3 The Shortest Path

3.1 The Primal-Dual Method

Consider the standard linear programming

$$(P) \qquad \begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax = b \geq 0 \\ & x \geq 0 \end{array}$$

and its dual

$$(D) \qquad \begin{array}{ll} \max & \pi^T b \\ \text{s.t.} & \pi^T A \leq c^T \,. \end{array}$$

Suppose that we have a current π which is feasible to the dual problem (D). Define the index set J by

$$J = \{j : \pi^T A_j = c_j\}\,,$$

where A_j is the jth column of A. Then for any $j \notin J$, we have $\pi^T A_j < c_j$. We call J the set of admissible columns. In order to search for an x such that it is not only feasible to the primal problem (P) but also it, togther with π, satisfies the complementary condition of (P) and (D), we invent a new LP, called the restricted primal (RP), as follows

$$(RP) \qquad \begin{array}{rcll} \xi^* & = & \min & \displaystyle\sum_{i=1}^{m} x_i^a \\ & & \text{s.t.} & Ax + x^a = b \\ & & & x_j \geq 0\,,\ \text{for all } j\,, \\ & & & x_j = 0\,,\ j \notin J\,, \\ & & & x_i^a \geq 0\,,\ i = 1, \ldots, m\,, \end{array}$$

i.e.,

$$(RP) \qquad \begin{aligned} \xi^* = \quad & \min \quad 0^T x_J + \sum_{i=1}^m x_i^a \\ & \text{s.t.} \quad A_J x_J + x^a = b \\ & \qquad\quad x_J \geq 0, x^a \geq 0\,. \end{aligned}$$

The dual of (RP) is

$$(DRP) \qquad \begin{aligned} w^* = \quad & \max \quad \pi^T b \\ & \text{s.t.} \quad \pi^T A_j \leq 0,\ j \in J \\ & \qquad\quad \pi_i \leq 1,\ i = 1, \dots, m\,. \end{aligned}$$

Let $(\bar{x}_J, \bar{x}^a)$ be an optimal basic feasible solution to (RP) and $\bar{\pi}$ be an optimal basic feasible solution to (DRP) obtained from $(\bar{x}_J, \bar{x}^a)$. If $w^* = 0$, then $\xi^* = 0$. Such an x is found. Otherwise, $w^* > 0$ and we can update π to

$$\pi^{\text{new}} = \pi + \theta\bar{\pi}\,.$$

The new cost to (D) is

$$\boxed{(\pi^{\text{new}})^T b = \pi^T b + \theta\bar{\pi}^T b = \pi^T b + \theta w^*,}$$

which means that we shall get a better π if we can take $\theta > 0$. On the other hand, π^{new} should be feasible to (D), i.e.,

$$\boxed{(\pi^{\text{new}})^T A_j = \pi^T A_j + \theta\bar{\pi}^T A_j \leq c_j\,.}$$

Since for every $j \in J$, $\bar{\pi}^T A_j \leq 0$, we only need to consider those $\bar{\pi}^T A_j > 0$, $j \notin J$. Therefore, we can take

$$\theta \;=\; \min_{\substack{j \notin J \\ \text{such that} \\ \bar{\pi}^T A_j > 0}} \; \frac{c_j - \pi^T A_j}{\bar{\pi}^T A_j}\,.$$

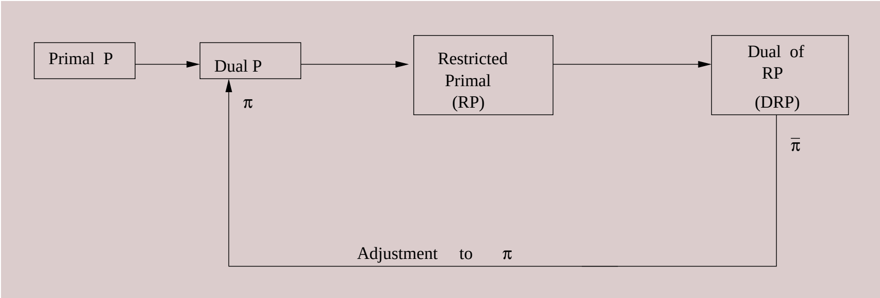


Figure 3.1: An illustration of the prima-dual method

3.2 The Primal-Dual Method for the Shortest Path Problem

Let $\tilde{A}$ be the incidence matrix of the digraph $G = (V, E)$, where $V = \{1, \ldots, m\}$ and $E = \{e_1, \ldots, e_n\}$. With each arc e_j we associate its length $c_j \geq 0$ and its flow $x_j \geq 0$. The shortest path problem, as we have already known, may be formulated as:

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} c_j x_j, \\ \text{s.t.} \quad & \tilde{A}x = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \\ +1 \end{bmatrix}, \\ & x \geq 0\,. \end{aligned} \tag{3.1}$$

Let $\bar{A}$ be the remaining submatrix of $\tilde{A}$ by removing the last row of $\tilde{A}$ (it is redundant because the sum of all rows of $\tilde{A}$ is zero). Then (3.1) turns into

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} c_j x_j , \\
\text{s.t.} \quad & \bar{A}x = \begin{bmatrix} -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \\
& x \geq 0\,.
\end{aligned}
\tag{3.2}
$$

The dual problem to (3.2) is

$$
\begin{aligned}
\max \quad & -\pi_1 \\
\text{s.t.} \quad & -\pi_i + \pi_j \leq c_{ij} \quad \text{for all } (i,j) \in E, \\
& \pi_m = 0\,,
\end{aligned}
\tag{3.3}
$$

where we must fix $\pi_m = 0$ because the last row of $\tilde{A}$ is omitted in $\bar{A}$.

The idea of **primal-dual algorithm** is derived from the idea of searching for a feasible point x such that

$$\boxed{x_{ij} = 0 \text{ (some } x_k\text{) whenever } -\pi_i + \pi_j < c_{ij}\,,}$$

for given feasible π (Remark: think about complementary conditions). We search for such an x by solving an auxiliary problem, called the **restricted primal (RP)**, determined by the π we are working with. If our search for the x is not successful, we nevertheless obtain information from the dual of RP, which we call **DRP**, and tells us how to improve the particular π with which we started.

Next, we give the details. The shortest-path problem can be written as

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{n} c_j x_j, \\
\text{s.t.} \quad & Ax = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \\
& x \geq 0\,,
\end{aligned}
\tag{3.4}
$$

where $A = -\bar{A}$. The purpose of introducing A is to make the right hand side of the constraint $Ax = b$ nonnegative. Now, the dual problem of (3.4) is

$$
\begin{aligned}
\max \quad & \pi_1 \\
\text{s.t.} \quad & \pi_i - \pi_j \leq c_{ij} \quad \text{for all } (i,j) \in E, \\
& \pi_m = 0\,.
\end{aligned}
\tag{3.5}
$$

For a given feasible π to (3.5), the set of admissible arcs is defined by

$$
J \;=\; \{\text{arcs } (i,j) :\; \pi_i - \pi_j \;=\; c_{ij}\}\,.
$$

The corresponding restricted primal problem (RP) is

$$
\begin{aligned}
\xi^* \;=\; \min \quad & \sum_{i=1}^{m-1} x_i^a, \\
\text{s.t.} \quad & Ax + x^a = \begin{bmatrix} +1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \\
& x_j \geq 0\,, \text{ for all } j\,, \\
& x_j = 0\,,\; j \notin J\,, \\
& x_i^a \geq 0\,,\; i = 1, \ldots, m-1
\end{aligned}
\tag{3.6}
$$

and the dual of the restricted primal (DRP) is

$$
\begin{aligned}
w^* \;=\; \max \quad & \pi_1 \\
\text{s.t.} \qquad & \pi_i - \pi_j \le 0 \quad \text{for all } (i,j) \in J\,, \\
& \pi_i \le 1 \quad \text{for all } i = 1, \dots, m-1\,, \\
& \pi_m = 0\,.
\end{aligned}
\tag{3.7}
$$

DRP (3.7) is evry easy to solve:

Since $\pi_1 \le 1$ and we wish to maximize π_1, we try $\pi_1 = 1$. If there is no path from π_1 to π_m (node 1 to node m), using only arcs in J, then we can propagate the 1 from node 1 to all nodes reachable by a path from node 1 without violating the $\pi_i - \pi_j \le 0$ constraints, and an optimal solution to the DRP is then

$$
\bar{\pi} = \begin{cases}
1 & \texttt{for all nodes reachable by paths} \\
 & \texttt{from node 1 using arcs in } J \\
0 & \texttt{for all nodes from which node } m \\
 & \texttt{is reachable using arcs in } J \\
1 & \texttt{for all other nodes.}
\end{cases}
$$

(Notice that this $\bar{\pi}$ is not unique.)

We can then calculate

$$
\theta_1 = \min_{\substack{\text{arcs } (i,j) \notin J \\ \text{such that} \\ \bar{\pi}_i - \bar{\pi}_j > 0}} \{c_{ij} - (\pi_i - \pi_j)\}
$$

to update π and J, and re-solve the DRP.

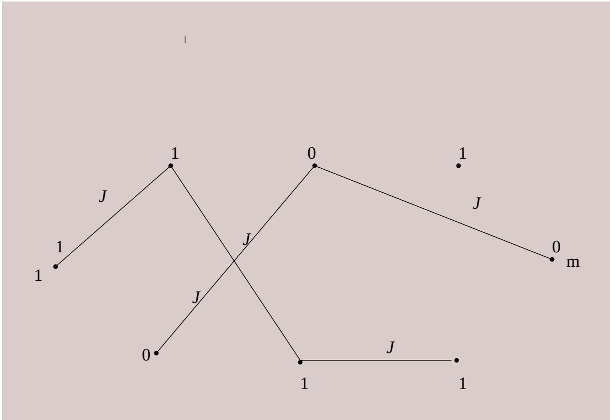


Figure 3.2: A solution to the restricted dual problem

$$\pi := \pi + \theta_1 \bar{\pi} \, .$$

If we get to a point where there is a path from node 1 to node m using arcs in J, $\pi_1 = 0$, and we find an optimal solution because $\xi^* = w^* = 0$. Any path from node 1 to node m using only arcs in J is optimal.

The primal-dual algorithm reduces the shortest path problem to repeated solution of the simpler problem of finding the set of nodes reachable from a given node.

Interpretation: Define at any point in the algorithm the set

$$\begin{aligned} W &= \{ i : \texttt{ node } m \texttt{ is reachable from } i \\ &\qquad \texttt{by admissible arcs} \} \\ &= \{ i : \bar{\pi}_i = 0 \} \, . \end{aligned}$$

Then the variable π_i remains fixed from the time that i enters W to the conclusion of the algorithm, because the corresponding $\bar{\pi}_i$ will always be zero.

Every arc that becomes admissible (enter J) stays admissable throughout the algorithm, because once we have

$$\pi_i - \pi_j = c_{ij} \quad \text{for } (i,j) \in E\,,$$

we always change π_i and π_j by the same amount.

- $\pi_i, i \in W$ is the length of the shortest path from node i to node m and the algorithm proceeds by adding to W, at each stage, the nodes not in W next closest to node m.
- At most $|v| = m$ stages.

Dijkstra's algorithm is an efficient implementation of the primal-dual algorithm for the shortest path problem.

3.3 Bellman's Equation

Let c_{ij} be the length of arc (i,j) (positive arcs if $c_{ij} > 0$; nonnegative if $c_{ij} \geq 0$).

Let u_{ij} be the length of the shortest path from $i \to j$. Define

$$u_i = u_{1i}.$$

Then *Bellman's Equations* are

$$\begin{cases} u_1 = 0, \\ u_i = \min\limits_{k \neq i}\{u_k + c_{ki}\} \end{cases}.$$

3.4 Dijkstra's Algorithm

In this section we assume that $c_{ij} \geq 0$. Denote

- P : permanently labeled nodes;
- T : temporarily labeled nodes.

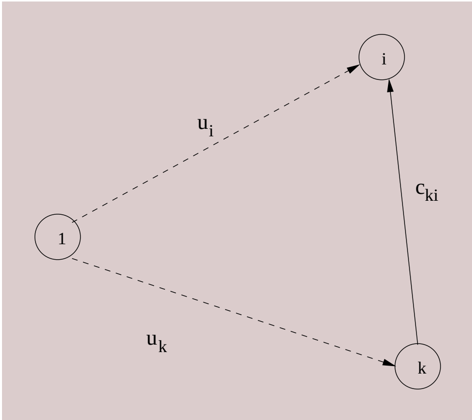


Figure 3.3: Bellman's equation

P and T always satisfy

$$P \cap T = \emptyset \quad \& \quad P \cup T = V.$$

Label for node j, $[u_j, l_j]$ where u_j : the length of the (may be temporary) shortest path from node 1 to j and l_j : the preceding node in the path.

Dijkstra's algorithm can be summarized as follows.

Step 0. $P = \{1\}$, $u_1 = 0$, $l_1 = 0$, $T = V \backslash P$. Compute

$$u_j = \begin{cases} c_{1j} & \text{if } (1, j) \in E, \\ \infty & \text{if } (1, j) \notin E, \end{cases}$$

$$l_j = \begin{cases} 1 & \text{if } (1, j) \in E, \\ 0 & \text{if } (1, j) \notin E. \end{cases}$$

Step 1. Find $k \in T$ such that

$$u_k = \min_{j \in T} \{u_j\}.$$

Let $P = P \cup \{k\}$ and $T = T \backslash \{k\}$. If $k = n$, stop.

Step 2. For $j \in T$, if $u_k + c_{kj} < u_j$, let $[u_j = u_k + c_{kj}, l_j = k]$ and go back to Step 1.

Claim: At any step, u_j is the length of the shortest path from 1 to j, only passing nodes in P.
[Suppose not and j is the first violation...].

Claim: The total cost is $O(n^2)$.

3.5 PERT or CPM Network

A large project is devisable into many unit "tasks". Each task requires a certain amount of time for its completion, and the tasks are partially ordered.

This network is sometimes called a PERT (Project Evaluation and Review Technique) or CPM (Critical Path Method) network. A PERT network is necessarily acyclic.

Theorem 3.1 *A digraph is acyclic if and only if its nodes can be renumbered in such a way that for all arc $(i,j),\ i < j$. [The work of this is $O(n^2)$]*

Claim: For any acyclic graph, at least one node has indegree 0. After renumbering it, we have for all $(i,j),\ i < j$.

Bellman's equations are

$$\begin{cases} u_1 = 0, \\ u_i = \min\limits_{k \neq i} \{u_k + c_{ki}\} \end{cases}$$

For **acyclic graphs**, they turn out to be

$$\begin{cases} u_1 = 0, \\ u_i = \min\limits_{k<i}\{u_k + c_{ki}\} \end{cases}$$

For a network with no cycles, one can replace each arc length by its negative value and still carry out the computation successfully.

$$\begin{cases} u_1 = 0, \\ u_i = \max\limits_{k<i}\{u_k + c_{ki}\} \end{cases}$$

Find the longest path = the time needs to finish the project.

3.6 Bellman-Ford Method

In this section we consider a general method of solution to Bellman's equations. Here we neither assume that the network is acyclic nor that all arc lengths are nonnegative. [We still assume that there are no negative cycles].

Step 1. $u_1^{(1)} = 0$, $u_j^{(1)} = c_{1j}$, $j \neq 1$.

Step k. For $k = 2, \dots, n$,

$$u_j^{(k)} = \min\{u_j^{(k-1)},\ \min_{i \neq j}\{u_i^{(k-1)} + c_{ij}\}\},\ j = 1, \cdots, n$$

Clearly, for each node j, successive approximations of u_j are monotone decreasing:

$$u_j^{(1)} \geq u_j^{(2)} \geq u_j^{(3)} \geq \dots$$

The total computational cost is $O(n^3)$.

Outline of Proof: $u_j^{(k)}$ is the length of the shortest path from node 1 to node j, subject to the condition that the path contains no more than k arcs.

3.7 Floyd-Warshall Method for Shortest Paths Between All Pairs

Again, we need the assumption that the networks contain no negative cycles in order that the Floyd-Warshall method works.

Step 0**.** $u_{ij}^{(1)} = c_{ij}$, $i, j = 1, \ldots, n$.

Step k**.** For $k = 1, \ldots, n$,

$$u_{ij}^{(k+1)} = \min\{u_{ij}^{(k)},\ u_{ik}^{(k)} + u_{kj}^{(k)}\},\ i, j = 1, \ldots, n$$

Claim: $u_{ij}^{(k)}$ is the length of a shortest path from i to j, subject to the condition that the path does no pass through $k, k+1, \ldots n$ (i and j excepted). [This means $u_{ij}^{(n+1)} = u_{ij}$].

Proof by induction. It is clearly true for Step 0. Suppose it is true for $u_{ij}^{(k)}$ for all i and j. Now consider $u_{ij}^{(k+1)}$. If a shortest path from node i to node j which does not pass through nodes $k+1, k+2, \ldots n$ does not pass through k, then $u_{ij}^{(k+1)} = u_{ij}^{(k)}$. Otherwise, if it does pass through node k, $u_{ij}^{(k+1)} = u_{ik}^{(k)} + u_{kj}^{(k)}$.

It is easy to see that the complexity of the Floyd-Warshall method is $O(n^3)$.

The Floyd-Warshall requires the storage of an $n \times n$ matrix. Initially this is $U^{(1)} = C$. Thereafter, $U^{(k+1)}$ is obtained from $U^{(k)}$ by using row k and column k to revise the remaining elements. That is, u_{ij} is compared with $u_{ik} + u_{kj}$ and if the later is smaller, $u_{ik} + u_{kj}$ is substituted for u_{ij} in the matrix.

There are other methods of the above type, e.g. G B Dantzig' method.

3.8 Other Cases

1. Sparse graphs

$$|A| << \frac{1}{2}|V|(|V| - 1).$$

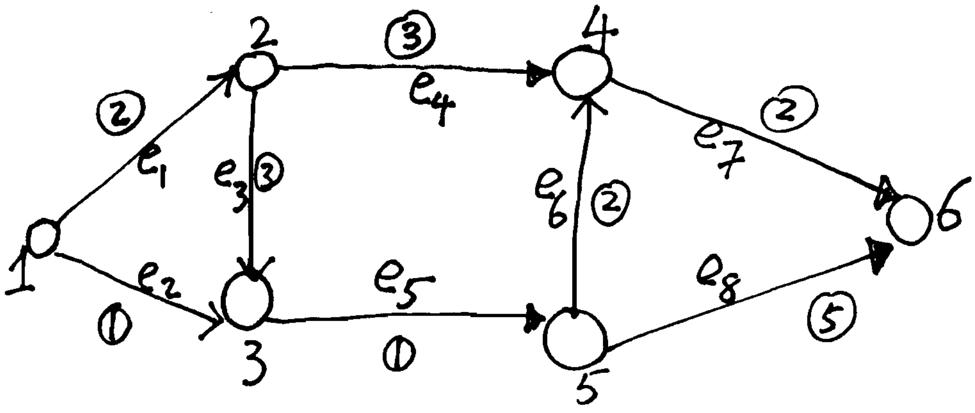
2. The kth shortest path problem.

• allow repetition

- not allow repetitive arcs
- not allow repetitive nodes

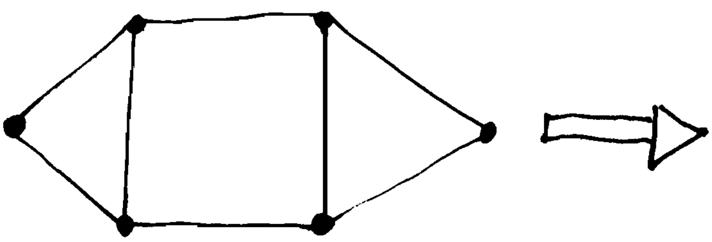
3. with time constraints
4. with fixed charge

CH3. Appendix A-1

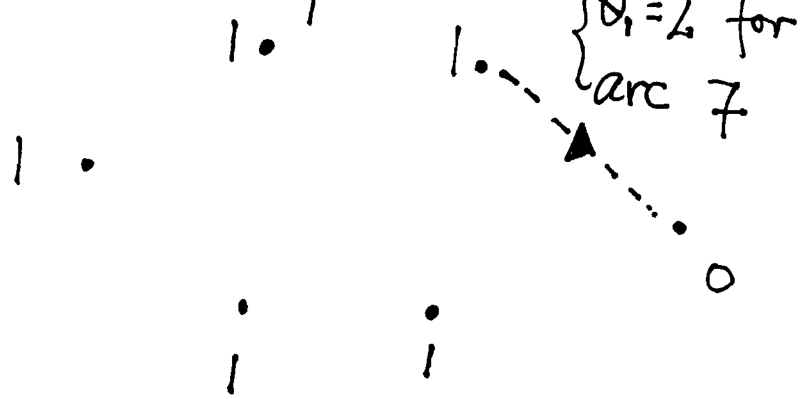


Shortest Path Example

Start with $\pi = (0, 0, 0, 0, 0)$



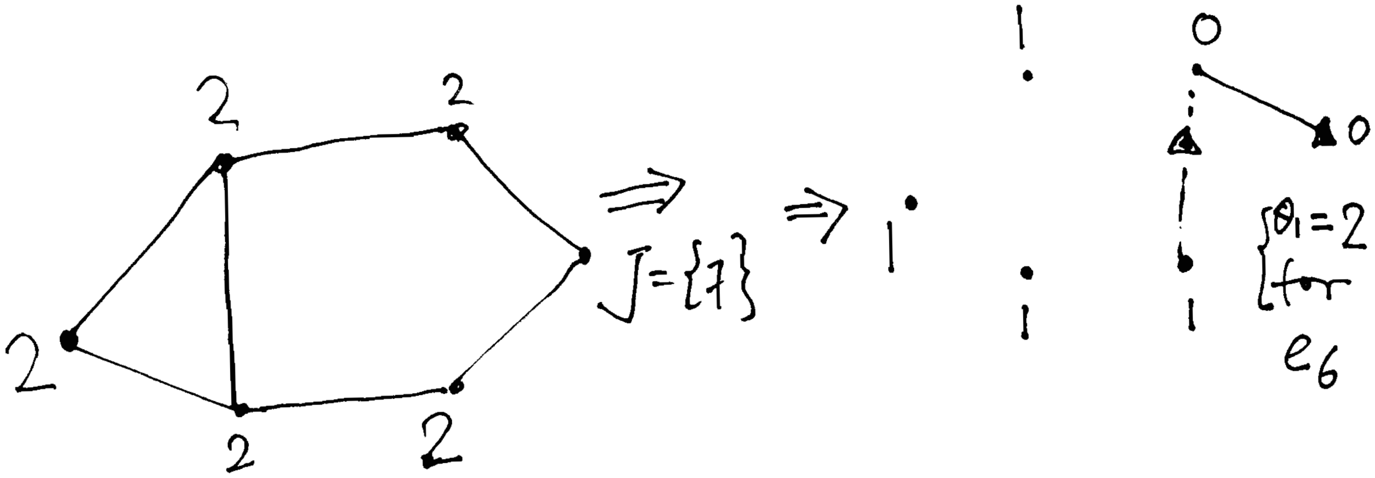
$\Rightarrow J = \phi$



D: $\pi = (0, 0, 0, 0, 0)$, DRP: $\bar{\pi} = (1, 1, 1, 1)$

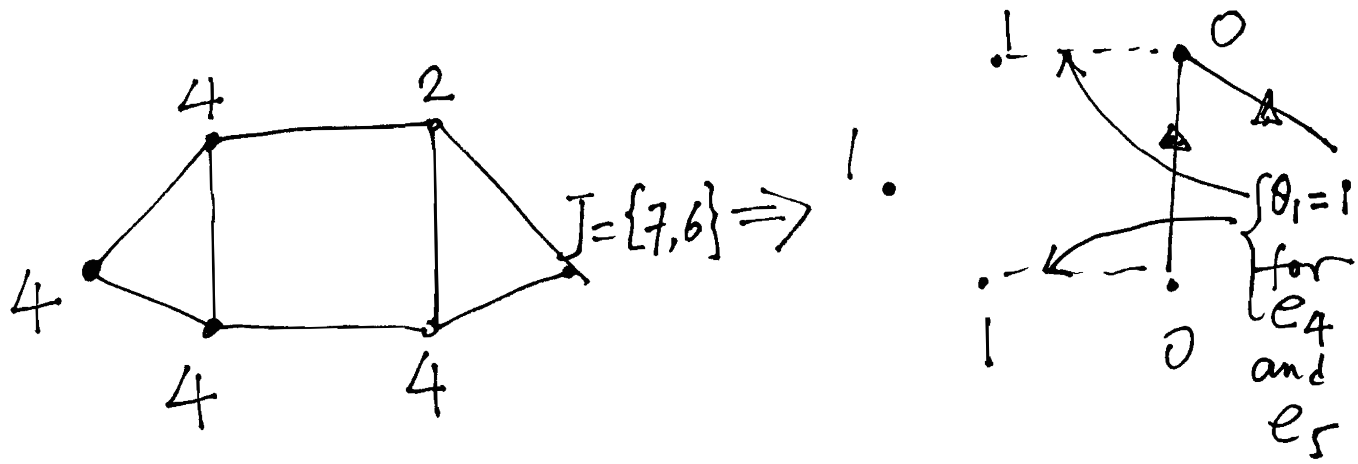
Iteration 2

CH3. A-2



D: $\Pi = (2, 2, 2, 2)$ DRP: $\bar{\pi} = (1,1,1,0,1)$

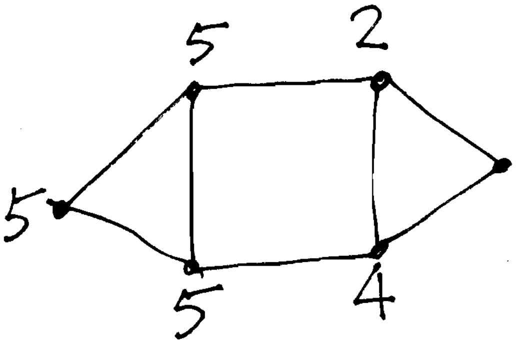
Iteration 2



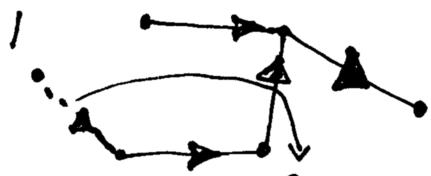
D: $\Pi = (4, 4, 4, 2, 4)$ DRP: $\bar{\pi} = (1,1,1,0,0)$

Iteration 3

A-3



$\Rightarrow J = \{7,6,5,4\} \Rightarrow$

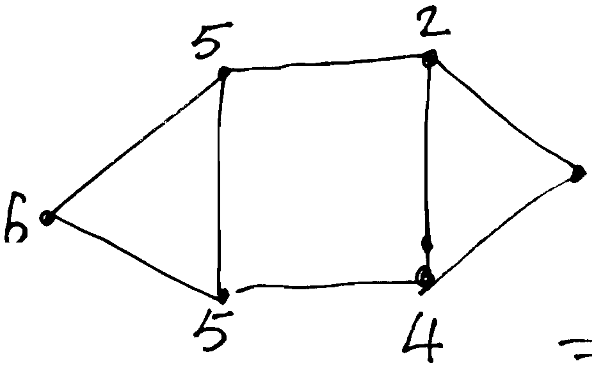


$\theta = 1$ for e_2

D: $\pi = (5,5,5,2,4)$

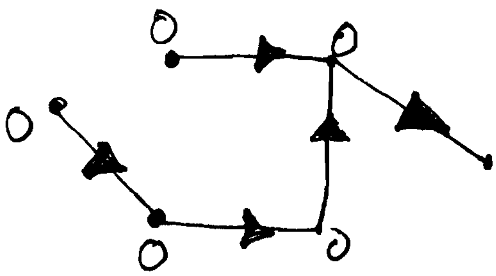
DRP: $\bar{\pi} = (1,0,0,0,0)$

Iteration 4



$\Rightarrow J = \{7,6,5,4,2)$

$\Rightarrow$



D: $\pi = (6,5,5,2,4)$

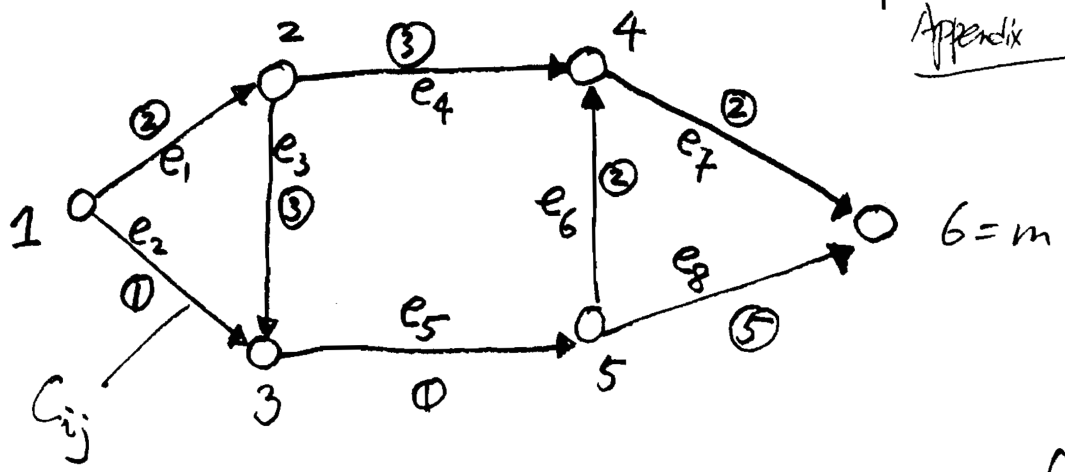
DRP: $\bar{\pi} = (0,0,0,0,0)$

$\pi_1^* = 6$

#

Consider the shortest path example ~~in Page 6~~

Appendix B-1



Use the Dijkstra's algorithm to find the shortest path from 1 to 6.

Step 1.

$P = \{1\}$, $T = \{2, 3, 4, 5, 6\}$

$u_1 := 0$, $l_1 = 0$

$u_2 := 2$, $l_2 = 1$

$u_3 := 1$, $l_3 = 1$

$u_4 := \infty$, $l_4 = 0$

$u_5 := \infty$, $l_5 = 0$

$u_6 := \infty$, $l_6 = 0$

step 2. $P = \{1, 3\}$, $T = \{2, 4, 5, 6\}$

$u_2 := \min\{u_2, u_3 + c_{32}\} = \min\{2, 1+\infty\} = 2$, $l_2 = 1$

$u_4 := \min\{u_4, u_3 + c_{34}\} = \min\{\infty, 1+\infty\} = \infty$, $l_4 = 0$

$u_5 = \min\{u_5, \ u_3 + c_{35}\} = \min\{\infty, \ 1+1\} = 2, \ l_5 = 3$ B-2

$u_6 = \min\{u_6, \ u_3 + c_{36}\} = \min\{\infty, 1+\infty\} = \infty, \ l_6 = 0$

step 2. $P = \{1, 3, 2\}, \ T = \{4, 5, 6\}$

$u_4 = \min\{u_4, \ u_2 + c_{24}\} = \min\{\infty, 5\} = 5, \ l_4 = 2$

$u_5 = \min\{u_5, \ u_2 + c_{25}\} = \min\{2, \ 2+\infty\} = 2, \ l_5 = 3$

$u_6 = \min\{u_6, \ u_2 + c_{26}\} = \min\{\infty, 2+\infty\} = \infty, \ l_6 = 0$

step3. $P = \{1, 3, 2, 5\} \quad T = \{4, 6\}$

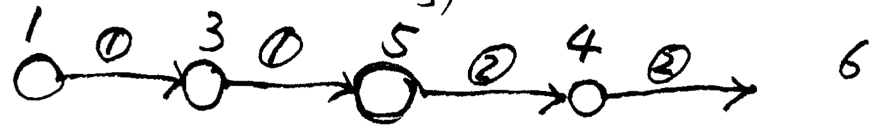
$u_4 = \min\{u_4, \ u_5 + c_{54}\} = \min\{5, 2+2\} = 4, \ l_4 = 5$

$u_6 = \min\{u_6, \ u_5 + c_{56}\} = \min\{\infty, 2+5\} = 7, \ l_6 = 5$

step4 $P = \{1, 3, 2, 5, 4\}, \ T = \{6\}$

$u_6 = \min\{u_6, \ u_4 + c_{46}\} = \min\{7, \ 4+2\} = 6, \ l_6 = 4$

step 5 $P = \{1, 3, 2, 5, 4, 6\}, T = \varphi$.



#