

ECE 158B, Data Networks II, Spring Quarter, 1999

Solutions of H.W. set 2

1. (a) Not possible. From \vec{D}_4 to \vec{D}_5 , the 2nd, 5th and 7th components increased (and all others did not change), hence $\vec{D}_4 \leq \vec{D}_5$. With this information, the monotonicity property of the Bellman-Ford iteration proved in class guarantees that $\vec{D}_5 \leq \vec{D}_6$ which is not true for this \vec{D}_6 .
 - (b) Possible. For a graph of 7 nodes, it is possible that Bellman-Ford algorithm converged at the 5th iteration, so that $\vec{D}_6 = \vec{D}_5$ which does not violate monotonicity.
 - (c) Possible. It is possible that the algorithm converged at the 6th iteration. The given \vec{D}_6 is possible because it is consistent with the monotonicity property exhibited from \vec{D}_4 to \vec{D}_5 , i.e., $\vec{D}_5 \leq \vec{D}_6$.
2. Define $D_i^k =$ the shortest distance (at the k^{th} iteration) from source node 1 to the destination node i .

k	P	D_1^k	D_2^k	D_3^k	D_4^k	D_5^k	D_6^k
1	{ 1 }	0	1	3	∞	∞	∞
2	{ 1, 2 }	0	1	2	7	3.5	∞
3	{ 1, 2, 3 }	0	1	2	7	3	∞
4	{ 1, 2, 3, 5 }	0	1	2	6	3	8
5	{ 1, 2, 3, 5, 4 }	0	1	2	6	3	7
6	{ 1, 2, 3, 5, 4, 6 }	0	1	2	6	3	7

^{*}The box indicates the node chosen to be added to the set P for the *next* iteration.

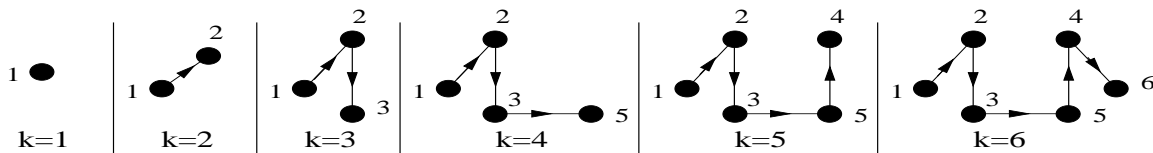


Figure 1: Illustration of the Dijkstra's algorithm. The *shortest path tree* is the last one on the right.

3. Define¹ $D_i^k =$ the shortest distance (at the k^{th} iteration) from node i to the destination node 1.

k	D_1^k	D_2^k	D_3^k	D_4^k	D_5^k	D_6^k
0	0	∞	∞	∞	∞	∞
1	0	1 (1) [*]	3 (1)	∞	∞	∞
2	0	1 (1)	2 (2)	7 (2)	3.5 (2)	∞
3	0	1 (1)	2 (2)	6.5 (5)	3 (3)	8 (4)
4	0	1 (1)	2 (2)	6 (5)	3 (3)	7.5 (4)
5	0	1 (1)	2 (2)	6 (5)	3 (3)	7 (4)
6	0	1 (1)	2 (2)	6 (5)	3 (3)	7 (4)

^{*}The number between parenthesis is the current "best-neighbor" node number.

¹Note the difference in the definitions of the D_i 's between Dijkstra's and Bellman-Ford's.

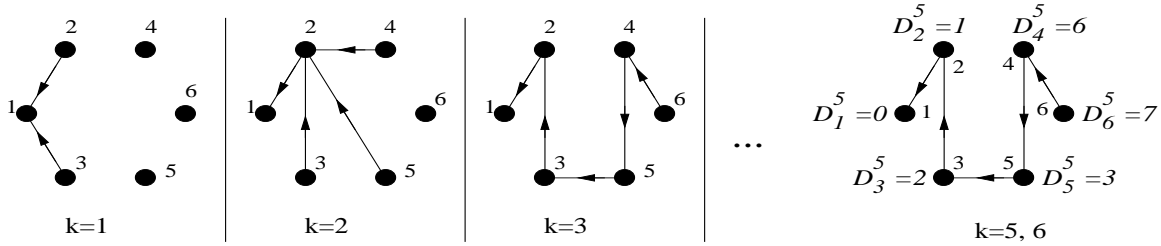


Figure 2: Illustration of the Bellman-Ford's algorithm. This shows the current “best-neighbor” and does not show the current D_i 's for all iterations. The *shortest path tree* is the last one on the right reached at iteration $k = 5$.

4. Define $D_i^k =$ the shortest distance (at the k^{th} iteration) from node i to the destination node 1. First, apply Bellman-Ford algorithm²:

k	D_1^k	D_2^k	D_3^k	D_4^k	D_5^k
0	0	∞	∞	∞	∞
1	0	2 (1)*	3 (1)	∞	∞
2	0	2 (1)	3 (1)	-3 (2)	4 (3)
3	0	2 (1)	0 (4)	-3 (2)	4 (3)
4	0	2 (1)	0 (4)	-3 (2)	1 (3)
5	0	2 (1)	0 (4)	-3 (2)	1 (3)

*The number between parenthesis is the current “best-neighbor” node number.

Hence, $\vec{x}_1 = (x_1, x_2, x_3, x_4, x_5) = (0, 2, 0, -3, 1)$ is the first solution to Bellman's equation. To produce another *distinct* solution, \vec{x}_2 , such that $\vec{x}_2 = \vec{f}(\vec{x}_2)$, we assume that $\vec{x}_2 = (x_1, x_2, x_3, x_4, x_5)$, where $x_1 = 0$. Since \vec{x}_2 is a solution to Bellman's equation, then the components of \vec{x}_2 satisfy Bellman's equation. Therefore (based on the given graph):

$$\begin{aligned}
 x_1 &= 0 \\
 x_2 &= \min\{2, 2 + x_3\} \\
 x_3 &= \min\{3, 3 + x_4\} \\
 x_4 &= \min\{-5 + x_2, 1 + x_5\} \\
 x_5 &= 1 + x_3
 \end{aligned}$$

Let $x_3 = \alpha$ and note that this implies $x_2 = \min\{2, 2 + \alpha\} = 2 + \alpha$ if we assume $2 + \alpha \leq 2$, i.e., if we assume $\alpha \leq 0$. Therefore, for any $\alpha \leq 0$, there exists a solution given by:

$$\begin{aligned}
 x_1 &= 0 \\
 x_2 &= 2 + \alpha \\
 x_3 &= \alpha \\
 x_5 &= 1 + \alpha, \quad \text{hence,} \\
 x_4 &= \min\{-5 + (2 + \alpha), 1 + (1 + \alpha)\} = \alpha - 3
 \end{aligned}$$

For example, if we choose $\alpha = -1$: $\vec{x}_2 = (0, 1, -1, -4, 0)$ is a solution. Note that there are infinite number of solutions (one for every $\alpha \leq 0$) but they have no physical meaning.

²Note that because of the existence of *non-positive* length cycle, the algorithm is not guaranteed to converge for *arbitrary* initial condition. However, if we start with the initial condition $\vec{D}^0 = (0, \infty, \infty, \dots, \infty)$ and if all cycles have *nonnegative* length (as in this example), the algorithm is guaranteed to converge and yield a solution. As shown in class, a solution is guaranteed to be *unique* only if all cycles have *positive* length.