Linear program formulations of the shortest path problem

Vassos Hadzilacos

Let $G = (V, E)$ be a digraph, let $s, t \in V$ be nodes such that there is an $s \to t$ path, and let $\mathbf{wt} : E \to \mathbb{R}$ be an edge weight function. We do not require edge weights to be non-negative, but we assume that G has no negative-weight cycles, so that shortest paths in G are well-defined. We want to express the problem of finding the weight of a shortest $s \to t$ path as a linear program.

We define a variable x_u for each $u \in V$. Intuitively, x_u represents the weight of a shortest $s \to u$ path. This is not quite accurate, but it will be true at least for x_s and x_t. We add the following constraints:

- $x_s = 0$. This states the obvious: the weight of a shortest path from s to itself must have weight 0.
- For every edge $(u, v) \in E$, add the constraint $x_v \leq x_u + \mathbf{wt}(u, v)$. This simply states the triangle inequality that we saw in Dijkstra's algorithm and in the Ford-Fulkerson algorithm: if (u, v) is an edge then the weight of a shortest $s \to v$ path cannot exceed the weight of a shortest $s \to u$ path plus the weight of the edge (u, v).

The objective function is to ***maximize*** x_t. This may sound paradoxical, as the shortest path problem is a minimization problem! Here is an insightful way to think about this LP that may explain this apparent paradox. Consider a physical model of the graph, where each edge (u, v) consists of a piece of string of length $\mathbf{wt}(u, v)$ that connects points u and v. Now imagine pulling node t as far away as possible from s — i.e., try to maximize their distance. Then shortest paths from s to t will be the paths consisting of the pieces of string (i.e., edges) that are stretched tight; edges not on shortest $s \to t$ paths will be hanging loose, because of the extra length (slack) of these paths. In this physical representation of the graph, maximizing x_t amounts to pulling t apart from s as far as possible, subject to the triangle inequality constaints. These constraints state that we can't pull away from s a node u on the shortest path from s to t more than we can pull away u's predecessor v on that path, plus the length of the piece of string that connects u and v.

So, our LP expressing the shortest path problem is as follows:

(1) **Variables:** x_u, for each node u.

(2) **Objective function:** Maximize x_t.

(3) **Constraints:**

- $x_s = 0$
- $x_v - x_u \leq \mathbf{wt}(u, v)$, for each edge (u, v).

The preceding intuitive explanation of the LP sounds plausible, but not entirely convincing. We now prove that the optimal value of the above LP really is the weight of a shortest $s \to t$ path.

Let x_u^*, for every node u, be an optimal solution to the LP; thus the optimal value of the LP is x_t^*. Let $\hat{x}_u$, for every node u, be the weight of a shortest $s \to u$ path (or ∞ if there is no such path). We will show that $x_t^* = \hat{x}_t$.

Since $\hat{x}_u$ is the weight of a shortest $s \to u$ path, it is clear that, for every edge (u, v), $\hat{x}_v \leq \hat{x}_u + \mathbf{wt}(u, v)$. Furthermore, $\hat{x}_s = 0$. Therefore the $\hat{x}_u$'s form a feasible solution. Since x_t^* maximizes the value of variable x_t over all feasible solutions, we have that $x_t^* \geq \hat{x}_t$.

It remains to show that $x_t^* \leq \hat{x}_t$. Let $v_1, v_2, \ldots, v_k$ be a shortest $s \to t$ path. Thus, $v_1 = s$ and $v_k = t$. By induction, we show that $x_{v_i}^* \leq \hat{x}_{v_i}$, for each $i = 1, \ldots, k$.

For the basis, $i = 1$, we have: The constraint on x_s forces $x_s^* = 0$, and the weight of a shortest path from s to itself is clearly 0, so $\hat{x}_s = 0$. Since $v_1 = s$, we have $x_{v_1}^* \leq \hat{x}_{v_1}$.

For the induction step, suppose $x_{v_i}^* \leq \hat{x}_{v_i}$, for some $i < k$. We will prove that $x_{v_{i+1}}^* \leq \hat{x}_{v_{i+1}}$:

$$\begin{aligned}
x_{v_{i+1}}^* &\leq x_{v_i}^* + \mathbf{wt}(v_i, v_{i+1}) && \text{[by the triangle inequality, since the } x_u^*\text{'s satisfy the constraints]} \\
&\leq \hat{x}_{v_i} + \mathbf{wt}(v_i, v_{i+1}) && \text{[by the induction hypothesis]} \\
&= \hat{x}_{v_{i+1}} && \text{[since } v_1, v_2, \ldots v_{i+1} \text{ is a shortest } s \to v_{i+1} \text{ path]}
\end{aligned}$$

as wanted.

In particular, since $v_k = t$, it follows that $x_t^* \leq \hat{x}_t$. Since we previously showed that $x_t^* \geq \hat{x}_t$, we conclude that $x_t^* = \hat{x}_t$, as wanted.

Question 1. Does the LP have a unique optimal solution?

Question 2. What happens to the LP if the given graph has negative-weight cycles?

Question 3. What happens to the LP if there is no $s \to t$ path in the given graph?

We now describe another LP that expresses the shortest path problem.

(1) **Variables:** x_e, for each edge e.

(2) **Objective function:** Minimize $\sum_{e \in E} \mathbf{wt}(e) x_e$.

(3) **Constraints:**

- $x_e \geq 0$, for every edge e.
- For every node u, $\sum_{e \in \text{out}(u)} x_e - \sum_{e \in \text{in}(u)} x_e = \begin{cases} 1, & \text{if } u = s \\ -1, & \text{if } u = t \\ 0, & \text{if } u \neq s \text{ and } u \neq t \end{cases}$

The intuition behind this LP is as follows: The value of variable x_e is the amount of traffic on edge e. The second constraint says that 1 unit of flow is generated in s and arrives at t, being conserved at every other node. If we think of $\mathbf{wt}(e)$ as the cost per unit of traffic that goes on e, $\sum_{e \in E} \mathbf{wt}(e) x_e$ is the cost of sending this flow from s to t. So, we are trying to minimize the cost of one unit of flow from s to t. The minimum cost is the weight of a shortest $s \to t$ path.

Question 4. Show that the optimal value of the above LP is the weight of a shortest $s \to t$ path in G.