

Linear Programming and simplex algorithm.

Seminar

Optimization for ML. Faculty of Computer Science. HSE University

Linear Programming Recap. Common Forms

For some vectors $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and matrix $A \in \mathbb{R}^{m \times n}$

- Basic form of Linear Programming Problem is:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^\top x && (\text{LP.Basic}) \\ \text{s.t.} \quad & Ax \leq b \end{aligned}$$

- Standard Form of Linear Programming Problem is:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & c^\top x && (\text{LP.Standard}) \\ \text{s.t.} \quad & Ax = b \\ & x_i \geq 0, \quad i = 1, \dots, n \end{aligned}$$

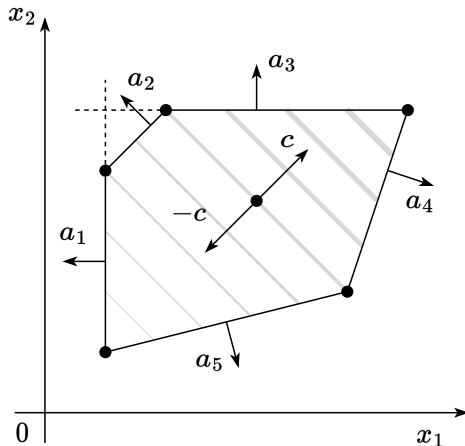


Figure 1: Illustration of the LP Problem.

Linear Programming Recap. Primal and Dual Problems

There are four possibilities:

1. Both the primal and the dual are infeasible.
2. The primal is infeasible and the dual is unbounded.
3. The primal is unbounded and the dual is infeasible.
4. Both the primal and the dual are feasible and their optimal values are equal.

Simplex Algorithm Foundations

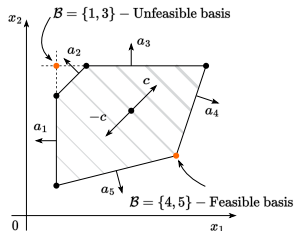


Figure 2: Simplex Algorithm main notions.

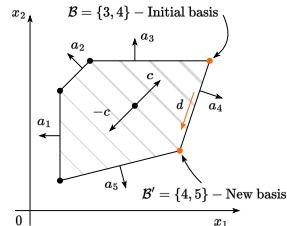


Figure 3: Simplex Algorithm basis change.

i Simplex Algorithm main notions

- A **basis** B is a subset of n (integer) numbers between 1 and m , so that $\text{rank} A_B = n$. Note, that we can associate submatrix A_B and corresponding right-hand side b_B with the basis B . Also, we can derive a point of intersection of all these hyperplanes from basis: $x_B = A_B^{-1}b_B$.
- If $Ax_B \leq b$, then basis B is **feasible**.
- A basis B is **optimal** if x_B is an optimum of the LP.Basic.

Simplex Algorithm Foundations

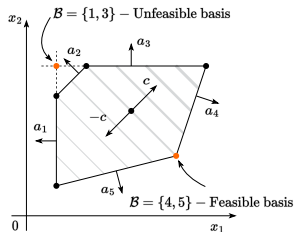


Figure 4: Simplex Algorithm main notions.

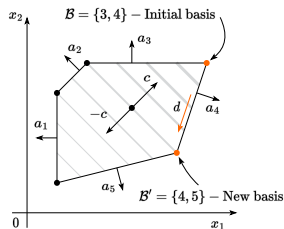


Figure 5: Simplex Algorithm basis change.

i Simplex Algorithm Intuition

- The Simplex Algorithm walks along the edges of the polytope, at every corner choosing the edge that decreases $c^\top x$ most
- This either terminates at a corner, or leads to an unconstrained edge ($-\infty$ optimum)

Simplex Algorithm Foundations

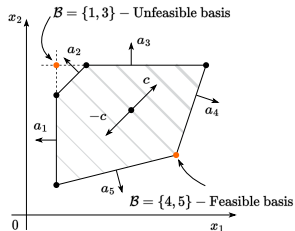


Figure 6: Simplex Algorithm main notions.

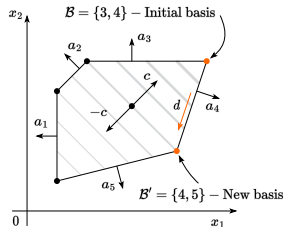


Figure 7: Simplex Algorithm basis change.

💡 Existence of the Standard LP Problem Solution

1. If Standard LP has a nonempty feasible region, then there is at least one basic feasible point
2. If Standard LP has solutions, then at least one such solution is a basic optimal point.
3. If Standard LP is feasible and bounded, then it has an optimal solution.

Simplex Algorithm Foundations

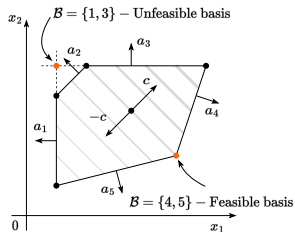


Figure 8: Simplex Algorithm main notions.

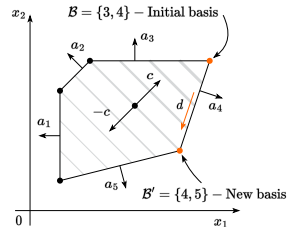


Figure 9: Simplex Algorithm basis change.

💡 Corner Optimality Theorem

Let λ_B be the coordinates of our objective vector c in the basis B :

$$\lambda_B^\top A_B = c^\top \leftrightarrow \lambda_B^\top = c^\top A_B^{-1}$$

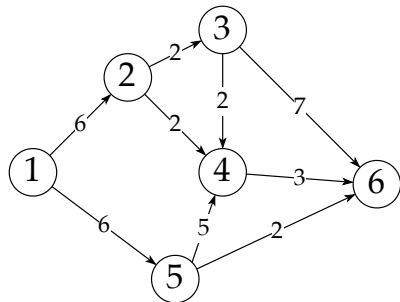
If all components of λ_B are non-positive and B is feasible, then B is optimal.

LP Problems Examples. Production Plans

Suppose you are thinking about starting up a business to produce a *Product X*.

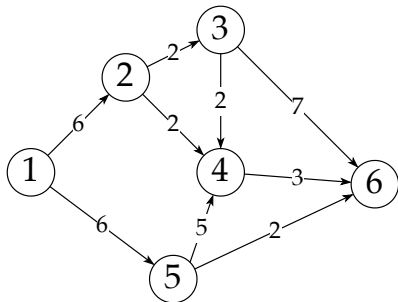
Let's find the maximum weekly profit for your business in the 🐍Production Plan Problem.

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Max-flow problem example

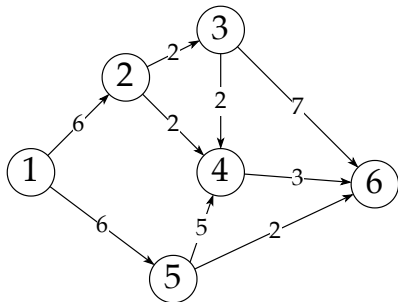


Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

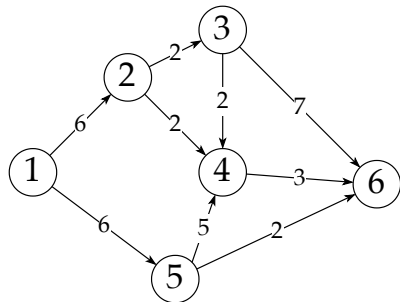
Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

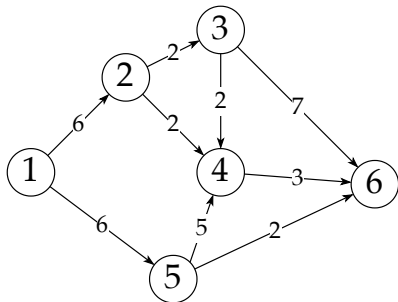
- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Flow Matrix: $X[i, j]$ represents flow from node i to node j .

Max-flow problem example



The nodes are routers, the edges are communications links; associated with each node is a capacity — node 1 can communicate to node 2 at as much as 6 Mbps, node 2 can communicate to node 4 at up to 2 Mbps, etc.

Question:

- A network of nodes and edges represents communication links, each with a specified capacity.
- Example: Can node 1 (source) communicate with node 6 (sink) at 6 Mbps? 12 Mbps? What is the maximum rate?

Capacity Matrix:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

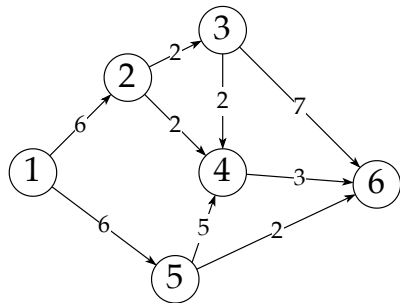
Flow Matrix: $X[i, j]$ represents flow from node i to node j .

Constraints:

$$0 \leq X \leq C$$

$$\text{Flow Conservation: } \sum_{j=2}^N X(i, j) = \sum_{k=1}^{N-1} X(k, i), \quad i = 2, \dots, N-1$$

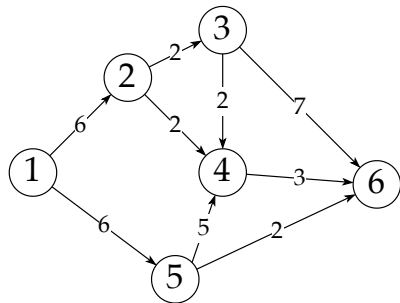
Max-flow problem example



Given the setup, when everything, that is produced by the source will go to the sink. The flow of the network is simply the sum of everything coming out of the source:

$$\sum_{i=2}^N X(1, i) \quad (\text{Flow})$$

Max-flow problem example



Given the setup, when everything, that is produced by the source will go to the sink. The flow of the network is simply the sum of everything coming out of the source:

$$\sum_{i=2}^N X(1, i) \quad (\text{Flow})$$

$$\text{maximize } \langle X, S \rangle$$

$$\text{s.t. } -X \leq 0$$

$$X \leq C$$

$$\langle X, L_n \rangle = 0, \quad n = 2, \dots, N-1,$$

(Max-Flow Problem)

L_n consists of a single column (n) of ones (except for the last row) minus a single row (also n) of ones (except for the first column).

$$S = \begin{bmatrix} 0 & 1 & \dots & 1 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & -1 & \dots & -1 \\ 0 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}.$$

Deriving dual to the Max-flow

Deriving dual to the Max-flow

$$\begin{aligned} & \text{minimize } \langle \Lambda, C \rangle \\ & \Lambda, \nu \\ \text{s.t. } & \Lambda + Q \geq S \\ & \Lambda \geq 0 \end{aligned} \quad (\text{Max-Flow Dual Problem})$$

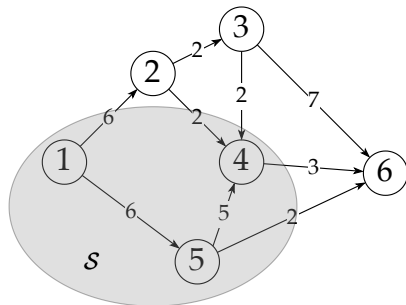
where

$$Q = \begin{bmatrix} 0 & \nu_2 & \nu_3 & \cdots & \nu_{N-1} & 0 \\ 0 & 0 & \nu_3 - \nu_2 & \cdots & \nu_{N-1} - \nu_2 & -\nu_2 \\ 0 & \nu_2 - \nu_3 & 0 & \cdots & \nu_{N-1} - \nu_3 & -\nu_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \nu_2 - \nu_{N-1} & \nu_3 - \nu_{N-1} & \cdots & 0 & -\nu_{N-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Min-cut problem example

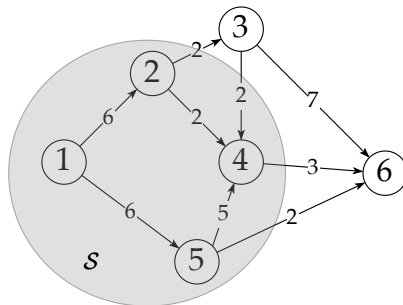
A cut of the network separates the vertices into two sets: one containing the source (we call this set \mathcal{S} , and one containing the sink. The capacity of the cut is the total value of the edges coming out of \mathcal{S} — we are separating the sets by “cutting off the flow” along these edges.

$$\mathcal{S} = \{1, 4, 5\}$$



The edges in the cut are $1 \rightarrow 2$, $4 \rightarrow 6$, and $5 \rightarrow 6$ the capacity of this cut is $6 + 3 + 2 = 11$.

$$\mathcal{S} = \{1, 2, 4, 5\}$$



The edges in the cut are $2 \rightarrow 3$, $4 \rightarrow 6$, and $5 \rightarrow 6$ the capacity of this cut is $2 + 3 + 2 = 7$.

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that \mathcal{S} is a valid cut. From \mathcal{S} , we can easily find a dual feasible point that matches its capacity: for $n = 1, \dots, N$, take

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \quad \text{and} \quad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1, j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that S is a valid cut. From S , we can easily find a dual feasible point that matches its capacity: for $n = 1, \dots, N$, take

$$\nu_n = \begin{cases} 1, & n \in S, \\ 0, & n \notin S, \end{cases} \quad \text{and} \quad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1, j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Notice that these choices obey the constraints in the dual and that $\lambda_{i,j}$ will be 1 if $i \rightarrow j$ is cut, and 0 otherwise, so

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Min-cut is the dual to max-flow

What is the minimum value of the smallest cut? We will argue that it is the same as the optimal value of the solution d^* of the dual program (Max-Flow Dual Problem).

First, suppose that S is a valid cut. From S , we can easily find a dual feasible point that matches its capacity: for $n = 1, \dots, N$, take

$$\nu_n = \begin{cases} 1, & n \in S, \\ 0, & n \notin S, \end{cases} \quad \text{and} \quad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1, j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Notice that these choices obey the constraints in the dual and that $\lambda_{i,j}$ will be 1 if $i \rightarrow j$ is cut, and 0 otherwise, so

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Every cut is feasible, so

$$d^* \leq \text{MINCUT}.$$

Min-cut is the dual to max-flow

Now we show that for every solution ν^*, λ^* of the dual, there is a cut that has a capacity at most d^* . We generate a cut *at random*, and then show that the expected value of the capacity of the cut is less than d^* — this means there must be at least one with a capacity of d^* or less.

Min-cut is the dual to max-flow

Now we show that for every solution ν^*, λ^* of the dual, there is a cut that has a capacity at most d^* . We generate a cut *at random*, and then show that the expected value of the capacity of the cut is less than d^* — this means there must be at least one with a capacity of d^* or less.

Let Z be a uniform random variable on $[0, 1]$. Along with $\lambda^*, \nu_2^*, \dots, \nu_{N-1}^*$ generated by solving (Max-Flow Dual Problem), take $\nu_1 = 1$ and $\nu_N = 0$. Create a cut \mathcal{S} with the rule:

if $\nu_n^* > Z$, then take $n \in \mathcal{S}$.

. . . The probability that a particular edge $i \rightarrow j$ is in this cut is

$$\begin{aligned} P(i \in \mathcal{S}, j \notin \mathcal{S}) &= P(\nu_j^* \leq Z \leq \nu_i^*) \\ &\leq \begin{cases} \max(\nu_i^* - \nu_j^*, 0), & 2 \leq i, j \leq N-1, \\ 1 - \nu_j^*, & i = 1; j = 2, \dots, N-1, \\ \nu_i^*, & i = 2, \dots, N-1; j = N, \\ 1, & i = 1; j = N. \end{cases} \\ &\leq \lambda_{i,j}^*, \end{aligned}$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda_{i,j}^* = d^*.\end{aligned}$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda_{i,j}^* = d^*.\end{aligned}$$

Thus there must be a cut whose capacity is at most d^* . This establishes that

$$\text{MINCUT} \leq d^*.$$

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda_{i,j}^* = d^*.\end{aligned}$$

Thus there must be a cut whose capacity is at most d^* . This establishes that

$$\text{MINCUT} \leq d^*.$$

Combining these two facts of course means that

$$d^* = \text{MINCUT} = \text{MAXFLOW} = p^*,$$

where p^* is the solution of the primal, and equality follows from strong duality for linear programming.

Min-cut is the dual to max-flow

The last inequality follows simply from the constraints in the dual program (Max-Flow Dual Problem). This cut is random, so its capacity is a random variable, and its expectation is

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda_{i,j}^* = d^*.\end{aligned}$$

Thus there must be a cut whose capacity is at most d^* . This establishes that

$$\text{MINCUT} \leq d^*.$$

Combining these two facts of course means that


$$d^* = \text{MINCUT} = \text{MAXFLOW} = p^*,$$

where p^* is the solution of the primal, and equality follows from strong duality for linear programming.

i Max-flow min-cut theorem.

The maximum value of an s-t flow is equal to the minimum capacity over all s-t cuts.

LP Problems Examples. Different Applications

Look at different practical applications of LP Problems and Simplex Algorithm in the  Related Collab Notebook.