Chapter 2. Iteration Bound

Exercise Solution

Problem 1 [Solution]:

```
L1=
    -1     0    -1
     7    -1     3
     3    -1    -1

L2=
     7    -1     3
     6     7    -1
    -1     3    -1

L3=
     6     7    -1
    14     6    10
    10    -1     6

ans =

    3.5000
```

Problem 2 [Solution]:

```
L1=
     4     0    -1
     5    -1     0
     5    -1    -1

L2=
     8     4     0
     9     5    -1
     9     5    -1

L3=
    12     8     4
    13     9     5
    13     9     5


ans =

     4
```

Problem 3 [Solution]:
i)

L1=

-1	0	-1	-1	-1	-1	-1	-1
-1	-1	0	-1	-1	-1	-1	-1
-1	-1	-1	0	-1	-1	-1	-1
3	-1	-1	-1	6	-1	-1	-1
-1	-1	-1	-1	-1	0	-1	-1
-1	-1	-1	-1	-1	-1	0	-1
-1	-1	-1	-1	-1	-1	-1	0
4	-1	-1	-1	7	-1	-1	-1

L2=

-1	-1	0	-1	-1	-1	-1	-1
-1	-1	-1	0	-1	-1	-1	-1
3	-1	-1	-1	6	-1	-1	-1
-1	3	-1	-1	-1	6	-1	-1
-1	-1	-1	-1	-1	-1	0	-1
-1	-1	-1	-1	-1	-1	-1	0
4	-1	-1	-1	7	-1	-1	-1
-1	4	-1	-1	-1	7	-1	-1

L3=

-1	-1	-1	0	-1	-1	-1	-1
3	-1	-1	-1	6	-1	-1	-1
-1	3	-1	-1	-1	6	-1	-1
-1	-1	3	-1	-1	-1	6	-1
-1	-1	-1	-1	-1	-1	-1	0
4	-1	-1	-1	7	-1	-1	-1
-1	4	-1	-1	-1	7	-1	-1
-1	-1	4	-1	-1	-1	7	-1

L4=

3	-1	-1	-1	6	-1	-1	-1
-1	3	-1	-1	-1	6	-1	-1
-1	-1	3	-1	-1	-1	6	-1
-1	-1	-1	3	-1	-1	-1	6
4	-1	-1	-1	7	-1	-1	-1
-1	4	-1	-1	-1	7	-1	-1
-1	-1	4	-1	-1	-1	7	-1
-1	-1	-1	4	-1	-1	-1	7

```
L5=
    -1     3    -1    -1    -1     6    -1    -1
    -1    -1     3    -1    -1    -1     6    -1
    -1    -1    -1     3    -1    -1    -1     6
    10    -1    -1    -1    13    -1    -1    -1
    -1     4    -1    -1    -1     7    -1    -1
    -1    -1     4    -1    -1    -1     7    -1
    -1    -1    -1     4    -1    -1    -1     7
    11    -1    -1    -1    14    -1    -1    -1
L6=
    -1    -1     3    -1    -1    -1     6    -1
    -1    -1    -1     3    -1    -1    -1     6
    10    -1    -1    -1    13    -1    -1    -1
    -1    10    -1    -1    -1    13    -1    -1
    -1    -1     4    -1    -1    -1     7    -1
    -1    -1    -1     4    -1    -1    -1     7
    11    -1    -1    -1    14    -1    -1    -1
    -1    11    -1    -1    -1    14    -1    -1

L7=
    -1    -1    -1     3    -1    -1    -1     6
    10    -1    -1    -1    13    -1    -1    -1
    -1    10    -1    -1    -1    13    -1    -1
    -1    -1    10    -1    -1    -1    13    -1
    -1    -1    -1     4    -1    -1    -1     7
    11    -1    -1    -1    14    -1    -1    -1
    -1    11    -1    -1    -1    14    -1    -1
    -1    -1    11    -1    -1    -1    14    -1

L8=
    10    -1    -1    -1    13    -1    -1    -1
    -1    10    -1    -1    -1    13    -1    -1
    -1    -1    10    -1    -1    -1    13    -1
    -1    -1    -1    10    -1    -1    -1    13
    11    -1    -1    -1    14    -1    -1    -1
    -1    11    -1    -1    -1    14    -1    -1
    -1    -1    11    -1    -1    -1    14    -1
    -1    -1    -1    11    -1    -1    -1    14


ans =

    1.7500
```

ii) From Fig 2.13, it can be observed that the number of delays in any loop is a multiple of 4. Hence, the iteration bound can be derived from the entries in the diagonals of $L^{(4)}$ and $L^{(8)}$. We can compute $L^{(2)}$ from $L^{(1)}$, $L^{(4)}$ from $L^{(2)}$, and then $L^{(8)}$ from $L^{(4)}$ using a formula very similar to that used in the LPM algorithm. $L^{(m)}$ can be computed from $L^{(m/2)}$ by

$$l_{i,j}^{(m)} = \max_{k \in K}(-1, l_{i,k}^{(m/2)} + l_{k,j}^{(m/2)})$$

Using the above formula, the derived $L^{(2)}$, $L^{(4)}$ and $L^{(8)}$ are the same as those computed in i).

Alternatively, the 4 delays can be considered as a mega delay, and $L^{(1)}$ can be constructed as a 2×2 matrix. Applying the LPM algorithm, you can find the longest paths between the mega delays and hence the loops with the longest computation time. Since there are only 2 mega delays, you only need to compute $L^{(2)}$. Then divide the entries in the diagonal of $L^{(1)}$ by 4 and those of $L^{(2)}$ by 8 (each mega delay represents 4 delays) also gives the correct iteration bound.