**CS 3510/3511 Algorithms** 4/11/2024

# Linear Programming

*Lecturer: Jan van den Brand* *Scribe(s): Ankith Thalanki*

Linear Programming is an optimization problem, where we determine the max value of some equation given linear constraints on the variables that make up that equation. For example, maximizing $4x_1 - x_2$ where $x_1 \geq 0$ and $x_2 \leq 0$ and $x - 1 + x_2 = 10$ would be an example of a linear programming problem.

## 1 Linear Programming

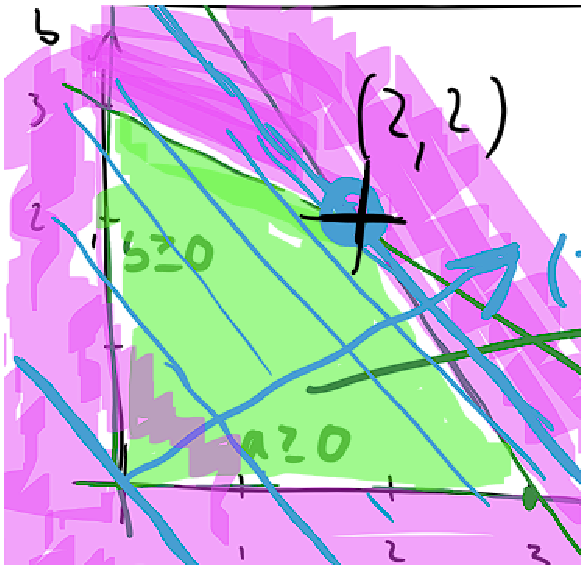
Let's say we have variables $x_i$, for $1 \leq 1 \leq d$, along with a $n$ linear constraints on these variables, like $x_1 + x_2 \leq 3$. The Linear Programming problem asks us what values of $x_i$ allow us to maximize some expression,like $3x_2 - x_4$, where $x_i$ follows all the linear constraints previously imposed. We can define the problem as 4 matrices, $A$, $b$,$x$, and $c$, where $A$ and $b$ represent the constraints, $c$ represents the quantity we wish to maximize, and $x$ represents the values of the variables. If we have $d$ variables with $n$ constraints, We have $A \in \mathbb{R}^{n \times d}$, $c \in \mathbb{R}^d \wedge x \in \mathbb{R}^d$, and $b \in \mathbb{R}^n$. We would wish to maximize $c^T x$, when we have the constraints $Ax \leq b$. This only allows $\leq$ constraints, but the below section shows that this doesn't end up making a difference.

### 1.1 Converting = and $\geq$ to $\leq$

While there are several online linear programming tools that allow you to have constraints of various definitions, the traditional definition of the problem using matrices only lets you use the $\leq$ conversion. However, it is trivial to convert $\geq$ and $=$ to $\leq$. For example, if we have some equation $a_1x_1 + a_2x_2 + a_3x_3 \geq c$, we can simply multiply by negative 1. This would give us $-a_1x_2 - a_2x_3 - a_3x_3 \leq -c$, which is equivalent to the previous condition and a $\leq$ condition. If we have $a_1x_1 + a_2x_2 + a_3x_3 = c$, we can simply split this into a lower and upper bound, like $a_1x_1 + a_2x_2 + a_3x_3 \leq c$ and $a_1x_1 + a_2x_2 + a_3x_3 \geq c$. As we already know how to convert $\geq$ into $\leq$, this allows us to convert an equality condition into less than or equal conditions.

### 1.2 Visualization

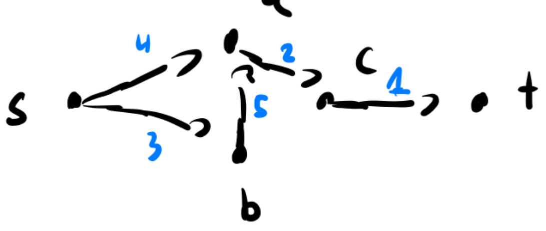
We can visualize the linear programming problem in $\mathbb{R}^n$, where $n$ represents the number of variables in our problem. Each condition can be visualized as a $\mathbb{R}^{n-1}$-dimensional hyperplane slicing away at the space to give us some sort of $\mathbb{R}^n$-dimensional polytope. For example, a linear programming problem with 2 variables could have the following polytope represent the space not cut of by the constraints:



The purple represents the color removed by the constraints, while the green represents the valid space for the values of the variables. How can we conceptualize the weighted sum of variables we wish to maximize, say $a$? One way to conceptualize this is to set a lower bound on $a$, such as $a \geq c$. We can gradually increase $c$ until the hyper-volume of the polytope defined just barely becomes zero. Increasing $c$ would mean that no points satisfy our constraints, so that would make $c$ the greatest possible value of $a$. The points intersected by all the constraints and $a \geq c$ represent the values of the variables where we maximize $a$. In the image above, we can visualize $a \geq c$ as the blue linee, where as we increase $c$, the blue line moves along with the blue arrow until the polytope has an area of zero.

## 1.3 Max Flow & Min Cut

We can easily solve the $st$ max-flow problem, and by equality the $st$ min-cut problem with linear programming. Let us say we have some flow network $G$ we want to find the $st$ max-flow of, like below:



In order to solve this $st$ max-flow problem, we can define all our variables as the respective flows for each edge. For each $f_e$, $f_e \geq 0$, and $f_e \leq w_e$, as the flow cannot be non-negative. We then need to enforce that the flow entering an edge is equivalent to the flow leaving an edge. We can simply add the constraints, $\sum_{(u,v)\in E} f_{(u,v)} = \sum_{(v,w)\in E} f_{(v,w)}$,for every vertex $v$ in $G$. What would we technically maximize in the linear program? We can't just maximize the sum or some linear combination of the flows, as we don't know how those flows are constructed to determine the flow between $s$ and $t$. We can do so by adding another directed edge $(t, s)$, and attempting to maximize the flow along this edge. We can set the capacity or weight of this edge to some ridiculous limit, to ensure that the capacity of this edge doesn't restrict the actual flow of the graph. When we add this extra edge, we get a flow network like the following:

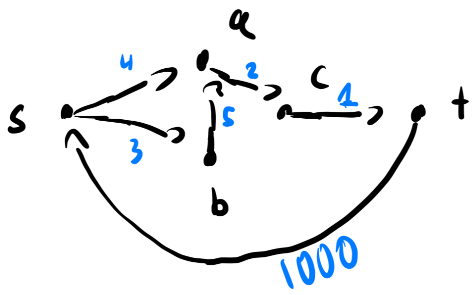


Figure 1: Enter Caption

In this case, our linear programming problem would simply be to maximize $f_{(t,s)}$, with the constraints specified above on every flow $f_e$.