Introduction to Mathematical Programming
IE406

Lecture 18

Dr. Ted Ralphs

Reading for This Lecture

- Bertsimas 7.4-7.6

Negative Cost Cycle Algorithm

- Recall the negative cost cycle algorithm from last lecture.
- The basic algorithm consisted of iteratively saturating unsaturated negative cost cycles.
- Questions to be answered
  - How do we start the algorithm (find a feasible flow)?
  - How do we find unsaturated, negative cost cycles?
  - If there are no unsaturated negative cost cycles, is the current solution optimal?
  - Is the algorithm guaranteed to terminate?

The Residual Network

- We are only interested in arcs on which flow can be pushed (forwards or backwards).
- Corresponding to a given flow f, we build a *residual network*.
- For each arc (i, j) in the original network, we introduce two arcs into the residual network:
  - an arc (i, j) with capacity $u_{ij} - f_{ij}$ and cost c_{ij}, and
  - an arc (j, i) with capacity f_{ij} and cost $-c_{ij}$.
- We delete arcs that have zero capacity from the network.
- Flow on the arc (j, i) in the residual network can be interpreted as a reduction in the flow on arc (i, j) in the original network.

Interpreting the Residual Network

- Consider a feasible flow f and another feasible flow $f + \bar{f}$.
- The *flow increment* $\bar{f}$ can be associated with a flow $\tilde{f}$ in the residual network.
  - $\bar{f}_{ij} > 0 \Rightarrow \tilde{f}_{ij} = \bar{f}_{ij}$,
  - $\bar{f}_{ij} < 0 \Rightarrow \tilde{f}_{ji} = -\bar{f}_{ij}$,
- In the residual network, all the flows are positive.
- $\tilde{f}$ is a circulation (why?).
- The cost of $\tilde{f}$ in the residual network is the same as $\bar{f}$ in the original network.

Using the Residual Network

- Starting with a circulation in the residual network, we can construct a corresponding flow increment in the original network.
- Unsaturated cycles in the original network correspond to directed cycles in the residual network.
- Hence, we have only to find a negative cost directed cycle in the residual network.
- This can be done efficiently.

Optimality Conditions

Theorem 1. *A feasible flow f is optimal if and only if there is no unsaturated cycle of negative cost.*

- One direction of the proof is easy (which one?).
- The other direction depends on the flow decomposition theorem.
- The flow decomposition theorem states that every nonzero circulation is a positive linear combination of simple circulations.

The Algorithm

- The negative cost cycle method:

  1. Start with a feasible flow f.
  2. Search for an unsaturated cycle C with negative cost.
  3. If no such cycle can be found, the current flow is optimal.
  4. Otherwise,
     - If $\delta(C) < \infty$, construct the new feasible flow $f + \delta(C)h^C$ and iterate.
     - If $\delta(C) = \infty$, the optimal cost is $-\infty$.

  Theorem 2. *Suppose that all the arc capacities u_{ij} are integer or infinite, and that the negative cost cycle algorithm is initialized with a feasible flow. Then, the flow variables remain integral throughout the algorithm and, if the optimal cost is finite, the algorithm terminates with an integer optimal solution.*

- The algorithm is not guaranteed to terminate if the arc capacities are not integral unless using special rules for selecting the cycles.

The Maximum Flow Problem

- In this problem, we are given a directed graph $G = (N, A)$ and a positive capacity on each arc.
- Two nodes s and t, called the *source* and the *sink* are specified.
- We want to find the maximum amount of flow we can route from s to t.
- This model has numerous applications.
- This problem is a special case of the network flow problem (why?), but special purpose algorithms can solve it quickly.

Augmenting Paths

Definition 1. *Given a feasible flow f, an augmenting path is a path from s to t such that $f_{ij} < u_{ij}$ for all forward arcs and $f_{ij} > 0$ for all backward arcs in the path.*

- Note that if we can find an augmenting path, we can increase the current flow from s to t.
- We do this by pushing flow along the path P. How much can we push?

$$\delta(P) = ?$$

- Notice that augmenting paths are negative cycles in a related network.

The Ford-Fulkerson Algorithm

1. Start with a feasible flow f.
2. Search for an augmenting path.
3. If no augmenting path exists, then the current flow is optimal.
4. If an augmenting path P is found, then:
   - If $\delta(P) < \infty$, push $\delta(P)$ units of flow along P.
   - If $\delta(P) = \infty$, the maximum flow is ∞.

Theorem 3. *Suppose that all the arc capacities u_{ij} are integer or infinite, and that the Ford-Fulkerson algorithm is initialized with an integral flow. Then, the flow variables remain integral throughout the algorithm and, if the optimal cost is finite, the algorithm terminates with an integer optimal solution.*

- The algorithm is not guaranteed to terminate if the arc capacities are not rational.

Finding an Augmenting Path

- Finding an augmenting path is easy.
- We can build a residual network as before.
- Any directed path from s to t is an augmenting path.
- To avoid building the residual network, a simple labeling algorithm can be used.
- At each step, we label nodes that can be reached via an unsaturated path through the network.

Cuts in a Graph

- A $s-t$ *cut* in a graph is defined as a subset S of the nodes N such that $s \in S$ and $t \notin S$.
- The *capacity* of a cut is the sum of the capacities of the arcs crossing from S to its complement.

$$C(S) = \sum_{\{(i,j) \in A | i \in S, j \notin S\}} u_{ij}$$

- Note that if we remove the set of edges crossing from S to its complement from the graph, we disconnect it.
- Any flow from s to t must cross from S to its complement.
- Therefore, the value v of any feasible flow must satisfy

$$v \leq C(S)$$

for every cut S.

The Max-flow Min-cut Theorem

Theorem 4. *The value of the maximum flow in a graph is always equal to the capacity of a minimum cut.*

Proof:

Duality and Sensitivity in Network Flow

- The dual variables are easy to interpret for network flow problems.
- The i^{th} dual variable can be interpreted as the cost of routing the commodity from node i to node n in the network.
- If A is the set of arcs in our network, the constraints in the dual state

$$p_i - p_j \leq c_{ij} \; \forall (i,j) \in A.$$

- This says that the difference between the costs for routing commodities from nodes i and j should be no more than the cost of arc (i,j).
- If not, we could have achieved a lower cost by using arc (i,j) to route commodity from node i.
- The complementary slackness conditions state that for any arc (i,j) that has flow on it, we must have $p_i - p_j = c_{ij}$.
- Otherwise, we could get a better solution by avoiding arc (i,j).