Sample Solution

Math 323 Operations Research HW9

8.2-2:

$[0^* \quad 2^* \quad 8 \quad \infty \quad \infty]$
$[0^* \quad 2^* \quad 7 \quad 6 \quad 14]$
$[0^* \quad 2^* \quad 7 \quad 6^* \quad 14]$
$[0^* \quad 2^* \quad 7^* \quad 6^* \quad 14]$
$[0^* \quad 2^* \quad 7^* \quad 6^* \quad 14^*]$

The shortest path is 1 to 2 to 5, yielding 14 unit length.

8.2-3:

	N2	N3	N4	N5	Supply
N1	2	8	M	M	1
N2	0	5	4	12	1
N3	M	0	6	M	1
N4	M	M	0	10	1
Demand	1	1	1	1	

Using Lowest Cost method, we got the optimal solution:

	N2	N3	N4	N5	Supply
N1	2	8	M	M; 1	1
N2	0; 1	5	4	12	1
N3	M	0; 1	6	M	1
N4	M	M	0; 1	10	1
Demand	1	1	1	1	

Iterating to get the optimal solution:

	N2	N3	N4	N5	Supply
N1	2 1	8 0	M	M	1
N2	0	5 0	4 0	12 1	1
N3	M	0 1	6	M	1
N4	M	M	0 1	10	1
Demand	1	1	1	1	

8.2-4:

$[0^* \quad 2 \quad 1^* \quad \infty]$

$[0^* \quad 2^* \quad 1^* \quad 2^*]$

The shortest path found by the Dijkstra's Algorithm is 1 to 3 to 4, yielding 2 unit length. However, we can see that the shortest pathway is actually 1-2-3-4, yielding 1 unit length. Dijkstra's Algorithm fails because there is a negatively weighted path from 2 to 3, which violates Dijkstra's assumption of all the path are of non-negative length or weight.

Dijkstra has such assumption, because it is a greedy algorithm, which means that to ensure the final outcome is the minimum, Dijkstra ensures every previous step is the minimum. Since the final weight/length is the sum of the previous one, the ideal result can only be realized with non-negative assumption. Once there exists a path that can reduce the total length, the assumption is violated and there might be a negative loop, meaning every time we bypass the negative path we can reduce the total length, which might result in a forever negative loop, and there might not exist a minimum solution.

Bellman-Ford Algorithm can be used to detect any negative loop. There could be a fix to Dijkstra when facing negative path, without a negative loop. To fix Dijkstra, we have to allow a node to be visited multiple times to modify its value. In other words, we have to allow processed node to be "un-do" to become unprocessed nodes, to update it with the negative effect, and update its neighbouring nodes correspondingly. On Wikipedia, there is a set of pseudo code that simulates the process. However, this process could take up much more time, since it allows each node to be visited multiple time, it loses the efficiency of ensuring minimality due to the non-negative condition.

8.2-6:

Let i represent purchase year, and j represent the year the cell phone is used.

$c_{01} = 40 + 20 = 60$,

$c_{02} = 40 + 20 + 30 = 90$,

$c_{03} = 40 + 20 + 30 + 40 = 130$,

$c_{04} = 40 + 20 + 30 + 40 + 60 = 190$,

$c_{05} = 40 + 20 + 30 + 40 + 60 + 70 = 260$,

$c_{06} = \infty$,

$c_{12} = 40 + 20 = 60$,

$c_{13} = 40 + 20 + 30 = 90$,

$c_{14} = 40 + 20 + 30 + 40 = 130$,

$c_{15} = 40 + 20 + 30 + 40 + 60 = 190$,

$c_{16} = 40 + 20 + 30 + 40 + 60 + 70 = 260$,

$c_{23} = 40 + 20 = 60,$
$c_{24} = 40 + 20 + 30 = 90,$
$c_{25} = 40 + 20 + 30 + 40 = 130,$
$c_{26} = 40 + 20 + 30 + 40 + 60 = 190,$

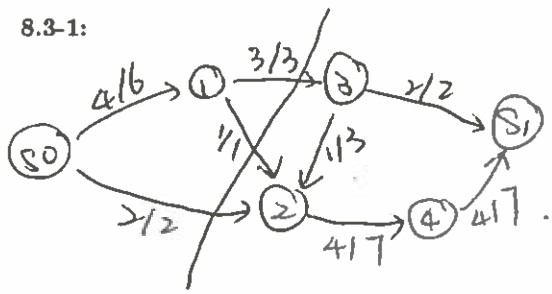
$c_{34} = 40 + 20 = 60,$
$c_{35} = 40 + 20 + 30 = 90,$
$c_{36} = 40 + 20 + 30 + 40 = 130,$

$c_{45} = 40 + 20 = 60,$
$c_{46} = 40 + 20 + 30 = 90,$

$c_{56} = 40 + 20 = 60,$

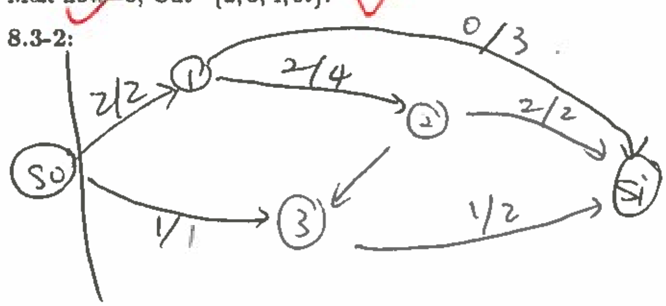
Using Dijkstra's Algorithm:
$[0^* \quad 60^* \quad 90 \quad 130 \quad 190 \quad 260 \quad \infty]$
$[0^* \quad 60^* \quad 90^* \quad 130 \quad 190 \quad 250 \quad 320]$
$[0^* \quad 60^* \quad 90^* \quad 130^* \quad 180 \quad 220 \quad 280]$
$[0^* \quad 60^* \quad 90^* \quad 130^* \quad 180^* \quad 220 \quad 260]$
$[0^* \quad 60^* \quad 90^* \quad 130^* \quad 180^* \quad 220^* \quad 260^*]$
The shortest path is year0 to year3 to year6, yielding 260 unit cost, which means that we have to purchase a new phone at the end of year 3 and use it until year 6.

8.3-1:



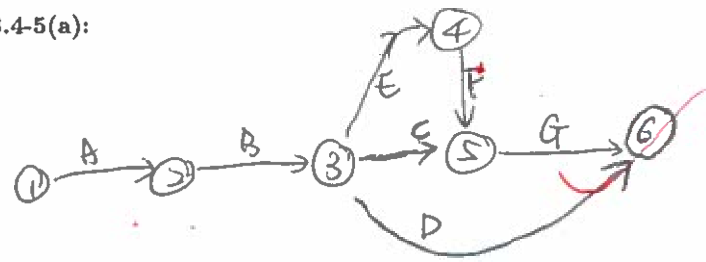
Max Z=x_0, subject to:
$x_{so,1} \leq 6,\ x_{so,2} \leq 2,\ x_{1,2} \leq 1,\ x_{1,3} \leq 3,\ x_{2,4} \leq 7,\ x_{4,si} \leq 7,\ x_{3,2} \leq 3,\ x_{3,si} \leq 2.$
$x_0 = x_{so,1} + x_{so,2},\ x_{so,1} = x_{1,3} + x_{1,2},\ x_{1,3} = x_{3,2} + x_{3,si},\ x_{2,4} = x_{4,si},\ x_0 = x_{3,si} + x_{4,si}.$
Max flow=6, Cut=$\{2, 3, 4, si\}$.

8.3-2:



Max Z=x_0, subject to:
$x_{so,1} \leq 2,\ x_{so,3} \leq 1,\ x_{1,2} \leq 4,\ x_{1,si} \leq 3,\ x_{2,si} \leq 2,\ x_{2,3} \leq 1,\ x_{3,si} \leq 2.$
$x_0 = x_{so,1} + x_{so,3},\ x_{so,1} = x_{1,si} + x_{1,2},\ x_{3,si} = x_{2,3} + x_{so,3},\ x_{1,2} = x_{2,si} + x_{2,3},\ x_0 = x_{3,si} + x_{2,si} + x_{1,si}.$
Max flow=3, Cut=$\{1, 2, 3, si\}$.

8.4-5(a):



Total Duration of the project is 5+8+10+3=26 days. ABEFG and ABCG are two critical paths.

	TF	FF
A	0	0
B	0	0
C	0	0
D	8	8
E	0	0
F	0	0
G	0	0

8.4-5(b):

minimize Z=30A+ 15B+ 20C+ 40D+ 20E+ 30F+ 40G, subject to:

$x_1 - x_6 \leq 20$, $x_2 \geq x_1 + 5 - A$, $x_3 \geq x_2 + 8 - B$, $x_4 \geq x_3 + 4 - E$, $x_5 \geq x_4 + 6 - F$, $x_5 \geq x_3 + 10 - c$, $x_6 \geq x_3 + 5 - D$, $x_6 \geq x_5 + 3 - G$.

$A \leq 2$, $B \leq 3$, $C \leq 1$, $D \leq 2$, $E \leq 2$, $F \leq 3$, $G \leq 1$. A, B, C, D, E, F, G ≥ 0, and x_1, x_2, x_3, x_4, x_5, x_6 urs. A–G are the days of reduction of corresponding process.

Solving by Matlab, we got the optimal solution:

$Z = 145$ $x_1 = 0$, $x_2 = 3$, $x_3 = 8$, $x_4 = 11$, $x_5 = 17$, $x_6 = 20$, $A = 2$, $B = 3$, $C = 1$, $D = 0$, $E = 1$, $F = 0$, $G = 0$.