Reading for This Lecture

- Papadimitriou and Steiglitz, Chapters 5 and 6.
The Assignment Problem

- The assignment problem can be interpreted as that of assigning $n$ items to $n$ people so as to maximize the total "value" of the assigned items.

- An LP formulation is as follows:

$$\begin{align*}
\max & \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} f_{ij} \\
\text{s.t.} & \sum_{i=1}^{n} f_{ij} = 1, \quad j = 1, \ldots, n \\
& \sum_{j=1}^{n} f_{ij} = 1, \quad i = 1, \ldots, n \\
& f_{ij} \geq 0, \forall i, j
\end{align*}$$

- Here, $c_{ij}$ can be interpreted as the value of item $i$ to person $j$.

- Note that this can be interpreted as a network flow problem, so there always exists an optimal solution for which $f_{ij} \in \{0, 1\}$.

- This allows us to interpret the solution as an assignment.
The Dual of the Assignment Problem

• The dual problem has the following form:

\[
\min \sum_{i=1}^{n} p_j + \sum_{j=1}^{n} r_i \\
\text{s.t.} \quad r_i + p_j \geq c_{ij}, \forall i, j.
\]

• Here, we will interpret \( r_i \) as the price of item \( i \) and \( p_j \) as the person profit of person \( j \).

• In order to minimize \( \sum_{i=1}^{n} r_i \), we must have

\[
r_i = \max_{j=1,\ldots,n} \{c_{ij} - p_j\}
\]

• Hence, we can rewrite the dual as

\[
\min \left( \sum_{j=1}^{n} p_j + \sum_{i=1}^{n} \max_j \{c_{ij} - p_j\} \right)
\]
• This is an unconstrained optimization problem with a piecewise concave objective function.
The Complementary Slackness Conditions

- The complementary slackness conditions tell us that

\[ f_{ij} > 0 \Rightarrow r_i + p_j = c_{ij} \]

- Substituting the previous form for \( r_i \), we get

\[ f_{ij} > 0 \Rightarrow c_{ij} - p_j = \max_k \{c_{ik} - p_k\} \]

- In other words, this says that each person should be assigned the item that maximizes their personal profit.

- This leads to an algorithm simulating an auction, in which we envision each person bidding for items in multiple rounds.
An Auction Algorithm

- We will assume that the costs are integral.
- Given a set of (integer) prices to be paid for the items, each person offers to buy the items that would maximize their personal profit.
- Let the set of items desired on by person $j$ be $D_j$.
- The auctioneer attempts to allocate all the items to people such that everyone ends up with an item they desire.
- If this works, then we know that complementary slackness, as well as primal and dual feasibility are satisfied and we have the optimal solution.
**Updating the Prices**

- If there is no feasible assignment at the current prices, we decrease the prices on all projects that were not desired on by $1 and start another round of bidding.

- **Question**: Will this work?

- **Answer**: Yes.

- This is a special case of a more general algorithm called the *primal-dual* algorithm.

- Note that if there is not feasible assignment, then there must be a set of items that is “overdemanded,” i.e., a subset $T$ of the players such that

  $$|\bigcup_{j \in T} D_j| < |T|$$

- Increasing the prices on the overdemanded items by $1 also works.
The Primal-Dual Algorithm

- The **primal-dual algorithm** can be used to solve general linear programs.

- Suppose we have an LP in standard form and assume without loss of generality that $b \geq 0$.

- We start with a feasible dual solution and try to construct a primal solution that obeys complementary slackness.

- This is done by attempting to solve $Ax = b$ with only the variables having zero reduced cost allowed to enter the basis.

- If we succeed, then the primal solution is optimal.

- Otherwise, we change the dual prices and continue.
Implementing the Primal-Dual Algorithm

• Beginning with a feasible dual solution, the first step is to attempt to find a primal solution satisfying complementary slackness.

• We can do this by setting up a Phase I LP, called the restricted primal, in which only the variables with reduced cost zero are present.

$$\begin{align*}
\min & \quad \sum_{i=1}^{m} y_i \\
\text{s.t.} & \quad \sum_{j \in J} a_{ij} x_j + y_i = b_i \quad \forall i \in 1, \ldots, m \\
& \quad x_j \geq 0 \quad \forall j \in J \\
& \quad y_i \geq 0 \quad \forall i \in 1, \ldots, m
\end{align*}$$

• If this LP has an optimal value of zero, we are done.
**Updating the Prices**

- If the restricted primal does not have an optimal value of zero, we must **update the dual prices**.

- The dual of this LP is

  \[
  \begin{align*}
  & \text{max} & p^\top b \\
  & \text{s.t.} & p^\top A_j \leq 0 \quad \forall j \in J \\
  & & p_i \leq 1 \quad \forall i \in 1, \ldots, m
  \end{align*}
  \]

- Note that this LP finds the feasible direction with maximum cost increase for the dual.

- We go as far as we can in this direction.
Comments on the Primal-Dual Algorithm

- This algorithm follows an improving search paradigm.
- It is a dual ascent algorithm like the dual simplex algorithm.
- Like dual simplex, we maintain dual feasibility and look for a complementary primal feasible solution.
- In the absence of primal degeneracy, the algorithms is guaranteed to terminate finitely using an argument similar to that for the simplex algorithm.
- Just as in simplex, anticycling rules can be used to deal with degeneracy.
- This algorithm can be viewed essentially as a column generation algorithm for the primal problem.
- After updating the dual prices, we add some columns with negative reduced cost and resolve from the previous basis.
- We can always do this because any column that was basic in the previous iteration must still have reduced cost zero after the update.
The Shortest Path Problem

- We are given a directed graph $G = (N, A)$ and a cost or length associated with each arc.
- We define the length of a path to be the sum of the lengths of the arcs in the path.
- The basic shortest path problem is that of finding the path of minimum length between a given origin and a given destination.
- This is equivalent to a certain minimum cost flow problem (why?).
Shortest Paths Trees

- A tree that consists of a directed path from nodes $1, \ldots, n - 1$ to node $n$ is called an *intree rooted at node $n$*.  

- An intree that consists of the shortest paths from nodes $1, \ldots, n - 1$ to node $n$ is called a *shortest paths tree*.  

- As long as there are no negative length cycles, calculating a shortest paths tree is equivalent to an uncapacitated minimum cost network flow problem with  
  - a supply of 1 at nodes $1, \ldots, n - 1$, and  
  - a demand of $n - 1$ at node $n$.  

- Furthermore, assuming $p^*_n = 0$, the unique solution to the dual problem consists of assigning  

$$p^*_i = \text{the path length from node } i \text{ to node } n.$$
Label Correcting Methods and Dijkstra’s Algorithm

- Applying the primal-dual algorithm to the shortest path problem yields a class of algorithms called \textit{Label correcting methods}.

- \textit{Dijkstra’s Algorithm} is a simple algorithm that can be applied when all arc costs are nonnegative.

- \textbf{Algorithm}

  1. Find a node \( l \neq n \) such that \( c_{ln} \leq c_{in} \) for all \( i \neq n \).
  2. For every node \( i \neq l, n \), set

      \[
      c_{in} := \min\{c_{in}, c_{il} + c_{ln}\}
      \]

  3. Remove node \( l \) from the graph and apply the same steps to the new graph.
Ford-Fulkerson Algorithm Revisited

- The Ford-Fulkerson Algorithm for maximum flow can also be viewed as an implementation of the primal-dual algorithm.
- In this case, the primal problem in the minimum cut problem and the dual variables are the flow variables.
- Primal feasibility consists of determining whether there exists a cut using only forward arcs that are saturated and backward arcs that have flow zero.
- The dual update is to find an augmenting path.
- Showing that this actually is an implementation of the primal-dual algorithm is a little messy.
- The general minimum cost network flow problem can also be cast as a primal-dual algorithm.