LE05: Maximum flows

Fill-in-the-blank text: Maximum flows

Definitions and fundamental theorem

A flow neto $N = (D, \kappa, q, s)$ consists of a directed graph D. a capacity function κ, the source q and the sink s. The number $\omega(f)$ is called the **flow value** and characterizes the total flow through the network. A flow fis called maximal on N if for every other flow f', the value of f' is less than or equal to the value of f ($\omega(T) \leq \omega(U)$).

A subset S of the node set V is called a cut in N if S contains **source** q but not the **sink** s. The capacity of a cut S, denoted by $\kappa(S)$ is the sum of the capacities of the edges running from nodes in S to nodes in **the complement of S** ($V \setminus S$).

The central theoretical result is stated in Lemma 22.13, which says that for every flow f and every cut S, the flow value $\omega(f)$ is always **smaller than or equal to** the cut capacity $\kappa(S)$ ($\omega(f) \leq \kappa(S)$).

Theorem 22.15 of **Ford-Fulkeson** (Max-Flow Min-Cut Theorem) states that the value of a maximum flow is equal to the capacity of a **minimal** cut.

The iterative algorithm

The proof of the max-flow min-cut theorem provides an algorithm for calculating the maximum flow.

In each step a **iterative** path X from q to s is constructed that allows the current flow f to be increased. This path can contain two types of edges:

1. **forward** where the capacity κ is not yet fully utilized ($\kappa(e) > f(e)$).
2. **backward** word where the current flow $f(e)$ is **positive** ($f(e) > 0$).

The flow in provement ε is defined as the **minimum** of all available capacities or flows along this path.

The algorithm is implemented by the two routines **Flow Mark** (Algorithm 22.4) and **Flow Increase** (Algorithm 22.5)

If the routine Flow Mark no longer reaches the sink s, the maximum flow has been reached, and the set S of marked nodes then forms the **minimal cut**.

Termination

For flow networks whose capacity values are **integer** or rational, the algorithm terminates reliably since the flow is increased by a positive integer at each step.

With **irrational** capacity values, termination may fail. To remedy this and guarantee finite time requirements, the algorithm was modified by **Edmonds & Karp** so that the flow-increasing path is found via **breadth-first search**. The Theorem 22.17 on integer values states that with integer capacities an **integer** maximum flow also exists.