CS599: Convex and Combinatorial Optimization
Fall 2013
Lecture 14: Combinatorial Problems as Linear Programs I

Instructor: Shaddin Dughmi

Announcements

- Posted solutions to HW1
- Today: Combinatorial problems as linear programs
  - Shortest paths

Outline

Combinatorial Vs Convex Optimization

- In CS, discrete problems are traditionally viewed/analyzed using discrete mathematics and combinatorics
  - Algorithms are combinatorial in nature (greedy, dynamic programming, divide and conquor, etc)

Combinatorial Vs Convex Optimization

- In CS, discrete problems are traditionally viewed/analyzed using discrete mathematics and combinatorics
  - Algorithms are combinatorial in nature (greedy, dynamic programming, divide and conquor, etc)
- In OR and optimization community, these problems are often expressed as continuous optimization problems
  - Usually linear programs, but increasingly more general convex programs

Combinatorial Vs Convex Optimization

- In CS, discrete problems are traditionally viewed/analyzed using discrete mathematics and combinatorics
  - Algorithms are combinatorial in nature (greedy, dynamic programming, divide and conquor, etc)
- In OR and optimization community, these problems are often expressed as continuous optimization problems
  - Usually linear programs, but increasingly more general convex programs
- Increasingly in recent history, it is becoming clear that combining both viewpoints is the way to go
  - Better algorithms (runtime, approximation)
  - Structural insights (e.g. market clearing prices in matching markets)
  - Unifying theories and general results (Matroids, submodular optimization, constraint satisfaction)

Discrete Problems as Linear Programs

- The oldest continuous formulations of discrete problems were linear programs
  - In fact, Dantzig's original application was the problem of matching 70 people to 70 jobs!

Discrete Problems as Linear Programs

- The oldest continuous formulations of discrete problems were linear programs
  - In fact, Dantzig's original application was the problem of matching 70 people to 70 jobs!
- This is not surprising, since almost any finite family of discrete objects can be encoded as a finite subset of Euclidean space
  - Convex hull of that set is a polytope
  - E.g. spanning trees, paths, cuts, TSP tours, assignments...

Discrete Problems as Linear Programs

- LP algorithms typically require representation as a “small” family of inequalities,
  - Not possible in general (Say when problem is NP-hard, assuming $(P \neq NP)$)
  - Shown unconditionally impossible in some cases (e.g. TSP)

Discrete Problems as Linear Programs

- LP algorithms typically require representation as a “small” family of inequalities,
  - Not possible in general (Say when problem is NP-hard, assuming $(P \neq NP)$)
  - Shown unconditionally impossible in some cases (e.g. TSP)
- But, in many cases, polyhedra in inequality form can be shown to encode a combinatorial problems at the vertices

Discrete Problems as Linear Programs

- LP algorithms typically require representation as a "small" family of inequalities,
  - Not possible in general (Say when problem is NP-hard, assuming $(P \neq NP)$)
  - Shown unconditionally impossible in some cases (e.g. TSP)
- But, in many cases, polyhedra in inequality form can be shown to encode a combinatorial problems at the vertices

Today

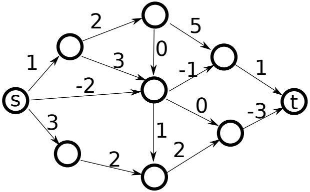
We examine shortest path through a polyhedral lense.

- Re-deriving and simplifying familiar results algorithmic results through the Primal-Dual paradigm
- This is a warmup for more intricate applications of LP and convex optimization to combinatorial problems

The Shortest Path Problem

Given a directed graph $G = (V, E)$ with cost $c_e \in \mathbb{R}$ on edge e, find the minimum cost path from s to t.

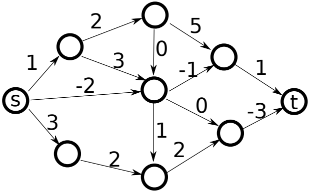
- We use n and m to denote $|V|$ and $|E|$, respectively.
- We allow costs to be negative, but assume no negative cycles



The Shortest Path Problem

Given a directed graph $G = (V, E)$ with cost $c_e \in \mathbb{R}$ on edge e, find the minimum cost path from s to t.

- We use n and m to denote $|V|$ and $|E|$, respectively.
- We allow costs to be negative, but assume no negative cycles



When costs are nonnegative, Dijkstra's algorithm finds the shortest path from s to every other node in time $O(m + n \log n)$.

Using primal/dual paradigm, we will design a polynomial-time algorithm that works when graph has negative edges but no negative cycles

Note: Negative Edges and Complexity

- When the graph has no negative cycles, there is a shortest path which is simple
- When the graph has negative cycles, there may not be a shortest path from s to t.
- In these cases, the algorithm we design can be modified to "fail gracefully" by detecting such a cycle
  - Can be used to detect arbitrage opportunities in currency exchange networks

Note: Negative Edges and Complexity

- When the graph has no negative cycles, there is a shortest path which is simple
- When the graph has negative cycles, there may not be a shortest path from s to t.
- In these cases, the algorithm we design can be modified to “fail gracefully” by detecting such a cycle
  - Can be used to detect arbitrage opportunities in currency exchange networks
- In the presence of negative cycles, finding the shortest simple path is NP-hard (by reduction from Hamiltonian cycle)

Outline

An LP Relaxation of Shortest Path

Consider the following LP

Primal Shortest Path LP

min $\sum_{e \in E} c_e x_e$

s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$

$$x_e \geq 0, \qquad \forall e \in E.$$

where $\delta_v = -1$ if $v = s$, 1 if $v = t$, and 0 otherwise.

An LP Relaxation of Shortest Path

Consider the following LP

Primal Shortest Path LP

$$
\begin{aligned}
&\min \textstyle\sum_{e \in E} c_e x_e \\
&\text{s.t.} \\
&\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad && \forall v \in V. \\
&x_e \geq 0, && \forall e \in E.
\end{aligned}
$$

where $\delta_v = -1$ if $v = s$, 1 if $v = t$, and 0 otherwise.

- This is a relaxation of the shortest path problem
  - Indicator vector x_P of $s - t$ path P is a feasible solution, with cost as given by the objective
  - Fractional feasible solutions may not correspond to paths

An LP Relaxation of Shortest Path

Consider the following LP

Primal Shortest Path LP

$$\begin{aligned} &\text{min } \textstyle\sum_{e \in E} c_e x_e \\ &\text{s.t.} \\ &\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V. \\ &x_e \geq 0, \qquad\qquad\qquad\qquad \forall e \in E. \end{aligned}$$

where $\delta_v = -1$ if $v = s$, 1 if $v = t$, and 0 otherwise.

- This is a relaxation of the shortest path problem
  - Indicator vector x_P of $s - t$ path P is a feasible solution, with cost as given by the objective
  - Fractional feasible solutions may not correspond to paths
- A-priori, it is conceivable that optimal value of LP is less than length of shortest path.

Integrality of the Shortest Path Polyhedron

$$
\begin{aligned}
&\text{min } \textstyle\sum_{e \in E} c_e x_e \\
&\text{s.t.} \\
&\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad && \forall v \in V. \\
&x_e \geq 0, && \forall e \in E.
\end{aligned}
$$

We will show that above LP encodes the shortest path problem exactly

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s-t$ path is an optimal solution to the LP.

Dual LP

We will use the following LP dual

Primal LP

min $\sum_{e \in E} c_e x_e$
s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$
$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$
s.t.

$$y_v - y_u \leq c_e, \quad \forall (u,v) \in E.$$

- Interpretation of dual variables y_v: "height" or "potential"
- Relative potential of vertices constrained by length of edge between them (triangle inequality)
- Dual is trying to maximize relative potential of s and t,

Proof Using the Dual

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s - t$ path is an optimal solution to the LP.

Proof Using the Dual

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s - t$ path is an optimal solution to the LP.

Primal LP

min $\sum_{e \in E} c_e x_e$
s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$

$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$
s.t.

$$y_v - y_u \leq c_e, \quad \forall (u, v) \in E.$$

Proof Using the Dual

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s-t$ path is an optimal solution to the LP.

Primal LP

min $\sum_{e \in E} c_e x_e$
s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$
$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$
s.t.

$$y_v - y_u \leq c_e, \quad \forall (u,v) \in E.$$

- Let x^* be indicator vector of shortest s-t path
  - Feasible for primal

Proof Using the Dual

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s-t$ path is an optimal solution to the LP.

Primal LP

min $\sum_{e \in E} c_e x_e$
s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$

$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$
s.t.

$$y_v - y_u \leq c_e, \quad \forall (u,v) \in E.$$

- Let x^* be indicator vector of shortest s-t path
  - Feasible for primal
- Let y^*_v be shortest path distance from s to v
  - Feasible for dual (by triangle inequality)

Proof Using the Dual

Claim

When c satisfies the no-negative-cycles property, the indicator vector of the shortest $s-t$ path is an optimal solution to the LP.

Primal LP

min $\sum_{e \in E} c_e x_e$
s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$

$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$
s.t.

$$y_v - y_u \leq c_e, \quad \forall (u, v) \in E.$$

- Let x^* be indicator vector of shortest s-t path
  - Feasible for primal
- Let y_v^* be shortest path distance from s to v
  - Feasible for dual (by triangle inequality)
- $\sum_e c_e x_e^* = y_t^* - y_s^*$, so both x^* and y^* optimal.

Integrality of Polyhedra

A stronger statement is true:

Integrality of Shortest Path LP

The vertices of the polyhedral feasible region are precisely the indicator vectors of simple paths in G.

- Implies that there always exists an optimal solution which is a path whenever LP is bounded and feasible
- Reduces computing shortest path in graphs with no negative cycles to finding optimal vertex of LP

Integrality of Polyhedra

A stronger statement is true:

Integrality of Shortest Path LP

The vertices of the polyhedral feasible region are precisely the indicator vectors of simple paths in G.

Proof

1. LP is bounded iff c satisfies no-negative-cycles
   - $\leftarrow$: previous proof
   - $\rightarrow$: If c has a negative cycle, there are arbitrarily cheap "flows" along that cycle

Integrality of Polyhedra

A stronger statement is true:

Integrality of Shortest Path LP

The vertices of the polyhedral feasible region are precisely the indicator vectors of simple paths in G.

Proof

1. LP is bounded iff c satisfies no-negative-cycles
   - $\leftarrow$: previous proof
   - $\rightarrow$: If c has a negative cycle, there are arbitrarily cheap "flows" along that cycle
2. Fact: For every LP vertex x there is objective c such that x is unique optimal. (Prove it!)

Integrality of Polyhedra

A stronger statement is true:

Integrality of Shortest Path LP

The vertices of the polyhedral feasible region are precisely the indicator vectors of simple paths in G.

Proof

1. LP is bounded iff c satisfies no-negative-cycles
   - $\leftarrow$: previous proof
   - $\rightarrow$: If c has a negative cycle, there are arbitrarily cheap "flows" along that cycle
2. Fact: For every LP vertex x there is objective c such that x is unique optimal. (Prove it!)
3. Since such a c satisfies no-negative-cycles property, our previous claim shows that x is integral.

Integrality of Polyhedra

A stronger statement is true:

Integrality of Shortest Path LP

The vertices of the polyhedral feasible region are precisely the indicator vectors of simple paths in G.

In general, the approach we took applies in many contexts: To show a polytope's vertices integral, it suffices to show that there is an integral optimal for any objective.

Outline

Ford's Algorithm

Primal LP

min $\sum_{e \in E} c_e x_e$

s.t.

$$\sum_{e \to v} x_e - \sum_{v \to e} x_e = \delta_v, \quad \forall v \in V.$$

$$x_e \geq 0, \quad \forall e \in E.$$

Dual LP

max $y_t - y_s$

s.t.

$$y_v - y_u \leq c_e, \quad \forall e = (u, v) \in E.$$

For convenience, add (s, v) of length ∞ when one doesn't exist.

Ford's Algorithm

1. $y_v = c_{(s,v)}$ and $pred(v) \leftarrow s$ for $v \neq s$
2. $y_s \leftarrow 0$, $pred(s) = null$.
3. While some dual constraint is violated, i.e. $y_v > y_u + c_e$ for some $e = (u, v)$
   - $y_v \leftarrow y_u + c_e$
   - Set $pred(v) = u$
4. Output the path $t, pred(t), pred(pred(t)), \ldots, s$.

Correctness

Lemma (Loop Invariant 1)

Assuming no negative cycles, $pred$ defines a path P from s to t, of length at most $y_t - y_s$.

Interpretation

- Ford's algorithm maintains an (initially infeasible) dual y
- Also maintains feasible primal P of length $\leq$ dual objective $y_t - y_s$
- Iteratively "fixes" dual y, tending towards feasibility
- Once y is feasible, weak duality implies P optimal.

Correctness

Lemma (Loop Invariant 1)

Assuming no negative cycles, $pred$ defines a path P from s to t, of length at most $y_t - y_s$.

Interpretation

- Ford's algorithm maintains an (initially infeasible) dual y
- Also maintains feasible primal P of length $\leq$ dual objective $y_t - y_s$
- Iteratively "fixes" dual y, tending towards feasibility
- Once y is feasible, weak duality implies P optimal.

Correctness follows from loop invariant 1 and termination condition.

Theorem (Correctness)

If Ford's algorithm terminates, then it outputs a shortest path from s to t

Correctness

Lemma (Loop Invariant 1)

Assuming no negative cycles, $pred$ defines a path P from s to t, of length at most $y_t - y_s$.

Interpretation

- Ford's algorithm maintains an (initially infeasible) dual y
- Also maintains feasible primal P of length $\leq$ dual objective $y_t - y_s$
- Iteratively "fixes" dual y, tending towards feasibility
- Once y is feasible, weak duality implies P optimal.

Correctness follows from loop invariant 1 and termination condition.

Theorem (Correctness)

If Ford's algorithm terminates, then it outputs a shortest path from s to t

Algorithms of this form, that output a matching primal and dual solution, are called Primal-Dual Algorithms.

Termination

Lemma (Loop Invariant 2)

Assuming no negative cycles, y_v is the length of some simple path from s to v.

Termination

Lemma (Loop Invariant 2)

Assuming no negative cycles, y_v is the length of some simple path from s to v.

Theorem (Termination)

When the graph has no negative cycles, Ford's algorithm terminates in a finite number of steps.

Proof

- The graph has a finite number N of simple paths
- By loop invariant 2, every dual variable y_v is the length of some simple path.
- Dual variables are nonincreasing throughout algorithm, and one decreases each iteration.
- There can be at most nN iterations.

Observation: Single sink shortest paths

Ford's Algorithm

1. $y_v = c_{(s,v)}$ and $pred(v) \leftarrow s$ for $v \neq s$
2. $y_s \leftarrow 0$, $pred(s) = null$.
3. While some dual constraint is violated, i.e. $y_v > y_u + c_e$ for some $e = (u, v)$
   - $y_v \leftarrow y_u + c_e$
   - Set $pred(v) = u$
4. Output the path $t, pred(t), pred(pred(t)), \ldots, s$.

Observation

Algorithm does not depend on t till very last step. So essentially solves the single-source shortest path problem. i.e. finds shortest paths from s to all other vertices v.

Loop Invariant 1

We prove Loop Invariant 1 through two Lemmas

Lemma (Loop Invariant 1a)

For every node w, we have $y_w - y_{pred(w)} \geq c_{pred(w),w}$

Proof

- Fix w
- Holds at first iteration
- Preserved by Induction on iterations
  - If neither y_w nor $y_{pred(w)}$ updated, nothing changes.
  - If y_w (and $pred(w)$) updated, then $y_w \leftarrow y_{pred(w)} + c_{pred(w),w}$
  - $y_{pred(w)}$ updated, it only goes down, preserving inequality.

Loop Invariant 1

Lemma (Invariant 1b)

Assuming no negative cycles, pred forms a directed tree rooted out of s.

We denote this path from s to a node w by $P(s, w)$.

Proof

- Holds at first iteration
- For a contradiction, consider iteration of first violation
  - v and u with $y_v > y_u + c_{u,v}$
- $P(s, u)$ passes through v
  - Otherwise tree property preserved by $pred(v) \leftarrow u$
- Let $P(v, u)$ be the portion of $P(s, u)$ starting at v.
- By Invariant 1a, and telescoping sum, length of $P(v, u)$ is at most $y_u - y_v$.
- Length of cycle $\{P(v, u), (u, v)\}$ at most $y_u - y_v + c_{u,v} < 0$.

Summarizing Loop Invariant 1

Lemma (Invariant 1a)

For every node w, we have $y_w - y_{pred(w)} \geq c_{pred(w),w}$.

- By telescoping sum, can bound $y_w - y_s$ when pred leads back to s

Lemma (Invariant 1b)

Assuming no negative cycles, pred forms a directed tree rooted out of s.

- Implies that y_s remains 0

Corollary (Loop Invariant 1)

Assuming no negative cycles, $pred$ defines a path $P(s, w)$ from s to each node w, of length at most $y_w - y_s = y_w$.

Loop Invariant 2

Lemma (Loop Invariant 2)

Assuming no negative cycles, y_w is the length of some simple path $Q(s, w)$ from s to w, for all w.

Proof is technical, by induction, so we will skip. Instead, we will modify Ford's algorithm to guarantee polynomial time termination.

The following algorithm fixes an (arbitrary) order on edges E

Bellman-Ford Algorithm

1. $y_v = c_{(s,v)}$ and $pred(v) \leftarrow s$ for $v \neq s$
2. $y_s \leftarrow 0$, $pred(s) = null$.
3. While y is infeasible for the dual
   - For $e = (u, v)$ in order, if $y_v > y_u + c_e$ then
     - $y_v \leftarrow y_u + c_e$
     - Set $pred(v) = u$
4. Output the path $t, pred(t), pred(pred(t)), \ldots, s$.

Bellman-Ford Algorithm

The following algorithm fixes an (arbitrary) order on edges E

Bellman-Ford Algorithm

1. $y_v = c_{(s,v)}$ and $pred(v) \leftarrow s$ for $v \neq s$
2. $y_s \leftarrow 0$, $pred(s) = null$.
3. While y is infeasible for the dual
   - For $e = (u, v)$ in order, if $y_v > y_u + c_e$ then
     - $y_v \leftarrow y_u + c_e$
     - Set $pred(v) = u$
4. Output the path $t, pred(t), pred(pred(t)), \ldots, s$.

Note

Correctness follows from the correctness of Ford's Algorithm.

Runtime

Theorem

Bellman-Ford terminates after $n - 1$ scans through E, for a total runtime of $O(nm)$.

Runtime

Theorem

Bellman-Ford terminates after $n-1$ scans through E, for a total runtime of $O(nm)$.

Follows immediately from the following Lemma

Lemma

After k scans through E, vertices v with a shortest $s-v$ path consisting of $\leq k$ edges are correctly labeled. (i.e., $y_v = distance(s,v)$)

Proof

Lemma

After k scans through E, vertices v with a shortest $s - v$ path consisting of $\leq k$ edges are correctly labeled. (i.e., $y_v = distance(s, v)$)

Proof

- Holds for $k = 0$
- By induction on k.
  - Assume it holds for $k - 1$.
  - Let v be a node with a shortest path P from s with k edges.
  - $P = \{Q, e\}$, for some $e = (u, v)$ and $s - u$ path Q, where Q is a shortest $s - u$ path and Q has $k - 1$ edges.
  - By inductive hypothesis, u is correctly labeled just before e is scanned – i.e. $y_u = distance(s, u)$.
  - Therefore, v is correctly labeled $y_v \leftarrow y_u + c_{u,v} = distance(s, v)$ after e is scanned

A Note on Negative Cycles