CS 435 : Linear Optimization Fall 2006

Lecture 26: Primal-Dual algorithm for shortest path problem

Lecturer: *Sundar Vishwanathan* Scribe: *Pragya Goyal*
Computer Science & Engineering Indian Institute of Technology, Bombay

1 Recap of last lecture

In last class we had formed the ILP for the Shortest Path problem as follows:

$$\min \sum_e w_{uv} x_{uv} \tag{1}$$

$$\sum_u x_{su} = 1 \tag{2}$$

$$\sum_v x_{vt} = 1 \tag{3}$$

$$\forall p \sum_q x_{pq} - \sum_r x_{rp} = 0 \tag{4}$$

$$1 \geq x_{uv} \geq 0 \tag{5}$$

$$x_{uv} \text{is an integer} \tag{6}$$

Expression(1)is the summation of weight of all the edges selected, where x_i takes value 1 if edge (u, v) is selected otherwise remains 0. Equations (2) and (3), represent the constraints that in the shortest path the source s must have exactly one edge leaving it and destination t must have exactly one edge entering into it, respectively. Equation (4) is written to satisfy the constraint that all the intermediate vertices (i.e. apart from s and t) in the path must have one edge entering and one edge leaving them. Inequality(5) is to ensure that x_{uv} takes values in $[0, 1]$. To reduce the problem to LP we drop the $x_{uv} \leq 1$ constraint. This can be done because equalities (2) and (3) along with x_{uv} constraint ensure that each x_{uv} does not take value greater than 1.

Now let us look at the dual. We will be solving the dual and will observe that it is Dijkstra's algorithm.

2 Solving the Dual

In the dual all the constraints of Primal are represented by variables. Here we have one variable for each vertex. Hence dual can be represented as:

$$\max\ y_s - y_t \tag{7}$$

$$\forall u, v \qquad y_v - y_u \leq w_{uv} \tag{8}$$

Now let us try solving the dual. Here y_i can be thought of variable corresponding to ith vertex. To maximize $y_s - y_t$ let us begin with assuming $y_t = 0$. Keeping y_t fixed at 0 we raise all other variables by 1 at a time until for atleast one of them, say y_1, $y_1 - y_t = w_{1t}$. at this point stop increasing the value of y_1 and fix it to the last reached value. Repeat the procedure keeping y_t and y_1 fixed now. Include any

new vertex y_2 as soon as $y_2 - y_1$ equals w_{12}. We iterate over all vertex variables until we reach y_s. We stop the iteration when y_s is reached.

Marble-string analogy:
The procedure is similar to fixing a marble tied to many strings and then pulling the strings apart. Constraint being limited length of the strings. Fixed marble is a vertex in this case, strings are the edges entering the vertex and string lengths correspond to edge weights. We can observe here that the string with least length (which corresponds to least weight edge entering fixed vertex) will become taut first. Becoming taut means equality is satisfied in the constraint. Select this edge. Now fix the vertex from which this edge emanates. Apply similar raising method for this vertex. We iterate over the vertices until we reach the source vertex. The edges selected at each iteration form the shortest path from s to t. This algorithm is **Dijkstra's algorithm**.

2.1 Reverse Delete

The edges we selected in the above procedure contain the shortest path along with some additional edges. To remove those edges we apply the procedure of *Reverse Delete*. This is done by starting deletion of maximum weight edges, each time assuring that a path exists from s to t. As soon as we reach an edge, deleting which leaves no path from s to t, we stop the procedure. The path left (including this last egde seen) is the required shortest path.