Shortest Path Algorithms

Luis Goddyn, Math 408

Given an edge weighted graph (G, d), $d : E(G) \to \mathbb{Q}$ and two vertices $s, t \in V(G)$, the *Shortest Path Problem* is to find an s, t-path P whose total weight is as small as possible. Here, G may be either directed or undirected. A path in a graph is a sequence $v_0 e_1, v_1, \dots, v_k$ of vertices and edges such that no vertex or edge appears twice, and e_i joins v_{i-1} to v_i. If G is directed, then e_i should be oriented from v_{i-1} to v_i.

1 Dijkstra's Algorithm

0. Input points (G, d, s). Label all vertices with $\ell(v) = \infty$, and set tree $T = \{s\}$. Set $\ell(s) = 0$. The current vertex is $v = s$.

1. For every arc vw where $w \notin T$, if $\ell(v) + d(vw) < \ell(w)$, then relabel w via $\ell(w) = \ell(v) + d(vw)$, and set a pointer $p(w) = v$.

2. Find a vertex $x \in V(G) - V(T)$ having the smallest ℓ-label. If there is no such vertex, or if $\ell(x) = \infty$, then output T, ℓ and STOP, as no other vertices are reachable from s.

3. Add the vertex x and the arc $p(x)x$ tree T. Go to step 1.

If d is a *conservative weighting*, that is, if G has no negative weight directed circuits (circuits C whose total weight $d(C)$ is negative), then Dijkstra's algorithm stops with a *shortest path tree* T rooted at s. Every vertex which is reachable from s is in T and for every $w \in V(T)$, the unique sw-path in T is a shortest sw-path in (G, d). We omit the proof that this algorithm works correctly and stops in polynomial time.

2 Conservative Weightings - An Algorithm

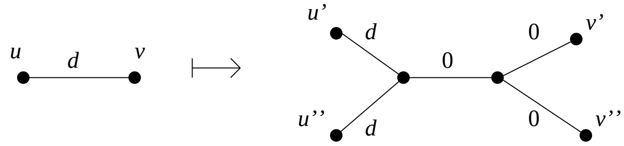
If G has negative weight circuits, then there is no known algorithm which finds a shortest s, t-path in (G, d), since we could solve any Hamilton Path problem by setting $d(e) = -1$ for every arc e, and the Hamilton Path Problem is known to be "NP-Hard".

What if G is undirected? One method here is to replace each edge uv in G by two oppositely-directed arcs uv and vu, and then run Dijkstra's algorithm on the resulting directed graph. This works well provided that (G, d) has no negative weight edges. Any negative-weight edge would convert into a digon (a directed circuit of length two) having negative weight, and so Dijkstra's algorithm no longer works. Other shortest-path algorithms, such as the Floydd-Warshall algorithm for undirected graphs has the same draw-back, failing to work correctly if even one edge has negative weight.

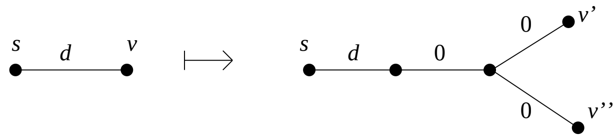
However, there is a way to solve shortest path problems for undirected graph with negative-weight edges, provided that (G, d) is conservatively weighted. Here is the method.

1. Input points (G, d, s, t). Replace every vertex $v \in V(G) - \{s, t\}$ with two new vertices v', v'' joined by a new edge of weight zero. Replace s and t with new vertices s' and t'.

2. For every edge uv where $u, v \neq s, t$ we replace uv with the following gadget, weighted as indicated below. Note that three of the five edges of the gadget get weight zero and the other two get weight $d(uv)$.



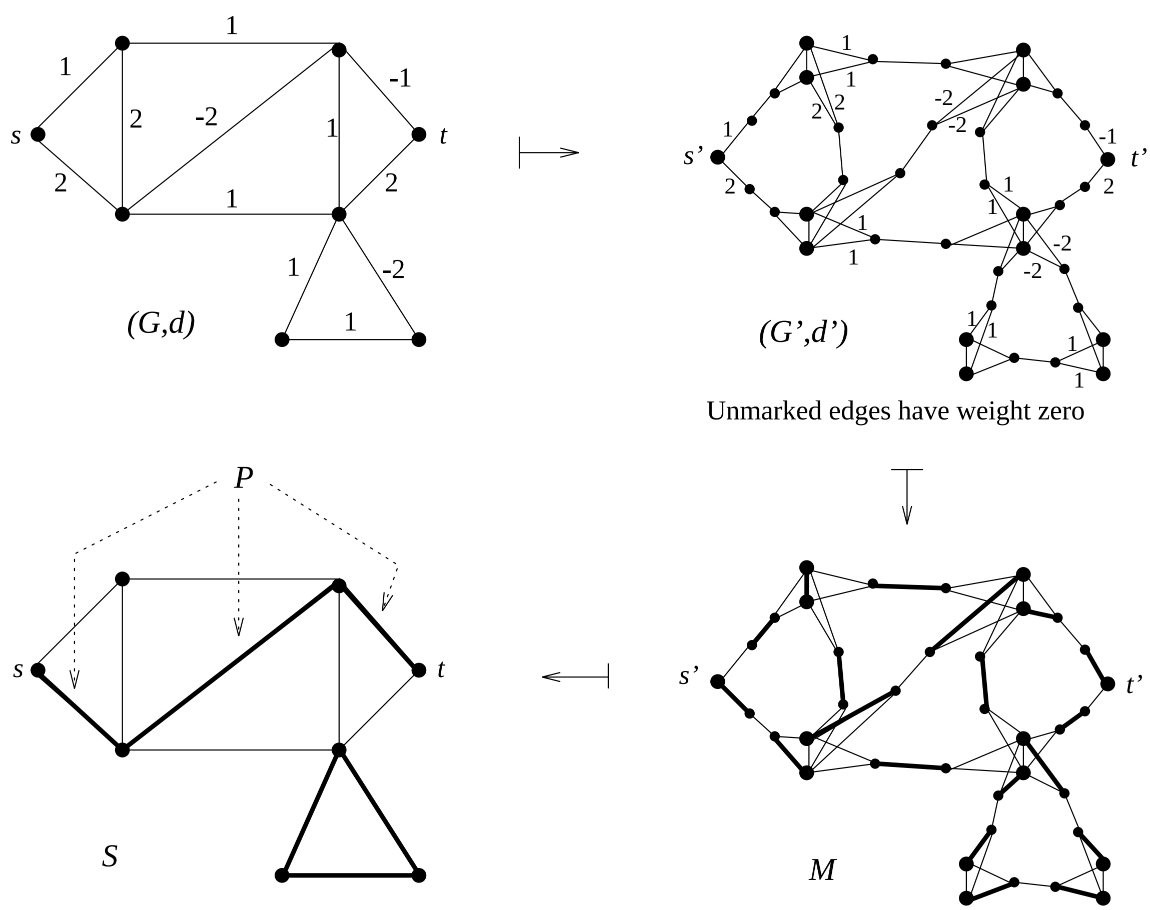
Replace any edge su with the following gadget, and similarly for any edge ut. (We leave it to the reader to decide what to do if there is an edge from s to t!)



3. Run Edmonds' Minimum Weight Perfect Matching Algorithm on the resulting weighted graph (G', d'), obtaining the matching M.

4. Interpret M as an st-path in G as follows. Let $g(uv)$ be the 5-edge gadget in G' corresponding to edge $uv \in E(G)$. Either one or two edges of each gadget belongs to M. Let S the set of edges uv in G such that two edges of $g(uv)$ belong to M. It is easy to check that each vertex in $V(G) - \{s, t\}$ is incident with exactly zero or one edges in S, whereas s and t are each incident with exactly one edge in S. Thus S consists of an st-path P and possibly some circuits. Each circuit in S must have total weight zero (Why? This will be a homework question). It it follows that $d'(M) = d(S) = d(P)$. Since M is a minimum weight perfect matching, P must be a minimum weight st-path.

Here is an example of this process.



3 Shortest Odd Path

Given an edge weighted undirected graph (G, d), $d : E(G) \to \mathbb{Q}$ and two vertices $s, t \in V(G)$, the *Shortest Odd Path Problem* is to find an s, t-path P having an *odd number of edges* whose total weight is as small as possible.

If (G, d) is conservative, then this problem can be reduced to a minimum weight perfect matching problem as follows.

1. Let G_1, G_2 be disjoint copies of G, and label with v_i the vertex in G_i corresponding to $v \in V(G)$, $i = 1, 2$. Each edge in $G_1 \cup G_2$ gets the weight of the corresponding edge in G. We form a new weighted graph (G', d') from $G_1 \cup (G_2 - \{s_2, t_2\})$ by adding edges of weight zero $E' = \{v_1v_2 : v \in V(G) - \{s, t\}$. Thus $d'(u_1v_1) = d'(u_2v_2) = d(uv)$ for $uv \in E(G)$, and $d'(u_1u_2) = 0$ for $u \in V(G) - \{s, t\}$.

2. Find a minimum weight perfect matching M in (G', d') using Edmonds' algorithm. If no such matching exists, then there is no s, t-path in G having an odd number of edges.

3. Let S be the set of edges $uv \in E(G)$ such that either u_1v_1 or u_2v_2 is in M. It is easy to see that S induces an s, t-path P together with some disjoint circuits. Here P has an odd number of edges (why?), and one can show that each of the circuits has weight zero. So $d(P) = d(S) = d'(M)$. Since this process is "reversible", and M is a minimum weight perfect matching, P is a minimum weight s, t-path having an even number of edges.

Here is an example of this process.