ELE617E
Lectures

Prof. Dr. Müştak E. Yalçın

Istanbul Technical University

mustak.yalcin@itu.edu.tr

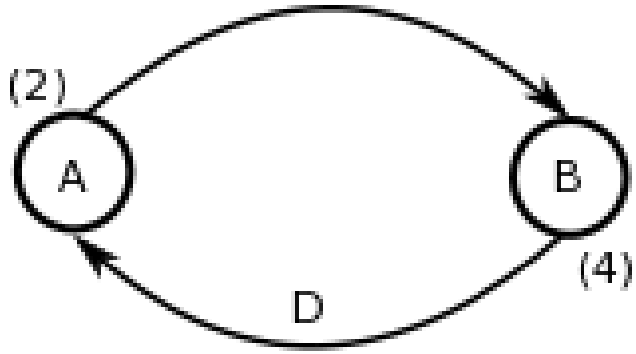
Iteration Bound

The iteration bound of the DSP program is the lower bound on the iteration of the DSP program regardless of the amount of computing resources available.

It is not possible to achieve an iteration less than the iteration bound even when **infinite processors** are available.
When the DFG is recursive, the iteration bound is the fundamental limit on the minimum sample period of a hardware implementation of the DSP program.

Iteration Bound

```
for (n=0;i<N;i++) {
y(n)=a y(n-1)+ x(n);
}
```

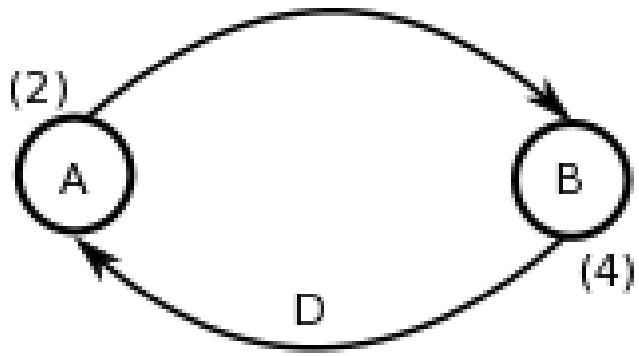


Node A represents addition and Node B represents multiplication. And edges represent data path, note that edge $B \rightarrow A$ has one delay.

$$\underbrace{A}_{ay(n-1)+x(n)} \xrightarrow{y(n)} \underbrace{B}_{a*y(n)} \xrightarrow{ay(n)} \underbrace{A}_{ay(n)+x(n+1)}$$

nth iteration of B must be executed before the $n+1$ *th* iterarion of A!
Periodic schedule: ABABA...
An iteration of a node is the execution of the node exactly once. An iteration of the DFG is the execution of each node in the DFG exactly once!

Iteration Bound



intra-iteration precedence constraint : $A_n \rightarrow B_n$
inter-iteration precedence constraint : $B_n \Rightarrow A_{n+1}$

$$\text{iteration} : A_n \rightarrow B_n \Rightarrow A_{n+1} \rightarrow B_{n+1} \Rightarrow A_{n+2}$$

Loop

A loop is a directed path that begin and ends at the same node ($A \rightarrow B \rightarrow A$).

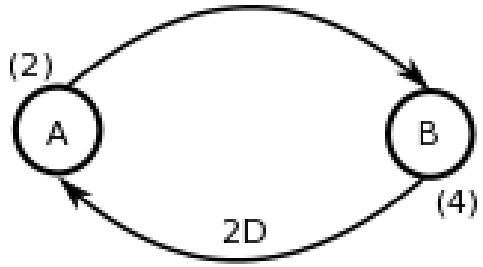
Iteration Bound

The loop bound

The loop bound of l-th loop is defined as

$$\frac{t_l}{w_l}$$

where t_l is the loop computation time and w_l is the number of delay in the loop.

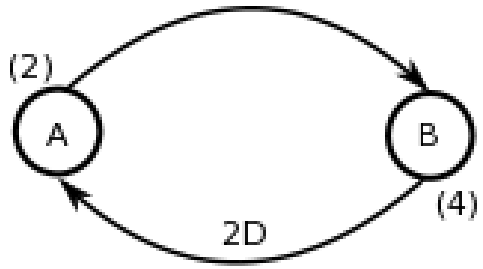


Loop bound $= \frac{6}{2} = 3$

How one iteration of this loop can be executed in 3 u.t.

Iteration Bound

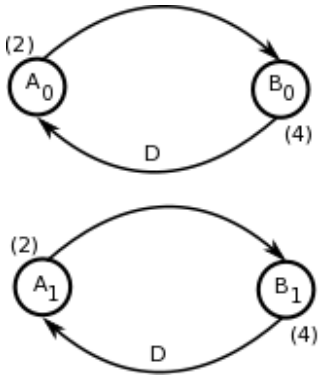
$$y(n) = ay(n-2) + x(n)$$



$$y(2k) = ay(2k-2) + x(2k) = ay(2(k-1)) + x(2k)$$
$$y(2k+1) = ay(2k-1) + x(2k+1) = ay(2(k-1)+1) + x(2k+1)$$

Iteration Bound

$$y(2k) = ay(2k-2) + x(2k) = ay(2(k-1)) + x(2k)$$
$$y(2k+1) = ay(2k-1) + x(2k+1) = ay(2(k-1)+1) + x(2k+1)$$



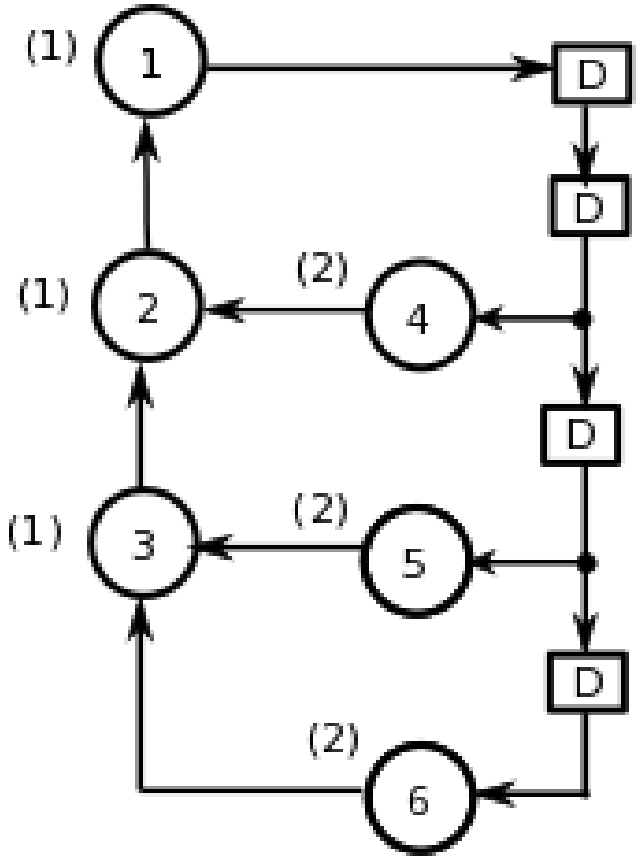
$$y(2*1) = ay(0) + x(2) = ay(2(1-1)) + x(2)$$
$$y(2*1+1) = ay(1) + x(3) = ay(2(1-1)+1) + x(3)$$
$$y(2*2) = ay(2) + x(4) = ay(2(2-1)) + x(4)$$

$A_0 \rightarrow B_0 \Rightarrow A_2 \rightarrow B_2 \Rightarrow A_4$

$A_1 \rightarrow B_1 \Rightarrow A_3 \rightarrow B_3 \Rightarrow A_5$

$$T_s = \frac{T_{clk}}{2} = \frac{6}{2}$$

Iteration Bound



$1 \rightarrow 4 \rightarrow 2 \rightarrow 1$, Loop bound $= \frac{4}{2}$u.t.

$1 \rightarrow 5 \rightarrow 3 \rightarrow 2 \rightarrow 1$, Loop bound $= \frac{5}{3}$u.t.

$1 \rightarrow 6 \rightarrow 3 \rightarrow 2 \rightarrow 1$, Loop bound $= \frac{5}{4}$u.t.

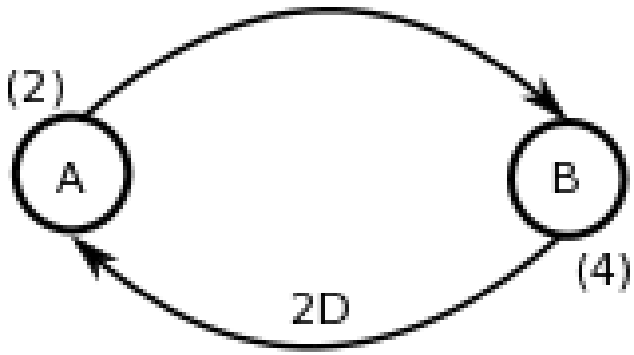
Iteration Bound

Critical Loop: the loops in which has maximum loop bound.

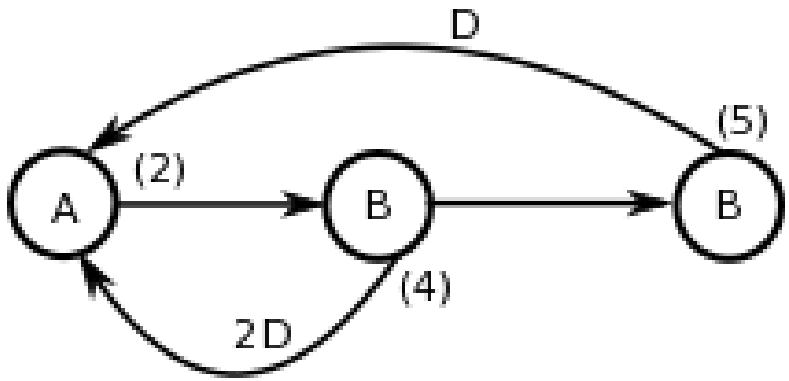
Iteration Bound

The maximum loop bound:

$$T_\infty = \max\{\frac{t_l}{w_l}\}$$



Iteration bound = 3 u.t.



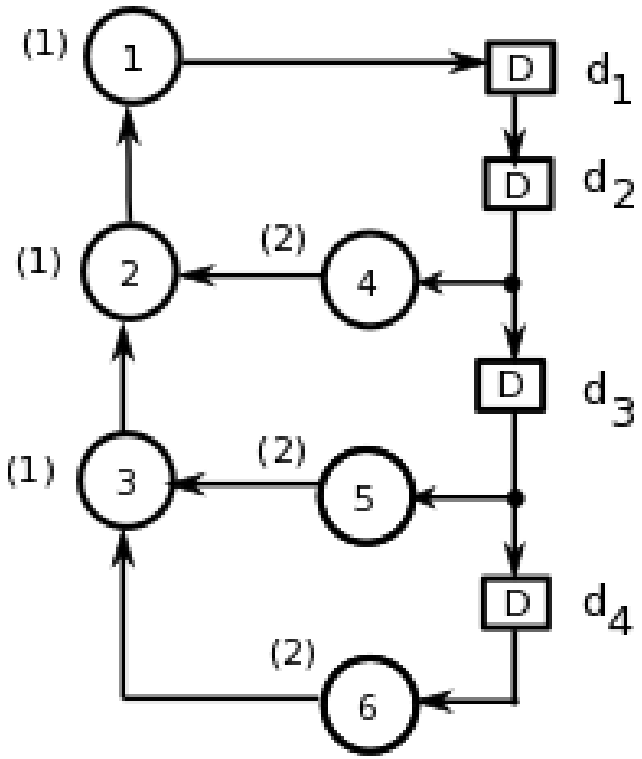
Iteration bound = 11 u.t.

Longest Path Matrix Algorithm

A series of matrix ($L^{(m)}$ where $m = 1, 2, ..., d$ and d is number of delays) is constructed, and the iteration bound is found by examining the diagonal elements of the matrices.

- $l_{i,j}^{(m)}$: the longest computation time of all paths from delay element d_i to delay element d_j that pass through exactly $m-1$ delays, where the delay d_i and delay d_j are not included for $m-1$ delays.
- If no path exists, then the value of l equals -1.

Longest Path Matrix Algorithm



$$L^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix}$$

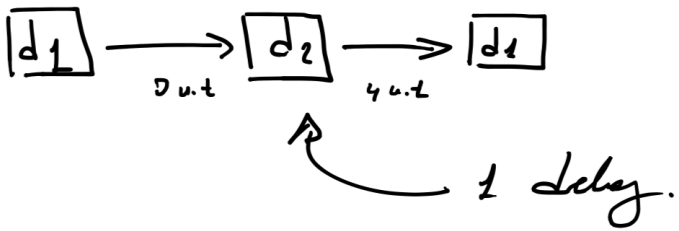
Longest Path Matrix Algorithm

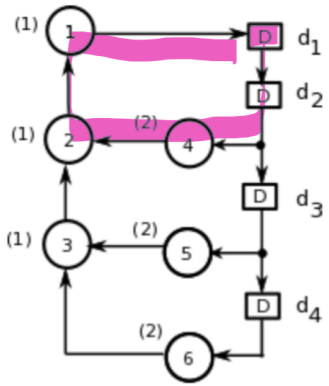
$$L^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix} \qquad l_{i,j}^{(m+1)} = \max\left(-1, l_{i,k}^{(1)} + l_{k,j}^{(m)}\right)$$

$l_{1,1}^{2} = \max(-1, l_{1,k}^{1} + l_{k,1}^{1})$

$L^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix}$ $L^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix}$

$l_{11}^{2} = \max\{-1, 0+4\}$

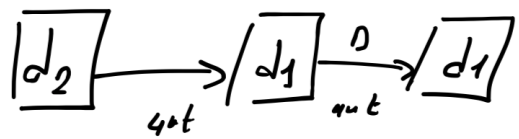


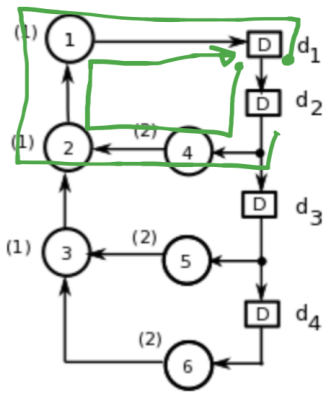


$l_{21}^{3} = \max\{-1, l_{2k}^{1} + l_{k1}^{2}\}$

$L^{(1)} = \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix}$ $L^{(2)} = \begin{bmatrix} 4 & -1 & 0 & -1 \\ 5 & 4 & -1 & 0 \\ 5 & 5 & -1 & -1 \\ -1 & 5 & -1 & -1 \end{bmatrix}$

$l_{21}^{3} = \max\{-1, 8, 5\} = 8$





$d_2 \to d_1 \to d_2 \to d_1$

2D

Longest Path Matrix Algorithm

The higher order matrices can be recursively computed

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

where K is the set of integers k in the interval $[1, d]$ such that neither $l_{i,k}^{(1)} = -1$ nor $l_{k,j}^{(1)} = -1$ holds.

$$l_{2,3}^{(2)} = \max_{k \in \emptyset} \left(-1, l_{2,k}^{(1)} + l_{k,3}^{(1)}\right)$$

$$l_{3,2}^{(2)} = \max_{k=1} \left(-1, l_{3,k}^{(1)} + l_{k,2}^{(1)}\right)$$

$$\begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & 0 & -1 & -1 \\ 4 & -1 & 0 & -1 \\ 5 & -1 & -1 & 0 \\ 5 & -1 & -1 & -1 \end{bmatrix} \longrightarrow \begin{bmatrix} 4 & -1 & 0 & -1 \\ 5 & 4 & -1 & 0 \\ 5 & 5 & -1 & -1 \\ -1 & 5 & -1 & -1 \end{bmatrix}$$

Longest Path Matrix Algorithm

The Iteration Bound can be calculated from the $L^{(m)}$ matrices by

$$T_\infty = \max\{\frac{l_{i,i}^{(m)}}{m}\}, \quad i, m \in \{1, 2, ...d\}$$

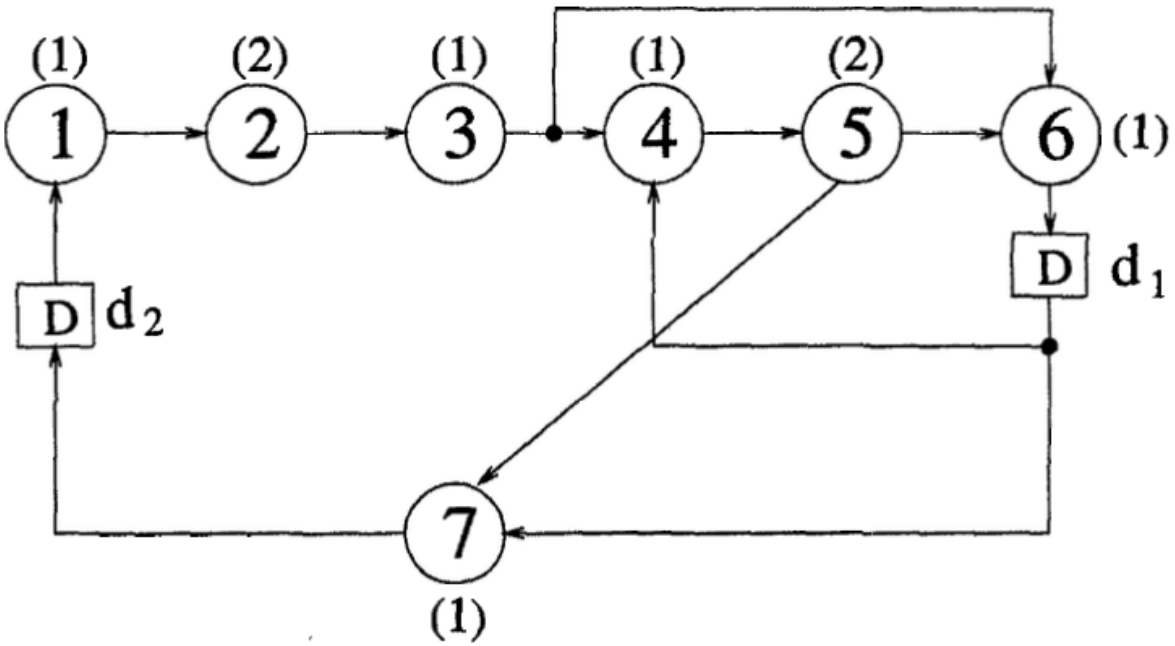
since the diagonal element represents the longest computation time of all loops with m delays which contains d_i.

$$L^{(2)} = \begin{bmatrix} 4 & -1 & 0 & -1 \\ 5 & 4 & -1 & 0 \\ 5 & 5 & -1 & -1 \\ -1 & 5 & -1 & -1 \end{bmatrix} L^{(3)} = \begin{bmatrix} 5 & 4 & -1 & 0 \\ 8 & 5 & 4 & -1 \\ 9 & 5 & 5 & -1 \\ 9 & -1 & 5 & -1 \end{bmatrix} L^{(4)} = \begin{bmatrix} 8 & 5 & 4 & -1 \\ 9 & 8 & 5 & 4 \\ 10 & 9 & 5 & 5 \\ 10 & 9 & -1 & 5 \end{bmatrix}$$

$$T_\infty = \max\{\frac{4}{2}, \frac{4}{2}, \frac{5}{3}, \frac{5}{3}, \frac{5}{3}, \frac{8}{4}, \frac{8}{4}, \frac{5}{4}, \frac{5}{4}\} = 2$$

▸ Longest Path Matrix Algorithm

Longest Path Matrix Algorithm



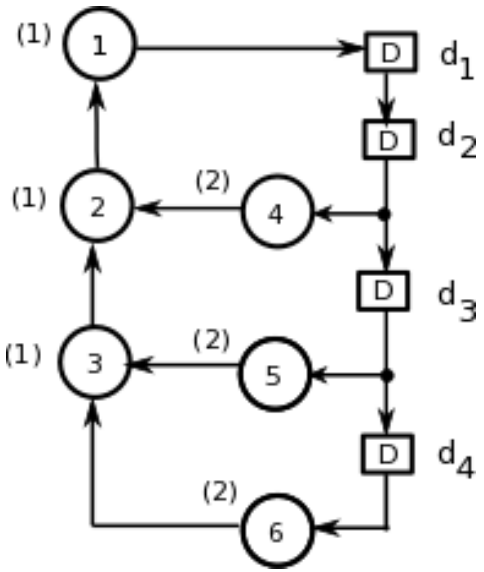
$$L^{(1)} = \begin{bmatrix} 4 & 4 \\ 8 & 8 \end{bmatrix}, L^{(2)} = \begin{bmatrix} 12 & 12 \\ 16 & 16 \end{bmatrix}$$

$$T_\infty = \max\{\frac{4}{1}, \frac{8}{1}, \frac{12}{2}, \frac{16}{2}\} = 8$$

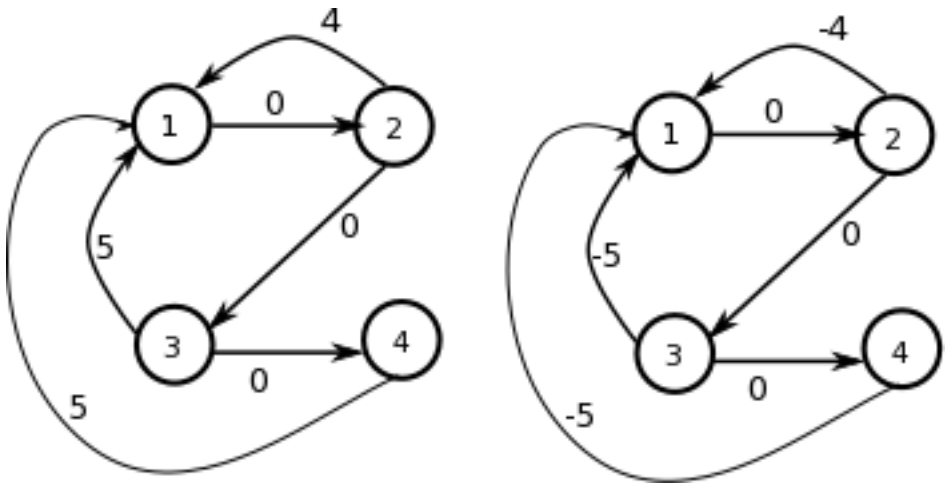
Minimum Cycle Mean Method (MCM)

$$G \quad \rightarrow \quad G_d \text{ and } \bar{G}_d$$

G_d which are used to compute the iteration bound, can be found from the DFG.



$\rightarrow$

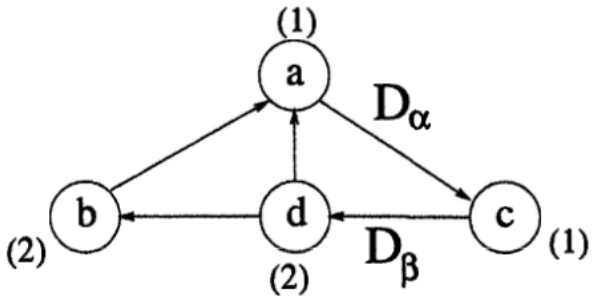


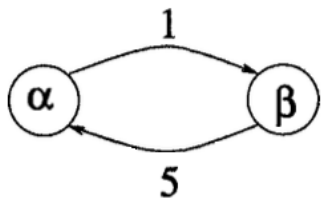
G_d has d nodes where each nodes corresponds to one of the delay in G, $w_{i,j}$ is the longest path length among all the path in G (do not pass through any delay!)

Note: $L^{(1)}$ and G_d.

Minimum Cycle Mean Method (MCM)

The sum of the edge weight in a cycle c in G_d is the max. comp. time of all cycles in G.



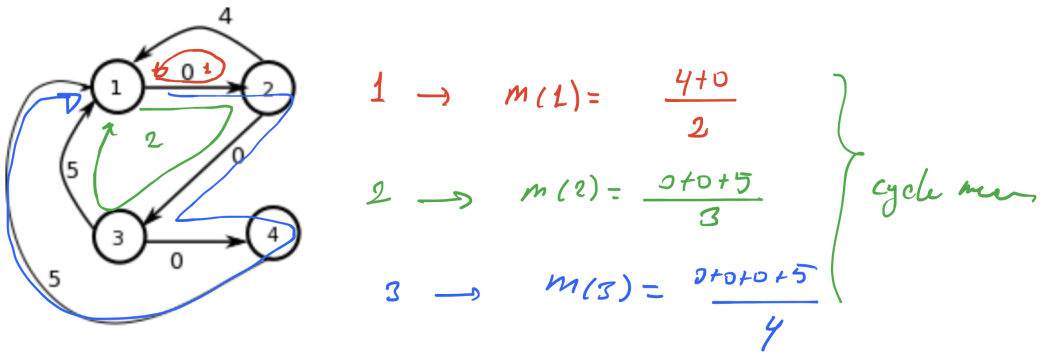


Two cycles contain the delay D_α and D_β in G and their comp. time 6 and 4 u.t.
The sum of the edge weights in the cycle G_d is 6 which is the max. comp. time of the two cycles in G.

Minimum Cycle Mean Method (MCM)

In G_d, the number of edges in a cycle equals the number of nodes in the cycle ! and this equals the number of delays in G.

$$\text{cycle mean}(m_c) = \frac{\sum \text{edge weights}}{\#\text{ of edges in the cycle}}$$



The maximum cycle mean of G_d is the maximum cycle bound of all cycles in G which is the iteration bound of G.

Minimum Cycle Mean Method (MCM)

The maximum cycle mean of G_d is simply the Minimum Cycle Mean (MCM) of $\bar{G}_d$ multiplyied by -1.

MCM Algorithm:

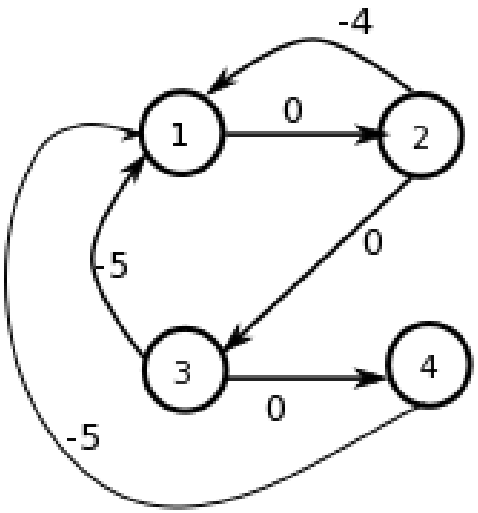
- The initial vector $f^{(0)}$ is formed by setting $f^{(0)}(s) = 0$ (s an arbitrary node in $\bar{G}_d$) and setting the remaining of $f^{(0)}$ to ∞ ($f^{(0)} = [0, \infty, \infty, \infty]^T$).
- The remaining vector $f^{(m)}$ are computed by

$$f^{(m)}(j) = \min_{i \in I}(f^{(m-1)}(i) + \bar{w}(i,j))$$

where I is the set of nodes in $\bar{G}_d$ such that there exists an edge from i to j.

We will find $d+1$ vectors , $f^{(m)}$ where $m = 0, 1, ..., d$ and dim. $d \times 1$.

Minimum Cycle Mean Method (MCM)



$$f^{(1)}(j) = \min_{i \in I}(f^{(0)}(i) + \bar{w}(i,j))$$

$$f^{(1)}(1) = \min_{i \in I}(f^{(0)}(i) + \bar{w}(i,1))$$

$$= \min_{i \in I}(f^{(0)}(2) + \bar{w}(2,1), f^{(0)}(3) + \bar{w}(3,1)), f^{(0)}(4) + \bar{w}(4,1)$$

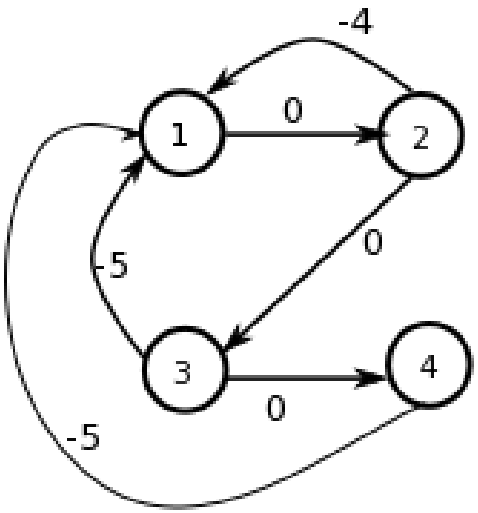
$$= \min_{i \in I}(\infty - 4, \infty - 5, \infty - 5)$$

$$f^{(1)}(2) = \min_{i \in I}(f^{(0)}(1) + \bar{w}(1,2))$$

$$= \min_{i \in I}(0 + 0)$$

where I is the set of nodes in $\bar{G}_d$ such that there exists an edge from i to j. $f^{(1)} = [\infty, 0, \infty, \infty]^T$

Minimum Cycle Mean Method (MCM)



$$f^{(2)}(j) = \min_{i \in I}(f^{(1)}(i) + \bar{w}(i,j))$$

$$f^{(2)}(1) = \min_{i \in I}(f^{(1)}(i) + \bar{w}(i,1))$$

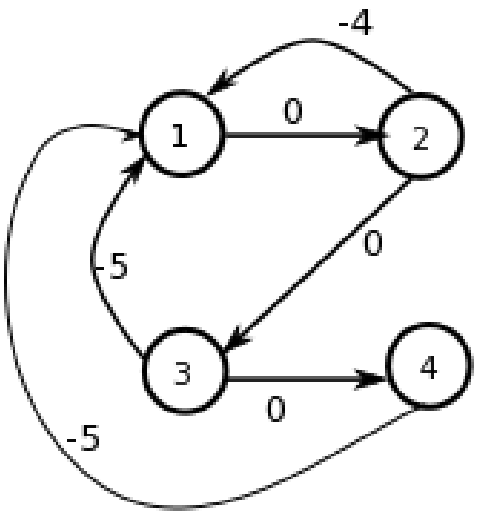
$$= \min_{i \in I}(f^{(1)}(2) + \bar{w}(2,1), f^{(1)}(3) + \bar{w}(3,1)), f^{(1)}(4) + \bar{w}(4,1)$$

$$= \min_{i \in I}(0 - 4, \infty - 5, \infty - 5)$$

$$f^{(2)}(2) = \min_{i \in I}(f^{(1)}(1) + \bar{w}(1,2)) = \min_{i \in I}(+0)$$

where I is the set of nodes in $\bar{G}_d$ such that there exists an edge from i to j. $f^{(2)} = [-4, \infty, 0, \infty]^T$

Minimum Cycle Mean Method (MCM)



$$f^{(3)}(j) = \min_{i \in I}(f^{(2)}(i) + \bar{w}(i,j))$$

$$f^{(3)}(1) = \min_{i \in I}(f^{(2)}(i) + \bar{w}(i,1))$$

$$= \min_{i \in I}(f^{(2)}(2) + \bar{w}(2,1), f^{(2)}(3) + \bar{w}(3,1)), f^{(2)}(4) + \bar{w}(4,1)$$

$$= \min_{i \in I}(\infty - 4, 0 - 5, \infty - 5)$$

$$f^{(3)}(2) = \min_{i \in I}(f^{(2)}(1) + \bar{w}(1,2)) = \min_{i \in I}(-4 + 0)$$

where I is the set of nodes in $\bar{G}_d$ such that there exists an edge from i to j. $f^{(3)} = [-5, -4, \infty, 0]^T$

Minimum Cycle Mean Method (MCM)

Iteration bound is

$$T_\infty = - \min_{i \in \{1,2,\ldots d\}} \left(\max_{m \in \{0,1,2,\ldots d-1\}} \left(\frac{f^{(d)}(i) - f^{(m)}(i)}{d-m} \right) \right)$$

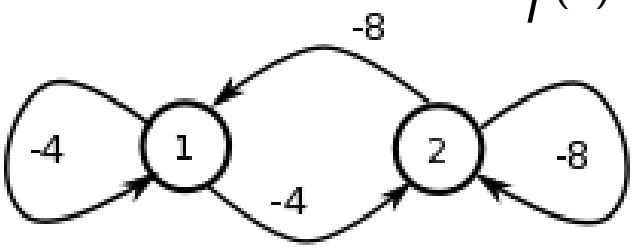
$f^{(0)} = [0, \infty, \infty, \infty]^T$, $f^{(1)} = [\infty, 0, \infty, \infty]^T$, $f^{(2)} = [-4, \infty, 0, \infty]^T$, $f^{(3)} = [-5, -4, \infty, 0]^T$, $f^{(4)} = [-8, -5, -4, \infty]^T$

$$T_\infty = - \min(\max(\frac{-8-0}{4}, \frac{-8-\infty}{3}, \frac{-8+4}{2}, \frac{-8+5}{1}),$$

$$\max(\frac{-5-\infty}{4}, \frac{-5-0}{3}, \frac{-5-\infty}{2}, \frac{-5+4}{1}), \ldots) = 2$$

▸ Minimum Cycle Mean Method Algorithm

Minimum Cycle Mean Method (MCM)



$$f^{(0)} = [0, \infty]^T \; f^{(1)} = [-4, -4]^T \; f^{(2)} = [-12, -12]^T$$

$$T_\infty = -\min(-8, -6) = 8$$

- 1. Draw a DFG for a 4th-order IIR digital filter implemented as cascade of two 2nd-order sections. Assume each multiplication requires and addition requires 4 u.t. and 2 u.t. What is the critical path of this DFG? What is the iteration bound of this DFG?
- 2. Repeat the previous question for a direct form 4th-order IIR filter.
- 3. Consider an FIR filter $y(n) = a_0x(n) + a_2x(n-2) + a_3x(n-3)$ assume that the time required for 1 multipliy operation is 4 u.t. and the time required for 1 add operation is 2 u.t.. Draw two types of direct-form realization structures of the above filter. What is the critical path length? What is the iteration bound?