph being integral, and that in some cases non-integer capacities can prevent them from terminating.

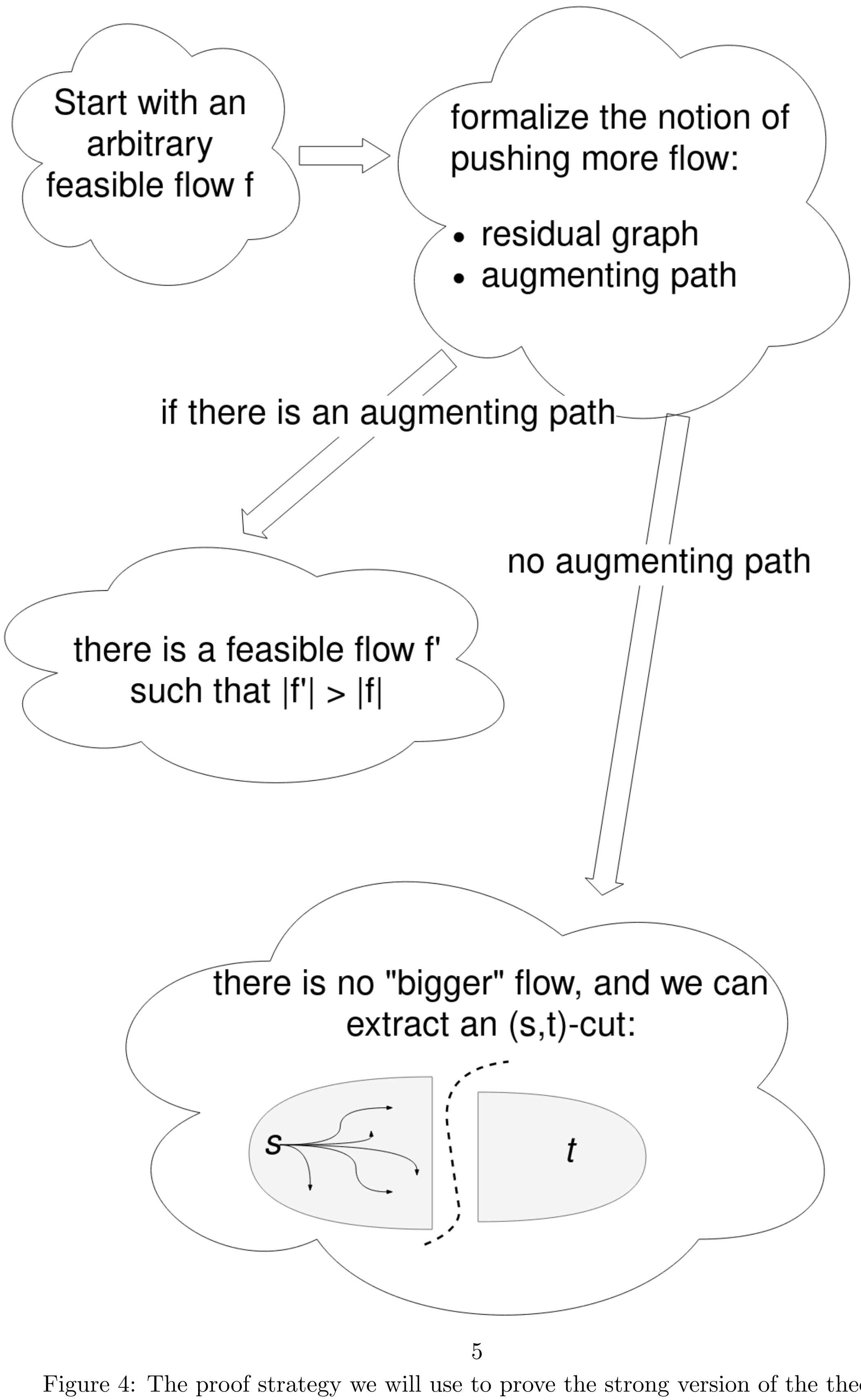

Figure 4: The proof strategy we will use to prove the strong version of the theorem