Shortest Path

1 Discrete Problems as Linear Programs

In computer science, discrete problems are often solved with discrete algorithms; in optimization, discrete problems are often solved with continuous optimization. Modern theory accepts that both methods are useful and each can provide insights into the problem.

To cast a discrete problem as an LP, note that any finite family of discrete objects is equivalent to a finite Euclidean subspace. The convex hull of this subspace is a polytope, which we know how to optimize over. Furthermore, LPs require a "small" family of inequalities, which is impossible when a problem is **NP**-hard.

2 Shortest Path LP

2.1 Shortest Path Problem

Problem. Given a directed graph $G = (V, E)$ with cost $c_e \in \mathbb{R}$ on each edge $e \in E$, find the minimum cost path from $s \to t$ with $s, t \in V$, assuming there is one. Let $n = |V|$ and $m = |E|$ and allow negative costs but not negative cycles. When the costs are positive, Dijkstra's Algorithm solves this problem in $\mathcal{O}(m + n \log n)$.

Note that no negative cycles implies that a simple path exists; if negative cycles exist but we want the shortest path which does not encounter the negative cycle, this problem is **NP**-hard by reduction from Hamiltonian Cycle. Usually, we will implement our algorithm to fail gracefully when it encounters a negative cycle.

2.2 Integrality of the Shortest Path

In order to encode this problem as an LP, we have to relax the problem and allow fractional paths. Let $\delta_v = -1$ if $v = s$, $\delta_v = 1$ if $v = t$, and $\delta_v = 0$ otherwise. Then we have the LP:

$$\begin{aligned} \min \ & \sum_{e \in E} c_e x_e \\ \text{s.t.} \ & \sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v \ \forall v \\ & \boldsymbol{x} \succeq 0 \end{aligned}$$

The primal LP encodes the idea that the path should leave s once, enter and leave every other node in the path once, and enter t once, with minimal cost. We will show that the shortest integral path from $s \to t$, denoted $\boldsymbol{x}^*$, is a solution to the primal LP by taking the dual:

$$\begin{aligned} \max \ & y_t - y_s \\ \text{s.t.} \ & y_v - y_u \leq c_e \ \forall (u, v) \in E \end{aligned}$$

The dual LP encodes the idea that we want to find the maximum distance we can "stretch" the graph at s and t without "over-stretching" the edge between any two nodes. Let y_v^* be the shortest path distance from $s \to v$. Then:

$$\sum_{e \in E} c_e x_e^* = y_t^* - y_s^*$$

So by strong duality, both $\boldsymbol{x}^*$ and $\boldsymbol{y}^*$ are optimal.

2.3 Integrality of Polyhedra

The above observation leads us to a stronger statement whose generalization is useful in many combinatorial contexts; that is, that the vertices of the polyhedral feasible region of a relaxed combinatorial LP are precisely the indicator vectors of the optimal solutions. Here, we show that the vertices are the indicator vectors of the shortest paths.

Proof. First, we know our LP is bounded if and only if we disallow negative cycles in the objective function c. Furthermore, for every vertex $\boldsymbol{x}$ (representing an integral path), there is an objective function c such that $\boldsymbol{x}$ is the unique optimal. Since such a c disallows negative cycles, we must have that $\boldsymbol{x}$ is integral. □

We can use this same proof strategy in the general case. To show that a polytope's vertices are integral, it suffices to show that there is an integral optimal for any objective which admits an optimal solution.

3 Primal-Dual Algorithms

For convenience, we can add an edge of infinite length from $s \to v$ when one does not exist. Then, we can study a **primal-dual algorithm** for finding the shortest path, which simultaneously builds optimal solutions for the primal and dual LPs. The algorithm then terminates when they are equal, since by duality we will have reached an optimal solution.

Algorithm 1 Ford's Algorithm

```
$y_v = c_{(s,v)}, pred(v) = s \ \forall v \neq s$
$y_s = 0, pred(s) = null$
while $y_v > y_u + c_e$ for some $e = (u,v) \in E$ do
    $pred(v) = u$
    $y_v = y_u + c_e$
end while
$t, pred(t), pred(pred(t)), \ldots$
```

Ford's Algorithm maintains an initially infeasible dual $\boldsymbol{y}$ and builds a candidate feasible primal $\boldsymbol{p}$ of length $\leq y_t - y_s$. Then, it iteratively fixes $\boldsymbol{y}$ towards feasibility; once $\boldsymbol{y}$ is feasible, weak duality implies that we have an optimal solution. Note that Ford's Algorithm actually solves the single-source shortest path problem, because it found the shortest paths from s to every other node, not just t. The proof of runtime of Ford's Algorithm is slightly technical, but by fixing an arbitrary order on the edges while looking for a violation of a dual constraint, we derive the Bellman-Ford Algorithm, which runs in $\mathcal{O}(mn)$ time