MATH 409 LECTURE 9
SHORTEST PATH PROBLEM

REKHA THOMAS

This material is taken mostly from [1].

Definition 1. A **directed graph (digraph)** $G = (V, E)$ consists of a vertex set V and directed edges $e = ij \in E$ called **arcs** such that i is the tail of e and j is the head of e.

Note that in a digraph G, $e = ij \neq f = ji$. Multiple edges between i and j in V are considered to be parallel arcs if and only if they all have the same tail and same head. The set of arcs $\{ij, ji\}$ form a **directed circuit** and are not considered to be parallel arcs.

Definition 2.
- A digraph G is **simple** if it has no parallel arcs or loops.
- The undirected graph obtained by erasing all the directions on the arcs of a digraph G is called the **underlying graph** of G.
- The digraph G is **connected** if its underlying graph is connected.
- A **directed path (dipath)** in a digraph G is a sequence of vertices and arcs: $v_0, e_1, v_1, e_2, \ldots, v_k$ such that v_i is the tail of e_{i+1} and v_{i+1} is the head of e_{i+1}.
- If there are costs $c_e \in \mathbb{R}$ on the edges of G, the **cost of a dipath** P in G is $c(P) = \sum_{e \in P} c_e$.

Definition 3. Given a digraph G with edge costs $c_e \in \mathbb{R}$ for all $e \in E(G)$ and two vertices r and s, the **shortest path problem** is to find the minimum cost dipath from r to s.

Note that, in the presence of edge costs, we are calling a min cost dipath a "shortest" path. It may not be the dipath between r and s with the least number of edges. If the edge costs were a constant for all $e \in E(G)$, then the shortest path would also be the path of least length. Before we can look for algorithms to solve the shortest path problem, we have to make some assumptions on the input graph G to avoid trivial instances. The assumptions that come before are assumed in the ones that come later.

Date: April 16, 2010.

Assumptions.

- G is **connected**.
- We assume that G has **no directed cycles of negative cost**. This is because, otherwise, there will be vertices r and s with a (r, s)-dipath that touches this circuit which will allow us to include an infinite number of loops around this cycle in the dipath, driving the cost of the dipath to $-\infty$. The shortest path problem would then be unbounded for these vertices.
- We assume that G is **simple**. By the above assumption, there are already no loops of negative cost. If there was a loop of non-negative cost, we would not include it in any shortest path. If there are parallel edges then we could erase all but the cheapest edge and leave the optimum of the shortest path problem intact.
- If the problem it to find the shortest (r, s)-path, we will assume that there is a dipath from r to v for every $v \in V(G)$. If this is not the case, we add a dummy arc rv with $c_{rv} = \infty$. The shortest (r, v)-path will use this dummy arc only if there was no dipath from r to v.

Lemma 4. Bellman's principle: *Let G be a digraph with all the above assumptions. If $e = vs$ is the final edge of a shorest path $P_{[r,s]}$ from r to s, then $P_{[r,v]}$, which is $P_{[r,s]}$ without $e = vs$ is a shortest path from r to v.*

Proof. We prove this by the method of contradiction. Suppose Q is a shorter (r, v)-dipath than $P_{[r,v]}$. Then $c(Q) + c_e < c(P_{[r,v]}) + c_e = P_{[r,s]}$. We consider two cases:
Case 1: Suppose $s \notin Q$. Then $Q + e$ is a dipath from r to s of cost less than $P_{[r,s]}$ which is a contradiction.
Case 2: Suppose $s \in Q$. Then $c(Q_{[r,s]}) = c(Q) + c_e - c(Q_{[s,v]} + e) < c(P_{[r,s]}) - c(Q_{[s,v]} + e) \leq c(P_{[r,s]})$. The last inequality comes from the fact that the cycle $Q_{[s,v]} + e$ has non-negative cost by assumption. □

Corollary 5. *If P is a shortest path from r to s, then for all v in this path, $P_{[r,v]}$ is a shortest path from r to v.*

We will look for an algorithm that computes the shortest (r, v)-dipath **for all** $v \in V(G)$. This of course includes the shortest (r, s)-path in G. Note that if we adopt this strategy, then we can store a shortest path from r to v by simply storing the last edge in the path. The shortest path between any two vertices can be reconstructed from this.

References

[1] B. Korte and J. Vygen. *Combinatorial Optimization*. Springer, Berlin, 2000.