Advanced Algorithms Course. Lecture Notes. Part 5

Disjoint Paths Again

In a directed graph with source s and sink t, we want to find as many as possible mutually edge-disjoint $s-t$ paths. We considered the problem earlier, but with k different source-sink pairs. That problem was NP-complete, but this one here is polynomial. Surprisingly, the mere fact that all sources and all sinks, respectively, coincide makes the problem easier.

We can solve it as follows. Give all edges the capacity 1. If k disjoint $s-t$ paths exist then, obviously, there is a flow with $val(f) = k$. The converse is also true: Once we have computed a flow with $val(f) = k$ and integer flow values on the edges (which can be only 0 or 1), we can successively extract k different $s-t$ paths consisting of "1-edges", thanks to the conservation constraints. The time is again $O(mC) = O(mn)$, including the time for this decomposition of the flow into paths.

Disconnecting Edge Sets

Given a directed graph with source s and sink t, we want to remove a minimum-cardinality set F of edges such that s and t are disconnected, that means, all directed $s-t$ paths are destroyed. This problem is motivated by questions of network reliability.

Not surprisingly, the smallest possible size of F equals the maximum number k of edge-disjoint $s-t$ paths; this is called **Menger's Theorem**. For the proof, consider any edge set F that disconnects s and t. Since F must contain some edge from every $s-t$ path, we have $|F| \geq k$. On the other hand, we have seen earlier that $k = val(f)$ holds for a maximal $s-t$ flow f (where all edge capacities are assumed to be 1). By the Max-Flow Min-Cut Theorem there exists a cut (A, B) with capacity k. Now let F_0 be the set of directed edges from A to B. Clearly $|F_0| = k$, and F_0 disconnects

s and t. Together this establishes the claimed equality. Moreover, we can solve the given problem using the previous algorithms.

In undirected graphs we can state the same problem and solve it in the same way. We only need a preprocessing step where we replace every (undirected) edge with two opposite directed edges. It is easy to show that any maximum flow uses at most one of the opposite edges, therefore the approach works.

Circulations with Demands

This is another very useful variant of flow problems. In a directed graph, let S and T be sets of sources and sinks, with given **supply and demand** values $d(v) < 0$ for $v \in S$, and $d(v) > 0$ for $v \in T$. We have $d(v) = 0$ for all nodes not in $S \cup T$. A **circulation** is a function f on the edges such that: $0 \leq f(e) \leq c_e$ for all edges e (capacity constraints) and $f^-(v) - f^+(v) = d(v)$ for all nodes v (demand constraints). A circulation is similar to a flow, except that we want the sources (sinks) to deliver (consume) an exactly prescribed amount; think of suppliers and customers of some goods in a network. The Circulation problem is only concerned with the *existence* of a feasible solution. This time we do not maximize or minimize anything.

Trivially, $\sum_v d(v) = 0$ is a necessary condition for the existence of a circulation. Once this condition is fulfilled, we can reduce Circulation to the Maximum Flow problem (yes, even though it is not an optimization problem). This works as follows. Insert new nodes s, t, and edges (s, u) and (v, t) for all $u \in S$ and $v \in T$. These edges have capacities $-d(u)$ and $+d(v)$, respectively. The idea of this construction should be obvious, as well as the equivalence: A circulation exists if and only if the extended graph has a flow whose value is the sum of all supplies.

Besides the usual upper bounds c_e on capacities, we may also have capacity constraints involving lower bounds: $l_e \leq f(e) \leq c_e$. That means, at least an amount of l_e must flow on edge e. In this case we can simply adjust the capacities to $c_e - l_e$ and also adjust the demands and supplies accordingly. (Think about the straightforward details.) Then we are back to the "classical" problem. However, the slightly generalized problem is sometimes more convenient for modelling other problems; an example will be shown soon.

Now we go through some applications where you would not even expect flows and cuts at first glance.

Planning for Data Mining: Survey Design

A company sells several products, and customers shall be asked about their satisfaction with the products. Each customer i gets questions about some products, but only about such products (s)he has purchased. The number of questions to customer i shall be between c_i and c'_i. (The survey must be informative enough, but not too long and tiresome.) Moreover, between p_j and p'_j customers shall be asked about each product j. (Reasons are similar: The survey must generate enough data to ensure statistical significance, but it should not be too large and costly.) The obvious problem is: Does there exist a survey with the given constraints, and if so, how can we construct one?

We represent customers and products by nodes of a bipartite graph. A directed edge (i, j) is inserted if customer i has purchased product j. We add nodes s and t, edges from s to all customers, and from all products to t. We also insert an edge from t to s with huge capacity. All demands are set to 0. The lower and upper capacity bounds are c_i, c'_i for edges starting in s, further $0, 1$ for customer-product edges, and p_j, p'_j for edges ending in the sink t.

We claim that the possible surveys correspond to circulations in this graph (with integer values on the edges), where the survey questions are the edges (i, j) with flow value 1. Think about the role of every edge in this construction.

A Simplified Airline Scheduling Problem

An airline has to operate m flight segments where each segment j is characterized by: origin and destination airport, departure and arrival time. For any two flight segments we define, by ad-hoc criteria, when flight j is "reachable" from flight i (for example: same airport, enough time between the arrival of i and departure of j). This reachability relation defines a directed acyclic graph (DAG) on the set of flight segments. If j is reachable from i, then both flight segments *may* be performed by the same plane. A fleet of k planes is available. The problem is: Can the given flight schedule be realized using at most k planes?

We model the problem as a circulation problem. Here is is tempting to represent the airports by nodes, but in order to allow for various reachability criteria we model everything by edges, whereas the nodes are merely the ends of edges but have no "physical" meaning. Now a possible construction follows. The pairs of numbers mean lower and upper bounds on capacities, and the directions of edges should be obvious.

We represent flight segments by mutually disjoint edges with capacities $(1, 1)$, since every flight must be performed. If a flight is reachable by another one, we connect the two edges by another edge with capacity $(0, 1)$, since we may use this connection but we are not obliged to do so. Furthermore, insert a source s and a sink t. Connect s with each flight, by an edge of capacity $(0, 1)$. Similarly, connect each flight with t, by an edge of capacity $(0, 1)$. Finally we connect s with t by an edge of capacity $(0, k)$. Nodes s and t have demands $-k$ and k, respectively, and all other nodes have demand 0.

It is not hard to see that the possible schedules correspond to the possible circulations, where the flow on edge (s, t) is the number of unused planes. Each of the other "units of flow" corresponds to one plane and connects all the flight segments operated by that plane.