CS602 Applied Algorithms **2019-20 Sem II**

Lecture 9: February 4

Scribe: Parashiv Sihaniya *Lecturer: Rohit Gurjar*

Shortest path in a directed graph (with non-negative weights)

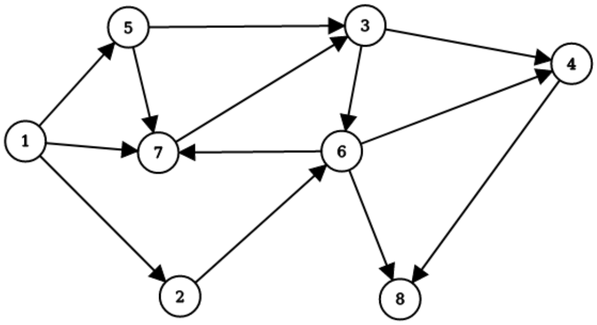


Figure 1: A directed graph

$$
\begin{aligned}
&E = \text{ Set of Edges, } |E| = m\\
&\text{Path } P = \text{ Subset of } E\\
&x_1, x_2, \ldots x_m \in \{0,1\} \text{ represents arbitrary subset of edges}\\
&x_i = \begin{cases} 0 & i\text{-th edge NOT in the subset} \\ 1 & i\text{-th edge in the subset}\end{cases}
\end{aligned}
$$

We can write the following integer program for the shortest path proplem. Let w_e be the weight of edge e for each $e \in E$.

$$\text{minimize} \quad \sum_{e=1}^{m} w_e x_e \quad \text{s.t.}$$

$$
\begin{array}{rcl}
\text{Conditions for } x \text{ to represent a path } (s \to t): & & \\
x_e & \in & \{0,1\} \\
\sum_{e \in \text{OUT}(s)} x_e & = & 1, \\
\sum_{e \in \text{IN}(t)} x_e & = & 1, \\
\sum_{e \in \text{IN}(u)} x_e = \sum_{e \in \text{OUT}(u)} x_e & \leq & 1, \text{ for } u \notin s, t
\end{array}
$$

Observation 9.1. *Any 0-1 solution which minimizes this must be an $(s \to t)$ path. The optimal 0-1 solution gives us a minimum weight path.*

Note that since it is a minimization program, the last inequality is redundant, and thus, can be ignored.

We can write the following LP relaxation for the above integer program:

$$\begin{array}{rcll} \text{minimize} \sum_{e \in E} w_e x_e & & \text{s.t.} & \\ \sum_{e \in \text{IN}(u)} x_e & = & \sum_{e \in \text{OUT}(u)} x_e, & \text{for } u \notin s, t \\ 0 & \leq & x_e & \\ \sum_{e \in \text{OUT}(s)} x_e & = & 1, & \\ \sum_{e \in \text{IN}(t)} x_e & = & 1 & \end{array}$$

There can be non-integral solutions for this LP. For example, see Figure 2. Interestingly, we can show that there is always an integral optimal solution, and thus, the LP gives us the correct shortest path length.

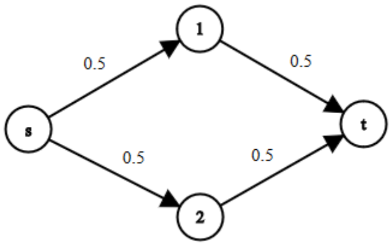


Figure 2: A feasible solution

Theorem 9.2. *There is always an integral optimal solution.*

Claim 9.3. *If there is a non-integral feasible solution with weight w then there is also an integral solution with weight at most w.*

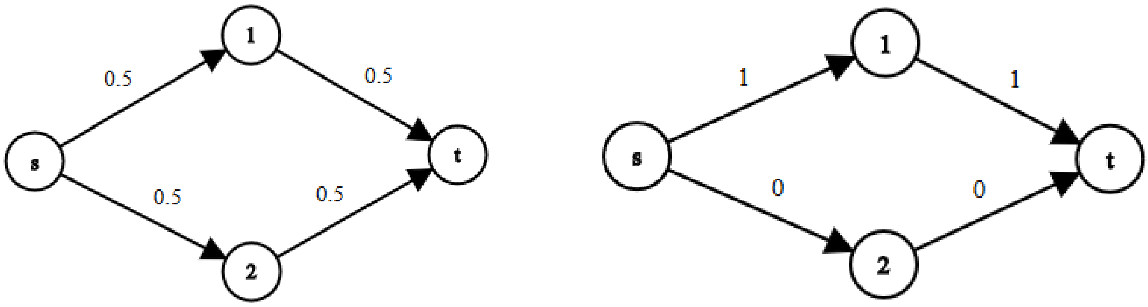
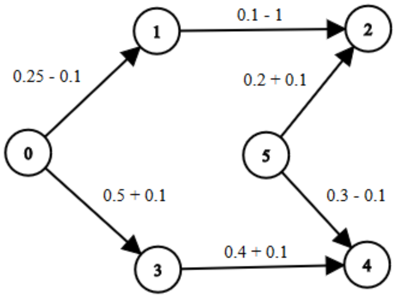


Figure 3: The left figure shows a fractional feasible solution. The integral solution corresponding to one of the two paths (the right figure) must have a smaller or same weight value.

Proof of Claim 9.3:

1. If x_e is non-integral for an edge e, then some neighbouring edge must be also non-integral. This is because the LP constraints say that the difference of the outgoing and incoming edges is integral.
2. This way we can get a cycle of non-integral edges (ignoring directions).
3. Now, one can add or subtract a number ϵ to the cycle edges so that the difference of the outgoing and incoming sums remains unchanged (see Figure 4).
4. Choose ϵ so that at least one edge variable becomes integral.



There are two possible ways of changing the values. The total change in the value of the solution is given by Δw and $\Delta w'$, respectively.

$$\Delta w = (-w_1 + w_2 - w_3 + w_4 + w_5 - w_6) \times 0.1$$

$$\Delta w' = (+w_1 - w_2 + w_3 - w_4 - w_5 + w_6) \times \epsilon$$

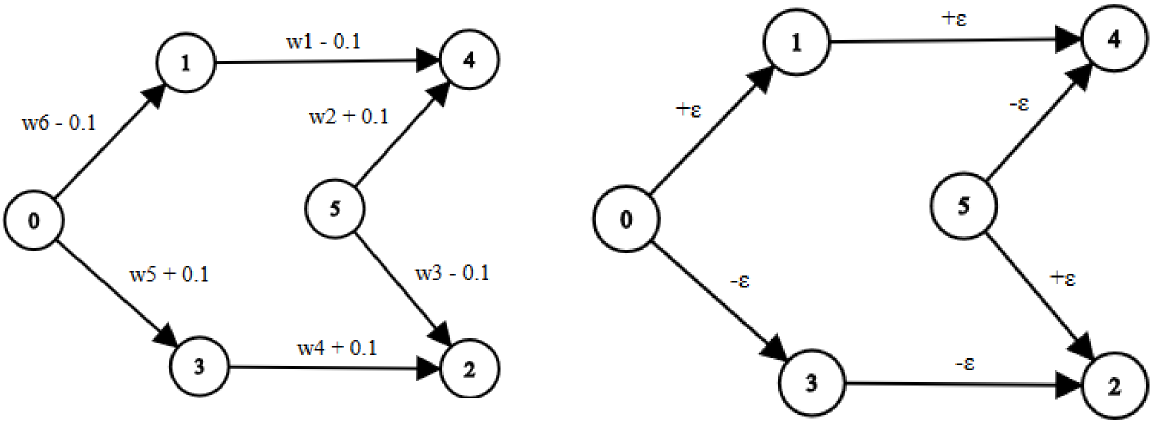


Figure 4: Two ways to alter the edge values, so that the vertex constraints remain satisfied.

Note that $sign(\Delta w) = -sign(\Delta w')$. Since Δw and $\Delta w'$ have opposite signs, we can choose the direction which decreases the function value (or keeps it the same). This way we get a new feasible solution with at least one more integral coordinate. Keep repeating this procedure till all coordinates become integral. In the end, we will have an integral feasible solution that has a better (or same) function value as the starting solution.

Theorem 9.4. *This LP is correct for shortest paths. In general, LP can have a fractional optimal which is strictly better than integral optimal.*

Dual LP

- Variables - Constraints in LP
- For every primal constraint → Dual Variable
- Don't need non-negative constraints on dual variables, since we have equalities in the primal.
- For every primal variable → Dual constraint
- ∴ One constraint for every edge

$$\begin{aligned} \max\ y_t - y_s \quad & \text{s.t.} \\ y_v - y_u \quad \leq \quad & w_e \text{ for each } e = (u,v) \in E. \end{aligned}$$

Think of an embedding of vertices on the real line and y_u as the real coordinate of vertex u. The objective is to maximize the distance between the source and the destination, while each vertex pair distance should be upper bounded by the corresponding edge weight. Dijkstra's algorithm essentially solves this LP. You can also think of this as follows: say for every edge $e = (u,v)$, the two endpoints u and v are tied by a thread of length w_e and you are trying to pull s and t as far apart as possible. This will basically tighten all the threads corresponding to edges that lie on a shortest path and the distance between s and t will be the shortest (s,t)-distance in the graph.