ELE617E
Lectures

Prof. Dr. Müştak E. Yalçın

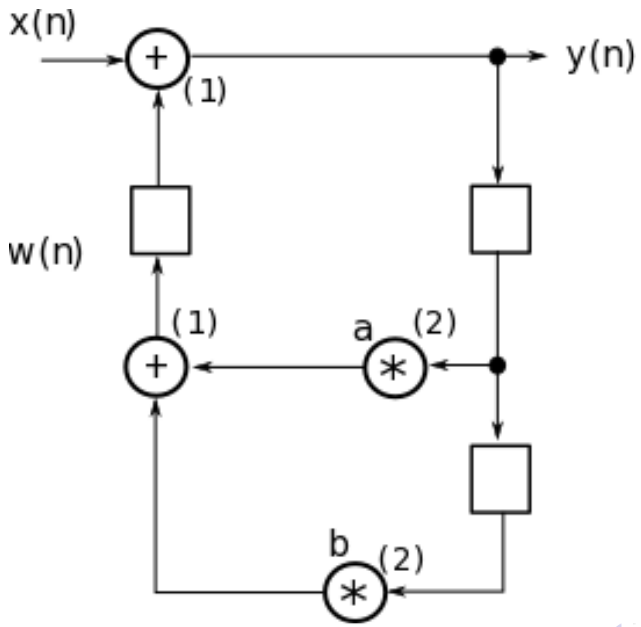
Istanbul Technical University

mustak.yalcin@itu.edu.tr

Retiming

Retiming is a transformation technique used to change the locations of delay elements without affecting the I/O characteristics.

$$y(n) = ay(n-2) + by(n-3) + x(n)$$
$$w(n) = ay(n-1) + by(n-2)$$
$$y(n) = w(n-1) + x(n)$$

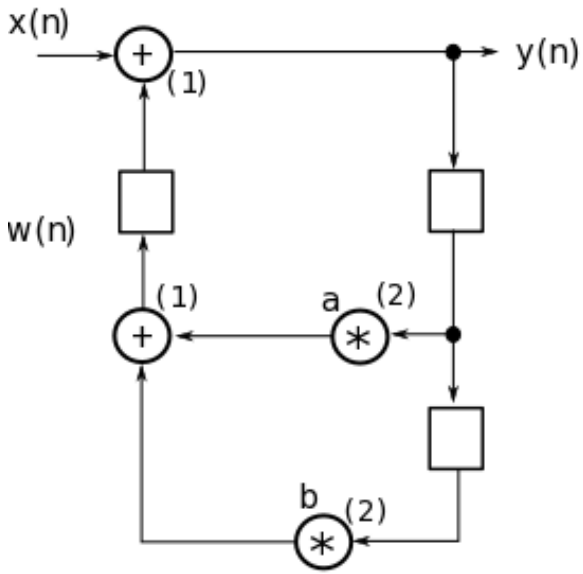


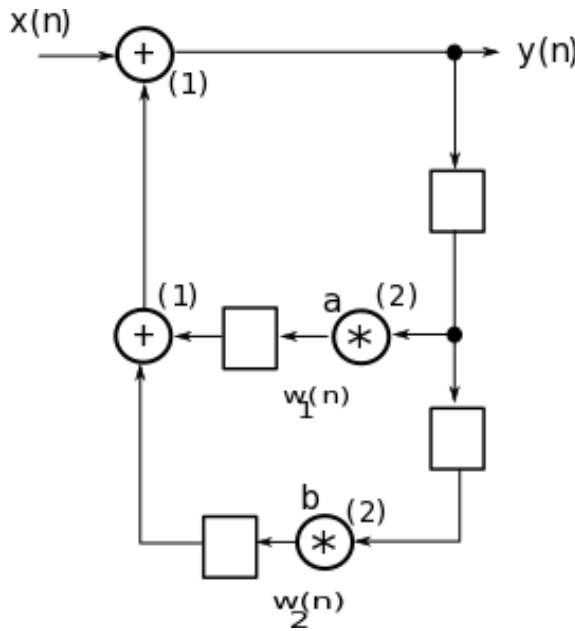
Retiming

Retiming is a transformation technique used to change the locations of delay elements without affecting the I/O characteristics.

$$y(n) = ay(n-2) + by(n-3) + x(n)$$

$$w_1(n) = ay(n-1)$$
$$w_2(n) = by(n-2)$$



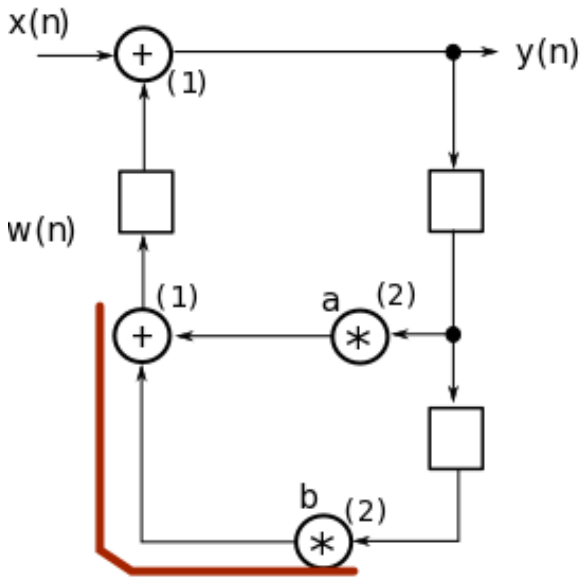


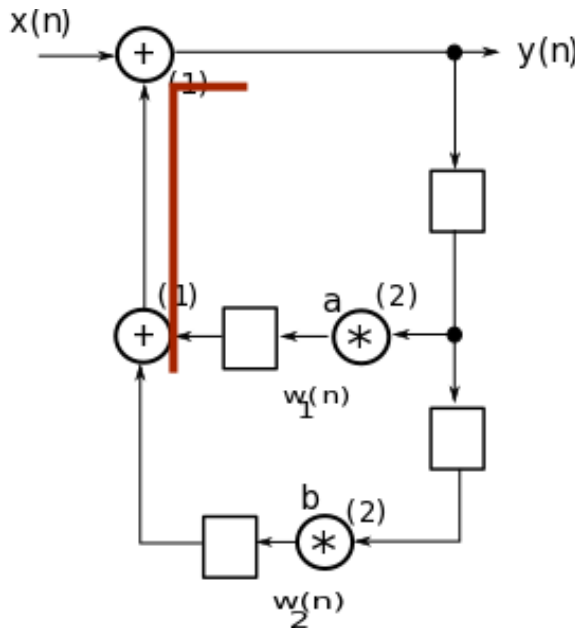
Retiming

Retiming is a transformation technique used to change the locations of delay elements without affecting the I/O characteristics.

$$y(n) = ay(n-2) + by(n-3) + x(n)$$

$$w_1(n) = ay(n-1)$$
$$w_2(n) = by(n-2)$$

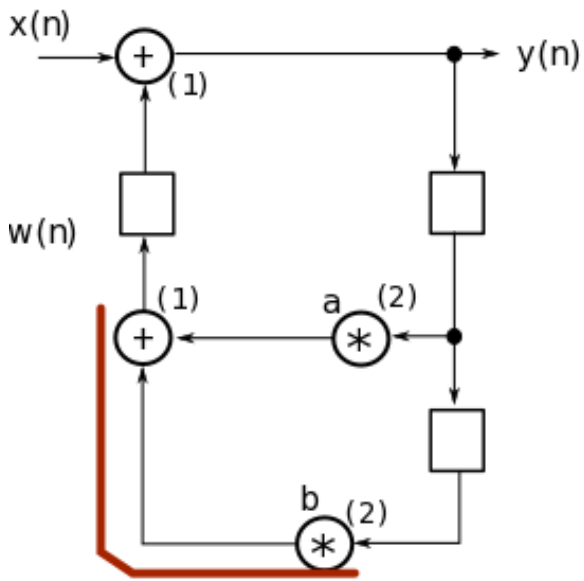


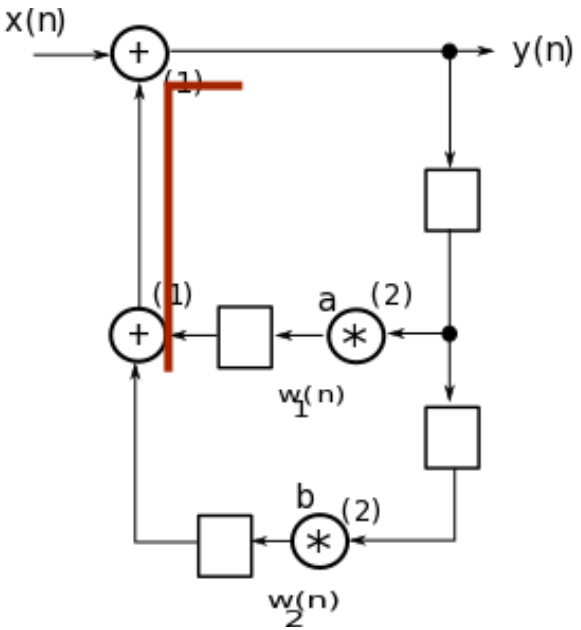


Retiming

Aims of Retiming:

- Reducing the clock period.
- Reducing the the number of registers.
- Reducing the power consumption. Section 17.5.4
- Reducing the logic synthesis. –



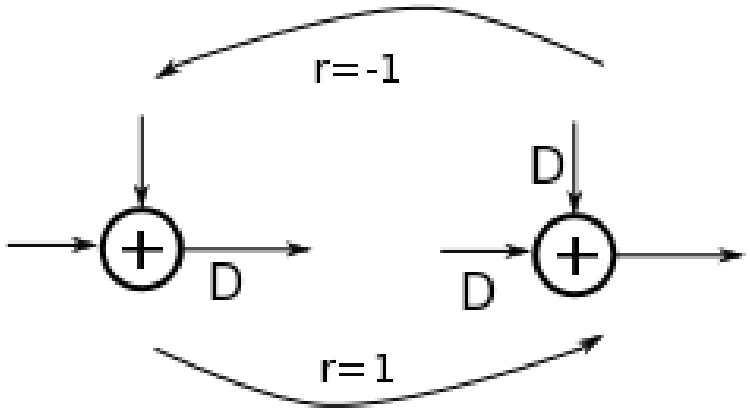


$T_{\text{critical}} = T_A + T_M$ and then $T_{clk} = 2T_A$

Retiming

Retiming maps a data-flow-graph **G** to a retimed graph $\mathbf{G_r}$.

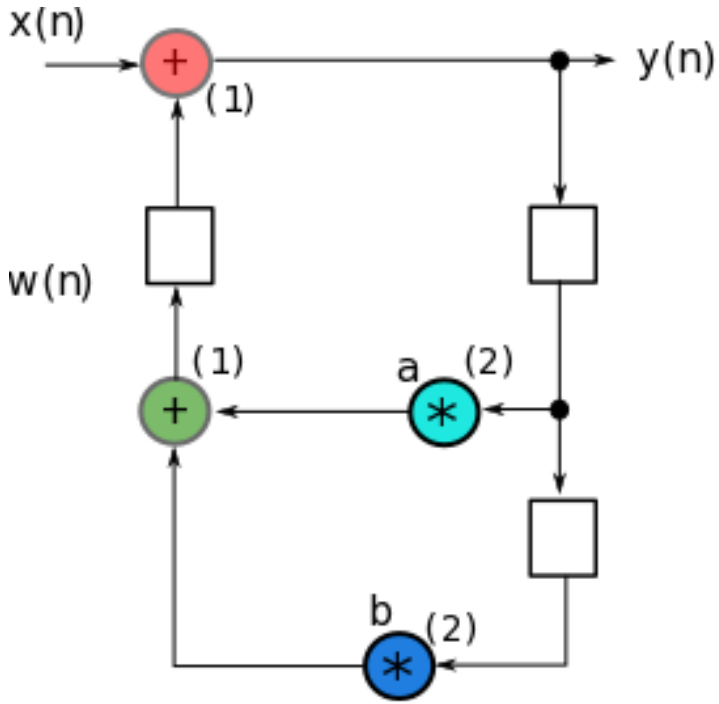
- A retimed solution is characterized by a value $r(U) \in Z$ (the set of integers) known as the retiming weight for each node U in the graph. $r(U)$ lag of U represents the number of registers that are to be moved in the circuit from each out-edge of U to each of its in-edges ! ▸ Retiming and Resynthesis: ..., IEEE TCAD, 1991

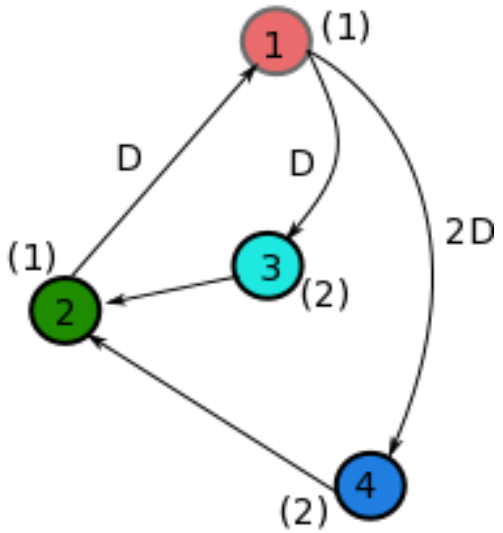


- $w_r(e)$ denote the weight of the edge $e : U \rightarrow V$ in the retimed graph $\mathbf{G_r}$

$$w_r(e) = w(e) + r(V) - r(U)$$

Retiming



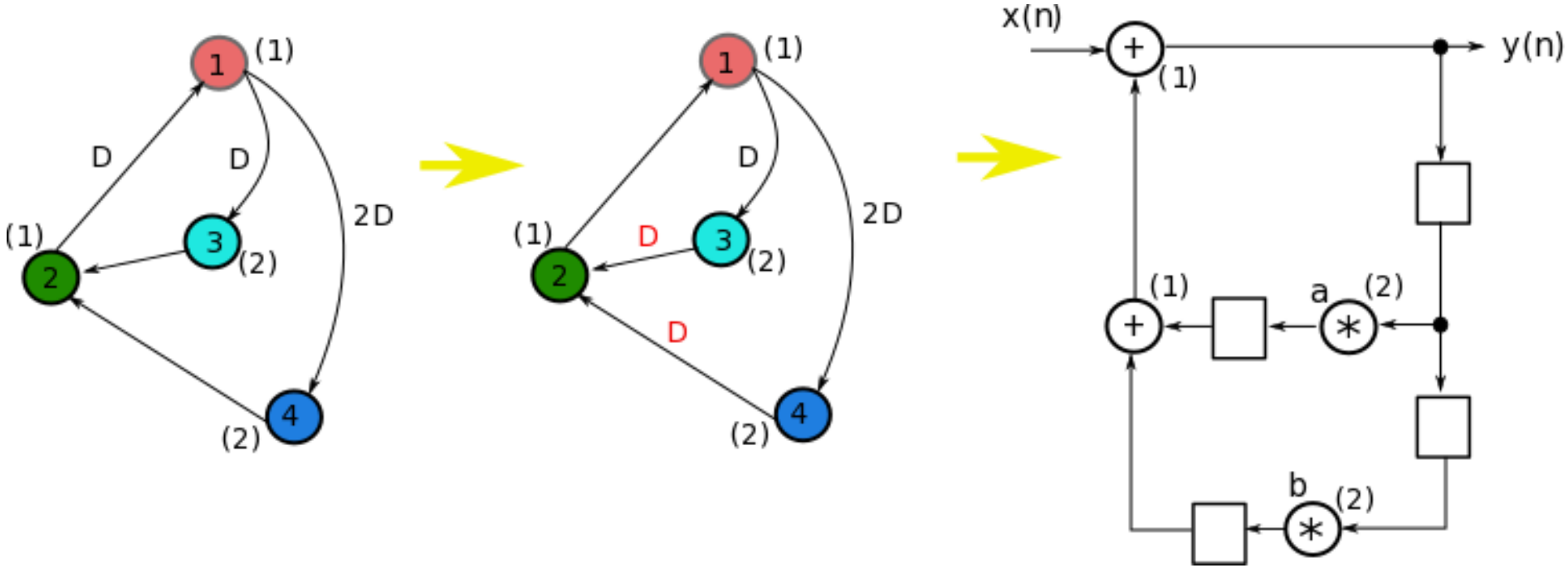


Let use $r(1) = 0, r(2) = 1, r(3) = 0, r(4) = 0$

$$w_r(2 \to 1) = w(2 \to 1) + r(1) - r(2) = 1 + 0 - 1 = 0$$
$$w_r(1 \to 4) = w(1 \to 4) + r(4) - r(1) = 2 + 0 - 0 = 2$$
$$w_r(3 \to 2) = w(3 \to 2) + r(2) - r(3) = 0 + 1 - 0 = 1$$
$$w_r(4 \to 2) = w(4 \to 2) + r(2) - r(4) = 0 + 1 - 0 = 1$$

Retiming

$$w_r(2 \to 1) = 0, w_r(1 \to 4) = 2, w_r(3 \to 2) = 1, w_r(4 \to 2) = 1$$



$$\mathbf{w} = [1\,2\,0\,0\,1] \to \mathbf{w_r} = [1\,2\,1\,1\,0]$$

$$T_{clk} = T_{\text{critical}} = \sum_{4 \to 2} d(p) = 2 + 1 \text{ after the retiming}$$

$$T_{clk} = \sum_{2 \to 1} d(p) = 1 + 1$$

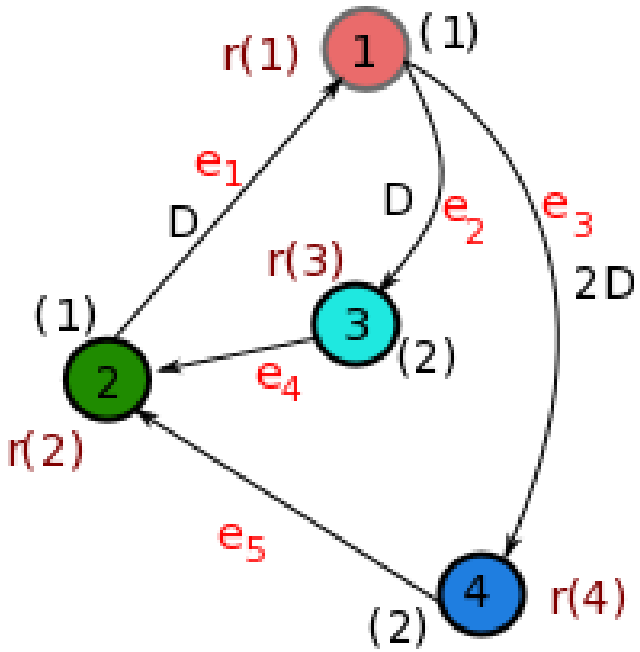
Retiming

- A retiming solution is feasible if $w_r(e) \geq 0$ holds for all edges.
- Retiming does not alter the total number of delays in a loop of the DFG.

$$B\mathbf{w} = B\mathbf{w_r}$$

  Since the number of delays in a loop is the same before and after retiming, retiming cannot change the iteration bound of the DFG.
- The weight of the retimed path $s\mathbf{p} = V_0 \rightarrow V_1 \rightarrow ...V_k$
  $w_r(\mathbf{p}) = w(\mathbf{p}) + r(V_k) - r(V_1)$
- Adding the constant value j to the retiming value of each node does not alter the number of delays in the edges of the retimed graph.
  $w_r(e) = w(e) + (r(V_k) + j) - (r(V_1) + j)$

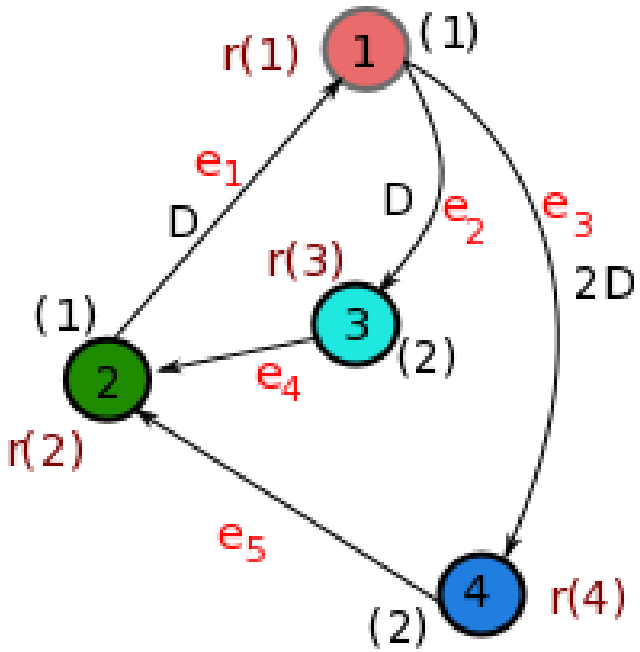
Retiming



$$w_r(e_1) = w(e_1) + r(1) - r(2)$$
$$w_r(e_2) = w(e_2) + r(3) - r(1)$$
$$w_r(e_3) = w(e_3) + r(4) - r(1)$$
$$w_r(e_4) = w(e_4) + r(2) - r(3)$$
$$w_r(e_5) = w(e_5) + r(2) - r(4)$$

Feasible retiming solution for r must ensure:

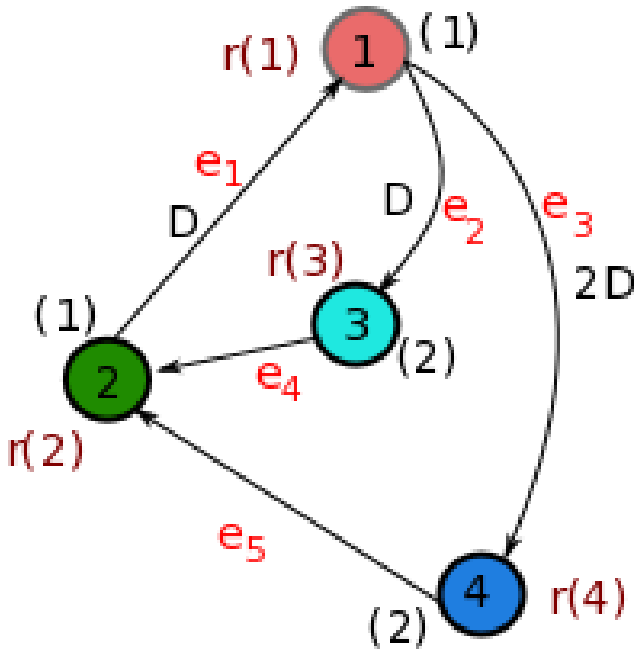
Retiming



$$w_r(e_1) = w(e_1) + r(1) - r(2) \geq 0$$
$$w_r(e_2) = w(e_2) + r(3) - r(1) \geq 0$$
$$w_r(e_3) = w(e_3) + r(4) - r(1) \geq 0$$
$$w_r(e_4) = w(e_4) + r(2) - r(3) \geq 0$$
$$w_r(e_5) = w(e_5) + r(2) - r(4) \geq 0$$

Feasible retiming solution for r must ensure: Non-negative edge weights $w_r(e)$,

Retiming



$$
\begin{aligned}
w_r(e_1) &= w(e_1) + r(1) - r(2) \geq 0 \\
w_r(e_2) &= w(e_2) + r(3) - r(1) \geq 0 \\
w_r(e_3) &= w(e_3) + r(4) - r(1) \geq 0 \\
w_r(e_4) &= w(e_4) + r(2) - r(3) \geq 0 \\
w_r(e_5) &= w(e_5) + r(2) - r(4) \geq 0
\end{aligned}
$$

$$
\begin{aligned}
w_r(e_1) &= 1 + r(1) - r(2) \geq 0 \\
w_r(e_2) &= 1 + r(3) - r(1) \geq 0 \\
w_r(e_3) &= 2 + r(4) - r(1) \geq 0 \\
w_r(e_4) &= r(2) - r(3) \geq 0 \\
w_r(e_5) &= r(2) - r(4) \geq 0
\end{aligned}
$$

Feasible retiming solution for r must ensure: Non-negative edge weights $w_r(e)$, Integer values of r and w_r.

Retiming

Feasible retiming solution for r must ensure: Non-negative edge weights $w_r(e)$, and Integer values of r and w_r.

Use Bellman-Ford/Folyd-Warshall algorithms to solve the inequalities;

$r(2) - r(1) \leq -1$

$r(1) - r(3) \leq -1$

$r(1) - r(4) \leq -2$

$r(3) - r(2) \leq 0$

$r(4) - r(2) \leq 0$

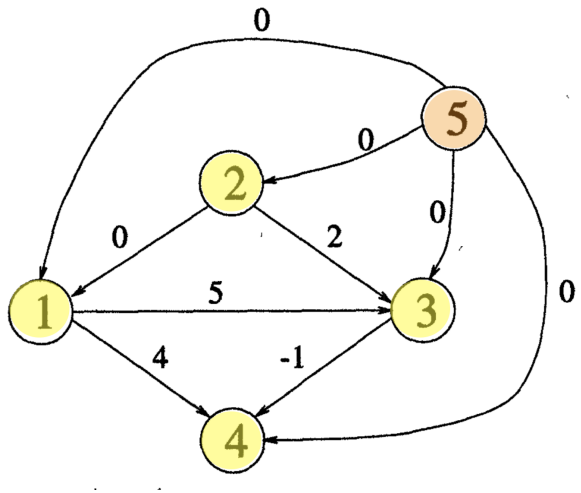
▸ Bellman-Ford.m

Solving Systems of Inequalities!

Shorthest path Alg. can be used to determine if a solution exists !

Page 95, Draw a constraint graph:

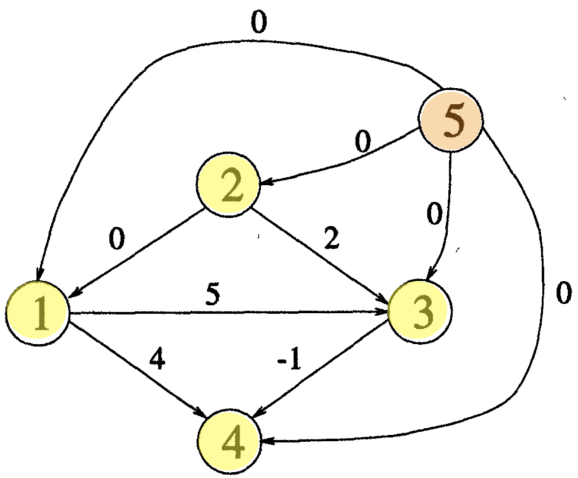
$$r(V_1) - r(V_2) \leq 0$$
$$r(V_3) - r(V_1) \leq 5$$
$$r(V_4) - r(V_1) \leq 4$$
$$r(V_4) - r(V_3) \leq -1$$
$$r(V_3) - r(V_2) \leq 2$$



Solving using SPA:

- iff no negative cycles.
- r_i is the min. lenght path from V_{N+1} to V_i.

Folyd-Warshall Alg.



- Construct an initial matrix from the graph $\mathbf{R}^{(1)}(U, V)$

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

- for k=1 to n

$$\mathbf{R}^{(k+1)}(U, V) = min\{\mathbf{R}^{(k)}(U, V), r^{(k)}(U, k) + r^{(k)}(k, V)\}$$

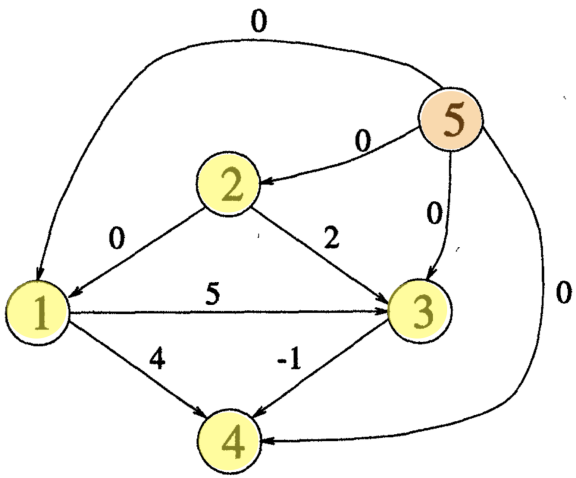
- the last row in $\mathbf{R}^{(k+1)}(U, V)$ gives a solution

$$r_V - r_n \leq \mathbf{R}^{(n+1)}(n, V)$$

set $r_n = 0$ gives $r_v = \mathbf{R}^{(n+1)}(n, V)$ as a solution.

▸ Folyd_Warshall.m

Folyd-Warshall Alg.



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

The shortest path from node U to node V between direct edge and the path including node 1.

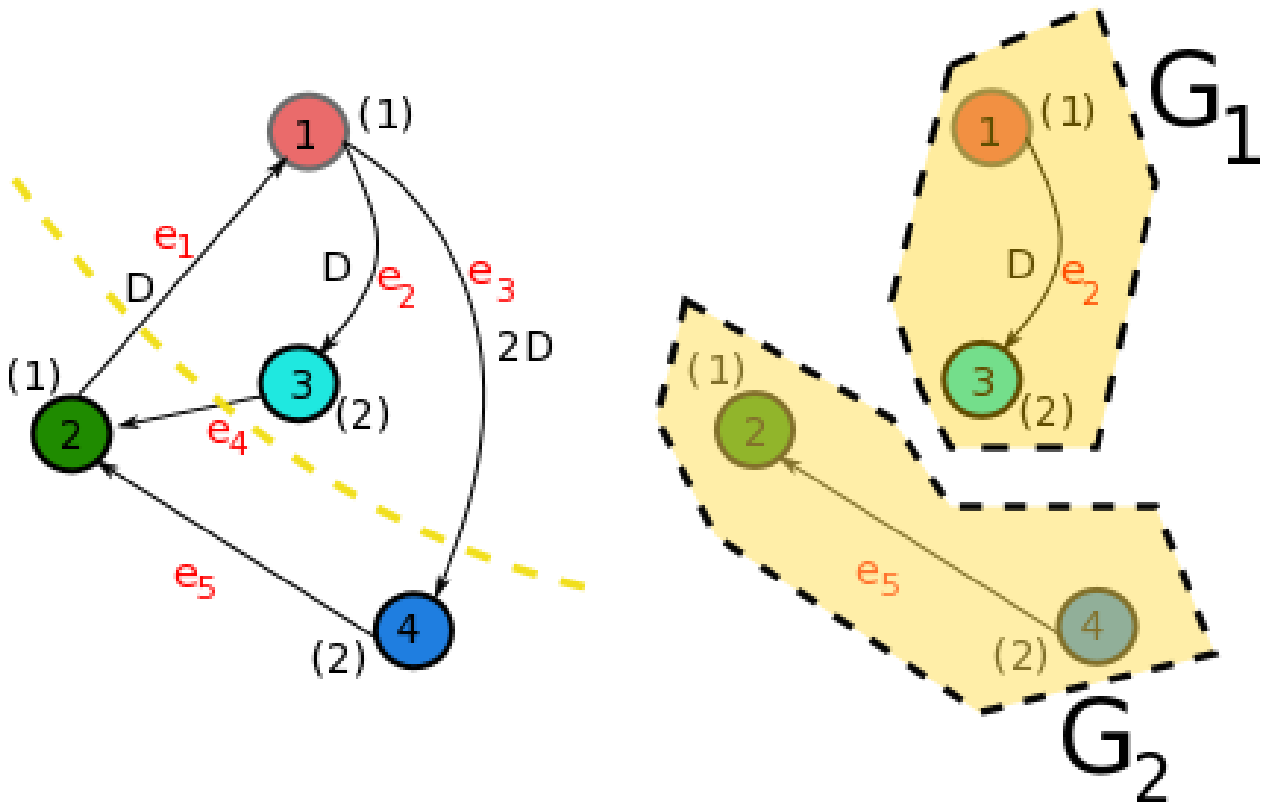
$$\mathbf{R}^{(2)} = \begin{bmatrix} \infty & \infty & 5 & 4 & \infty \\ 0 & \infty & 2 & 4 & \infty \\ \infty & \infty & \infty & -1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & 0 & 0 & \infty \end{bmatrix}$$

$$\mathbf{R}^{(3)} = \begin{bmatrix} \infty & \infty & 5 & 4 & \infty \\ 0 & \infty & 2 & 4 & \infty \\ \infty & \infty & \infty & -1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & 0 & 0 & \infty \end{bmatrix} \quad \mathbf{R}^{(4)} = \begin{bmatrix} \infty & \infty & 5 & 4 & \infty \\ 0 & \infty & 2 & 4 & \infty \\ \infty & \infty & \infty & -1 & \infty \\ \infty & \infty & \infty & \infty & \infty \\ 0 & 0 & 0 & -1 & \infty \end{bmatrix},$$

$\mathbf{R}^{(4)} = \mathbf{R}^{(5)} = \mathbf{R}^{(6)}$ ▸ Folyd_Warshall.m

Cut-set Retiming

A Cut-set is a set of edges that can be removed from the graph to create 2 disconnected subgraphs.



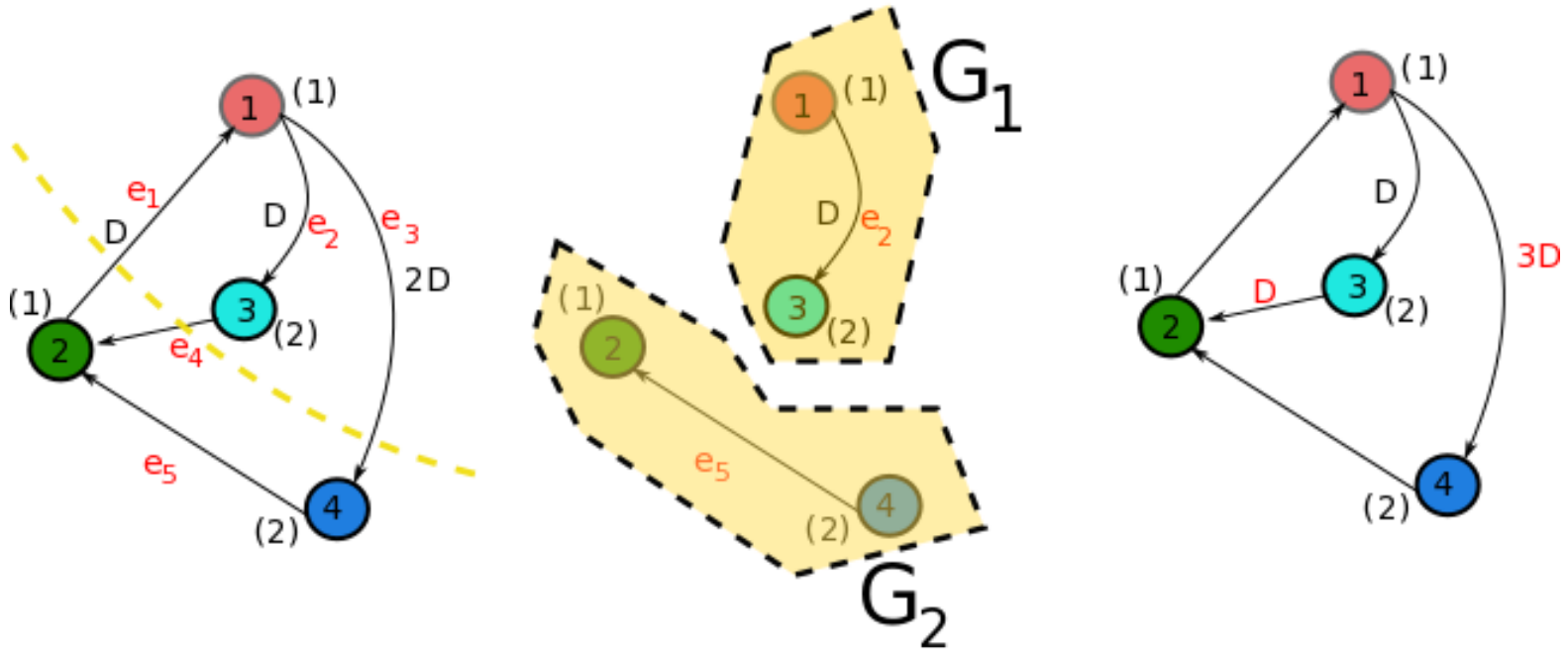
e_3 and e_4: $G_1 \rightarrow G_2$, e_1 : $G_2 \rightarrow G_1$

Cut-set retiming with k = adding k delays to each edge $G_1 \rightarrow G_2$ and removing k delays to each edge $G_2 \rightarrow G_1$.

Cut-set Retiming

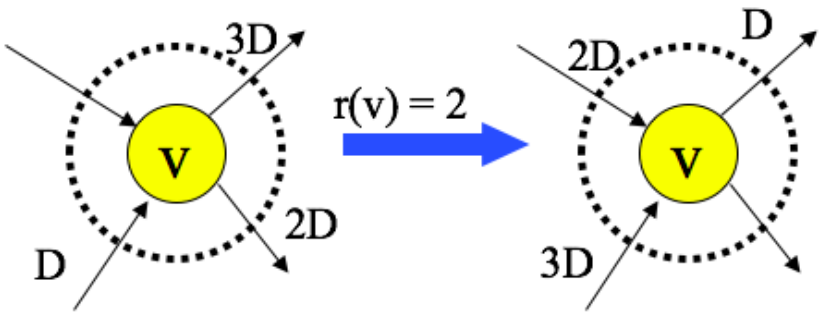
Condition on k is

$$-\min_{G_1 \to G_2}\{w(e)\} \leq k \leq \min_{G_2 \to G_1}\{w(e)\}$$

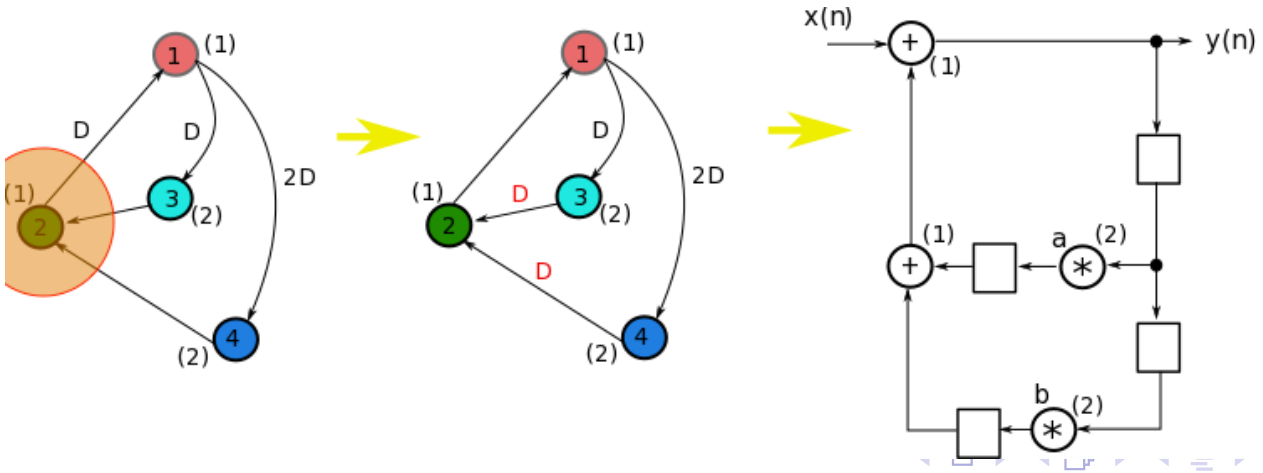


Node Retiming

G_2 is a single node!

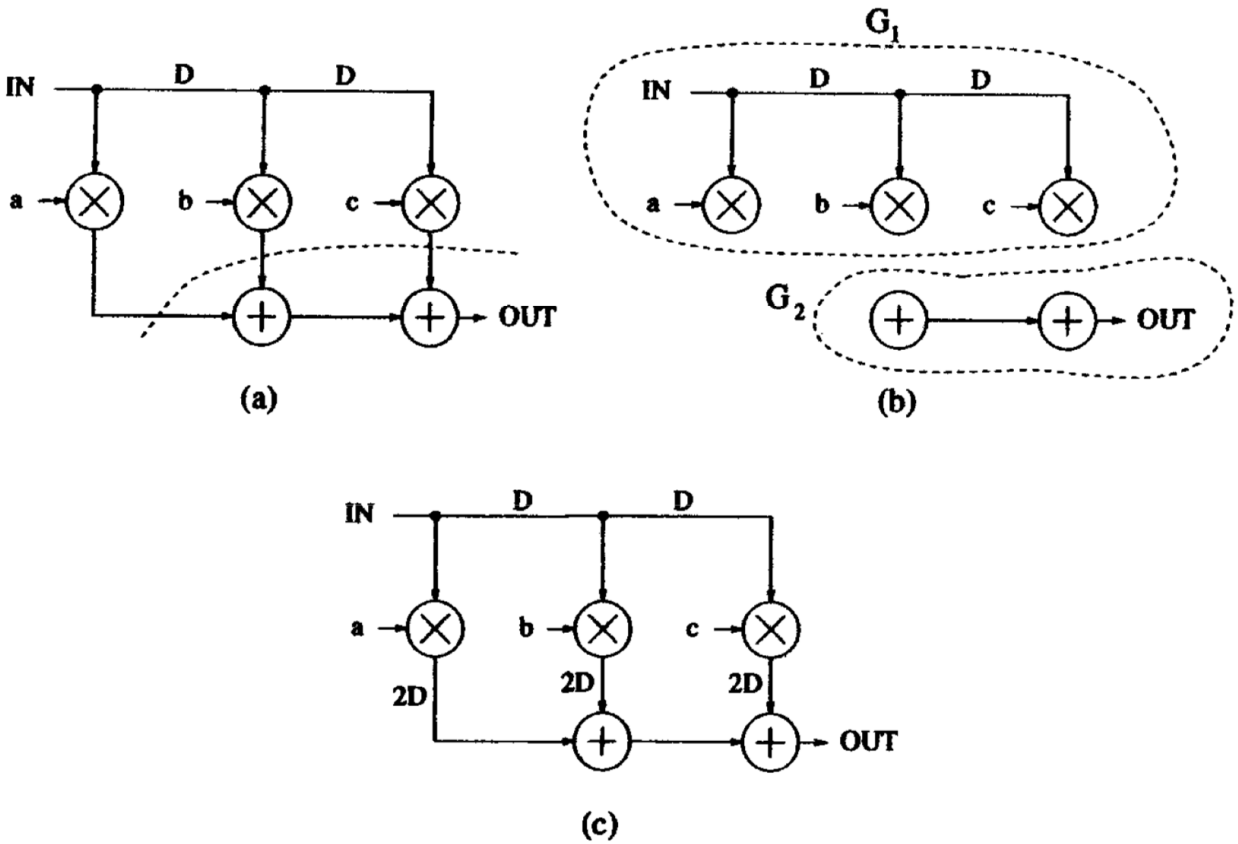


$r(v) = \#$ of delays transfered from out-going edges to incoming edges of node v, $w(e) = \#$ of delays on edge e, $w_r(e) = \#$ of delays on edge e after retiming.



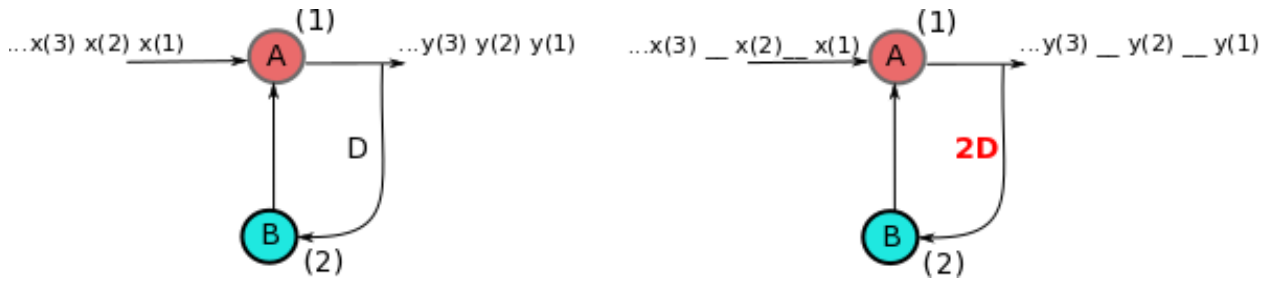
Pipelining

Pipelining is a special case of cutset retiming where are no edges in the cutset from the subgraph G_2 to the subgraph G_1, i.e, pipelining applies to graph without loops.
These cutsets are referred to as feed-forward cutsets.



Pipelining

- Cutset retiming is often used in combination with slow-down.
- The procedure is to first replace each delay in the DFG with N delays to create an N-slow version of the DFG and then to perform cutset retiming on the N-slow DFG.
- Note that in an N-slow system, N-1 null operations (or 0 samples) must be interleaved after each useful signal sample to preserve the functionality of the algorithm.

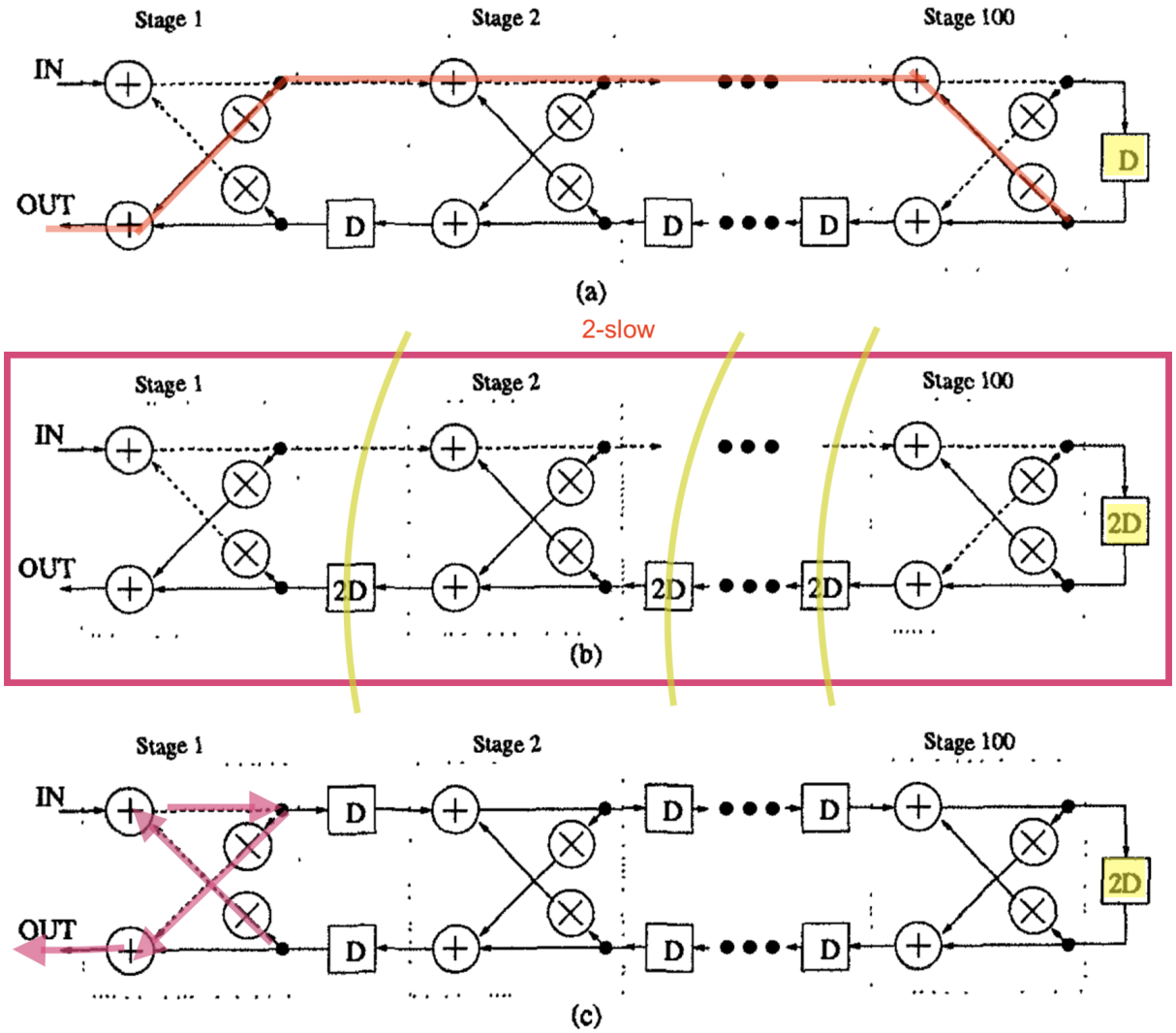


In slow down the clock cycle time remains unchanged. Only the sampling time is increased.

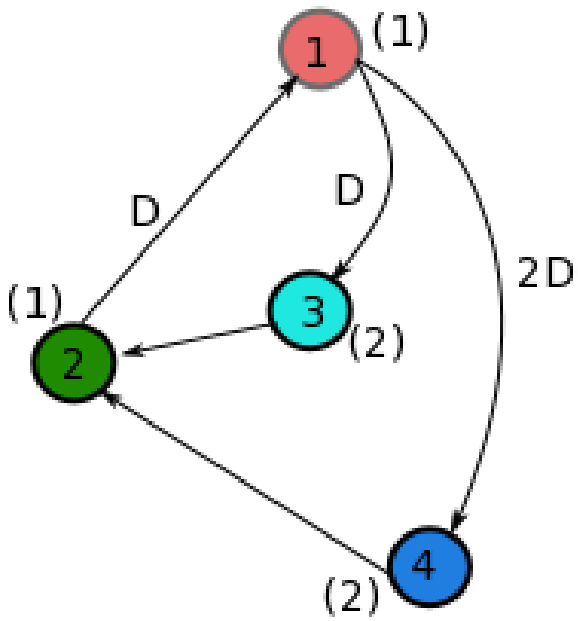
Pipelining

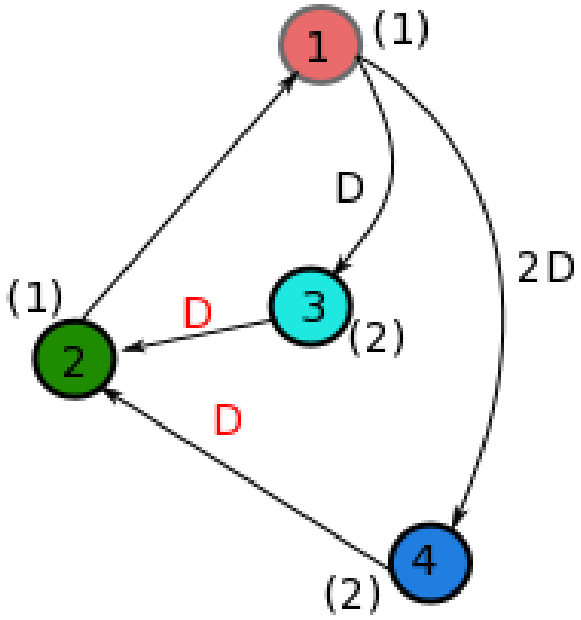
- Cutset retiming is often used in combination with slow-down.
- The procedure is to first replace each delay in the DFG with N delays to create an N-slow version of the DFG and then to perform cutset retiming on the N-slow DFG.
- Note that in an N-slow system, N-1 null operations (or 0 samples) must be interleaved after each useful signal sample to preserve the functionality of the algorithm.
- For example, cosider the 100-stage lattice filter, which has a critical path of 101 adders and 2 multipliers. Thus, the minimum sample period is $(101)(1) + (2)(2) = 105$ unit time.
- If the 2-slow version of the circuit is used, then it has the computation time of $(2)(1) + (2)(2) = 6$ unit time and requires the minimum sample period is $(2)(6) = 12$ unit time.

Pipelining


Stage 1
Stage 2
Stage 100
IN
OUT
D
(a)
2-slow
Stage 1
Stage 2
Stage 100
IN
OUT
2D
(b)
Stage 1
Stage 2
Stage 100
IN
OUT
D
2D
(c)

Retiming





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

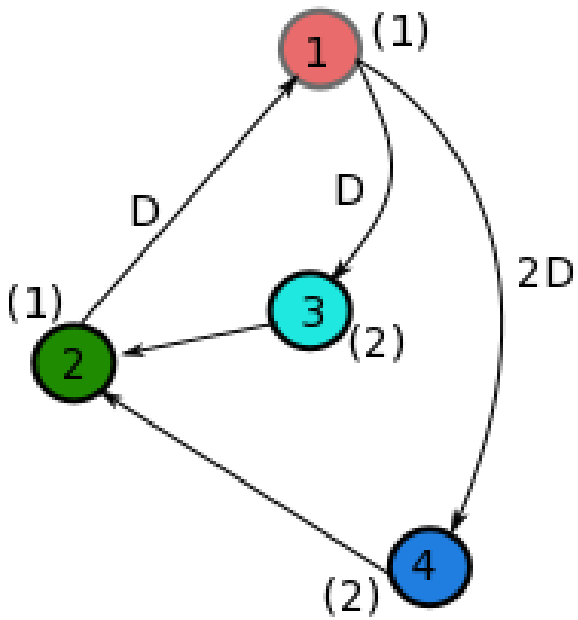
$$T_{clk} = 3u.t. \qquad T_{clk} = 2u.t.$$

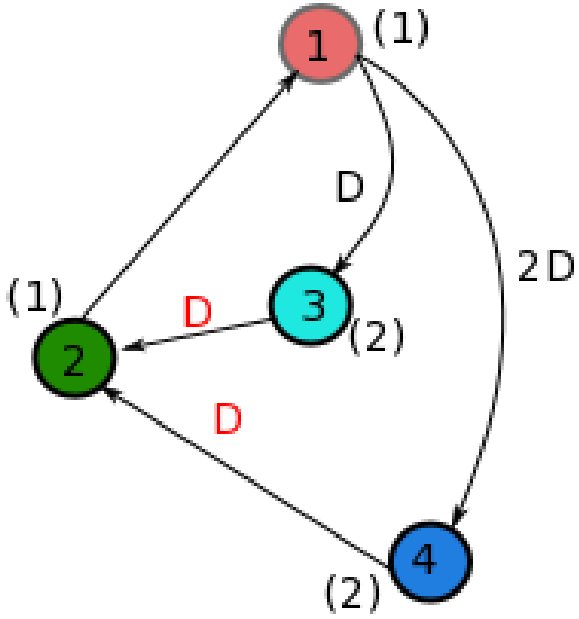
$$w_r(e_1)+w_r(e_2)+w_r(e_4) =$$

$$w(e_1) + r(1) - r(2) + w(e_2) + r(3) - r(1) + w(e_4) + r(2) - r(3)$$

$$= w(e_1) + w(e_2) + w(e_4)$$

Retiming





$$T_\infty = \max\{\frac{4}{2}, \frac{4}{3}\},$$
$$T_{clk} = 3u.t.$$

$$T_\infty = \max\{\frac{4}{2}, \frac{4}{3}\}$$
$$T_{clk} = 2u.t.$$

- Retiming does NOT change the total number of delays for each cycle.
- Retiming does not change loop bound or iteration bound of the DFG

Retiming for Clock Periode Minimization

$$T_{clk} = \max_{p:w(p)=0} \{d(p)\}$$

such that p is a path $V_0 \xrightarrow{e_0} V_1 \xrightarrow{e_1} V_2 \ldots \xrightarrow{e_L} V_N$, and

$$d(p) = \sum_{k=0}^{N} d(V_k), \text{and } w(p) = \sum_{k=0}^{L} w(e_k).$$

Mathematically, the minimum feasible clock periode is

$$\Phi(G) = \max_{p:w(p)=0} \{d(p)\}$$

and find a legal retiming r so that $\Phi(G_r) \leq \Phi(G)$ or

$$\min_r \Phi(G_r) \text{ such that r is legal}$$

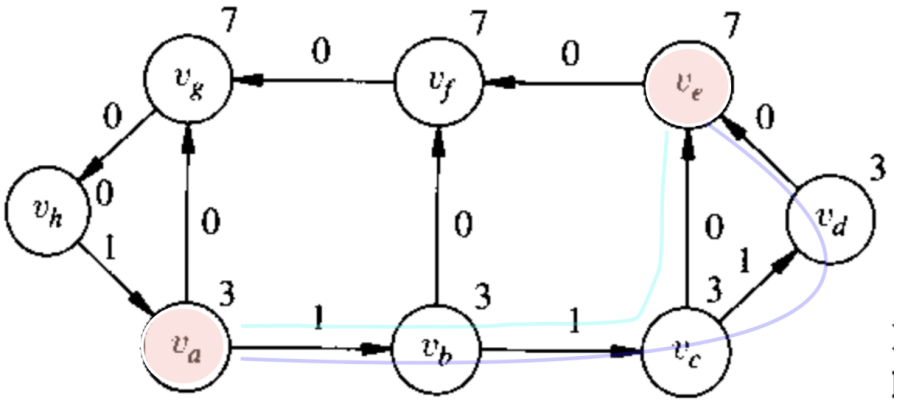
Retiming for Clock Periode Minimization

- Register weight matrix: The minimum number of registers on any path $U \rightarrow V$

$$W(U,V) = \min\{w(p) : U \xrightarrow{p} V\}$$

- Delay matrix: The maximum delay $(D(U,V))$ on a path (p) with minimum register count $(W(U,V))$ between two nodes:

$$D(U,V) = \max\{d(p) : U \xrightarrow{p} V \& w(p) = W(U,V)\}$$



$W(V_a, V_e) = 2$ and $D(V_a, V_e) = 16$

Retiming for Clock Periode Minimization

Retiming will NOT alter iteration bound (Iteration bound is the theoretical minimum clock period to execute the algorithm)!

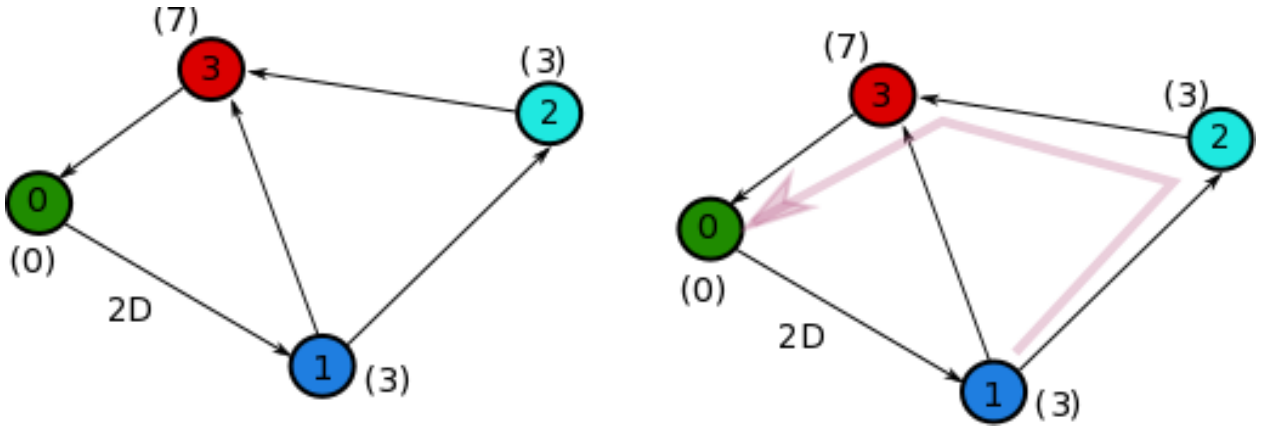
If the node computing time $d(p) = \sum_{i=0}^{j} d(V_i) > T_\infty$, then clock period $T > T_\infty$. For such an edge, we require that $w_r(p) \geq 1$

$V_0 \xrightarrow{e_0} V_1 \xrightarrow{e_1} V_2 \ldots \xrightarrow{e_L} V_{L+1}$

$$
\begin{aligned}
w_r(p) &= \sum_{i=0}^{L} w_r(e_i) = \sum_{i=0}^{L} (w(e_i) + r(V_{i+1}) - r(V_i)) \\
&= W(V_0, V_L) + r(V_L) - r(V_0) \geq 1 \\
r(V_0) - r(V_L) &\leq W(V_0, V_L) - 1
\end{aligned}
$$

Retiming for Clock Periode Minimization

$d(p) > 7$ must be considered!. Critical path constraint:



W

	V_0	V_1	V_2	V_3
V_0	0	2	2	2
V_1	0	0	0	0
V_2	0	2	0	0
V_3	0	2	2	0

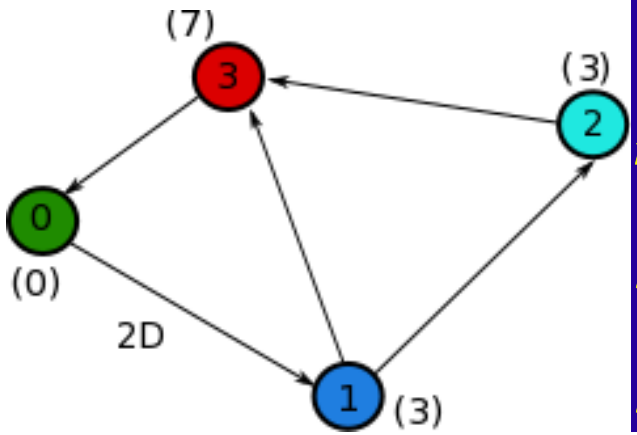
D

	V_0	V_1	V_2	V_3
V_0	0	3	6	13
V_1	13	3	6	13
V_2	10	13	3	10
V_3	10	13	13	7

Alg. for for computing W and D : page 101

Retiming for Clock Periode Minimization

$$r(V_0) - r(V_L) \leq W(V_0, V_L) - 1$$
$$r(V_0) - r(V_3) \leq 2 - 1$$
$$r(V_1) - r(V_0) \leq 0 - 1$$
$$r(V_1) - r(V_3) \leq 0 - 1$$
$$\ldots \leq \ldots$$



$$r(v_0) - r(v_3) \leq 1$$
$$r(v_1) - r(v_0) \leq -1$$
$$r(v_1) - r(v_3) \leq -1$$
$$r(v_2) - r(v_0) \leq -1$$
$$r(v_2) - r(v_1) \leq 1$$
$$r(v_2) - r(v_3) \leq -1$$
$$r(v_3) - r(v_1) \leq 1$$
$$r(v_3) - r(v_2) \leq 1$$

W

	V_0	V_1	V_2	V_3
V_0	0	2	2	2
V_1	0	0	0	0
V_2	0	2	0	0
V_3	0	2	2	0

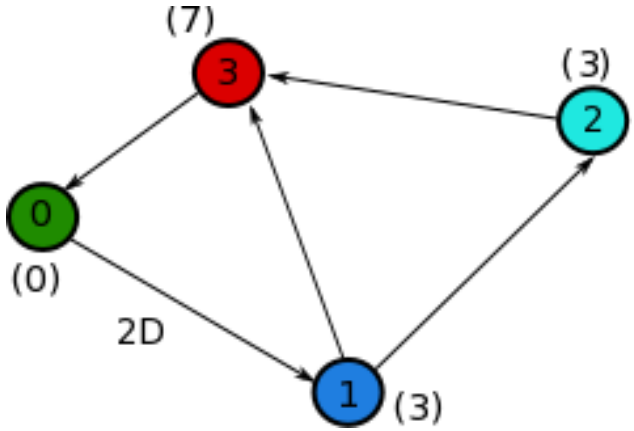
D

	V_0	V_1	V_2	V_3
V_0	0	3	6	13
V_1	13	3	6	13
V_2	10	13	3	10
V_3	10	13	13	7

Retiming for Clock Periode Minimization

$d(p) < 7$. Feasibility constraint :

$$r(V_0) - r(V_L) \leq W(V_0, V_L)$$
$$r(V_0) - r(V_3) \leq 2$$
$$r(V_0) - r(V_1) \leq 2$$
$$r(V_0) - r(V_2) \leq 0$$



W

	V_0	V_1	V_2	V_3
V_0	0	2	2	2
V_1	0	0	0	0
V_2	0	2	0	0
V_3	0	2	2	0

D

	V_0	V_1	V_2	V_3
V_0	0	3	6	13
V_1	13	3	6	13
V_2	10	13	3	10
V_3	10	13	13	7

Feasibility Const.

$V_0 - V_1 < 2$

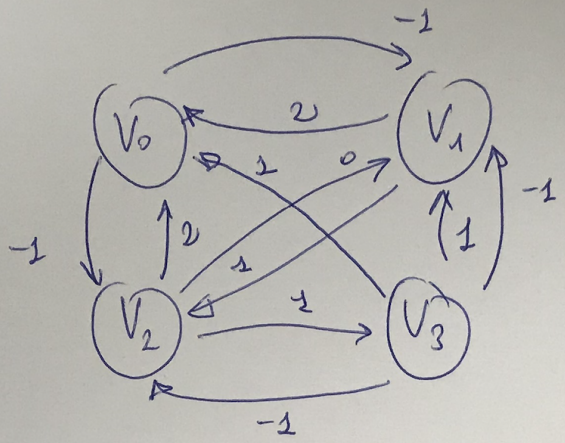
$V_0 - V_2 < 2$

$V_1 - V_2 < 0$

Critical path Const

see slayt.

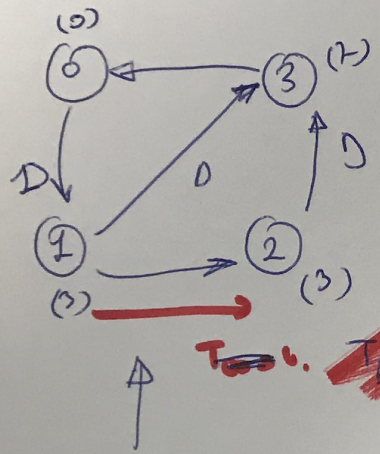
$T_{clk} < 7$



MATLAB

$$W = \begin{bmatrix} Inf & -1 & -1 & \infty \\ 2 & \infty & 1 & 1 \\ 2 & 0 & \infty & 1 \\ 1 & -1 & -1 & \infty \end{bmatrix}$$

$$r = \begin{bmatrix} 1 & -1 & -1 & 0 & \infty \\ 2 & 0 & 0 & 1 & \infty \\ 2 & 0 & 0 & 1 & \infty \\ 1 & -1 & -1 & 0 & \infty \\ 0 & -1 & -1 & 0 & 0 \end{bmatrix}$$



$T_{clk} = \left\{ \frac{13}{2}, \frac{10}{2} \right\}$

$r(0) = 0$

$r(1) = -1$

$r(2) = -1$

$r(3) = 0$

$w_r(0 \to 1) = 2 + r(1) - r(0) = 1$

$w_r(1 \to 3) = 0 + r(3) - r(1) = 1$

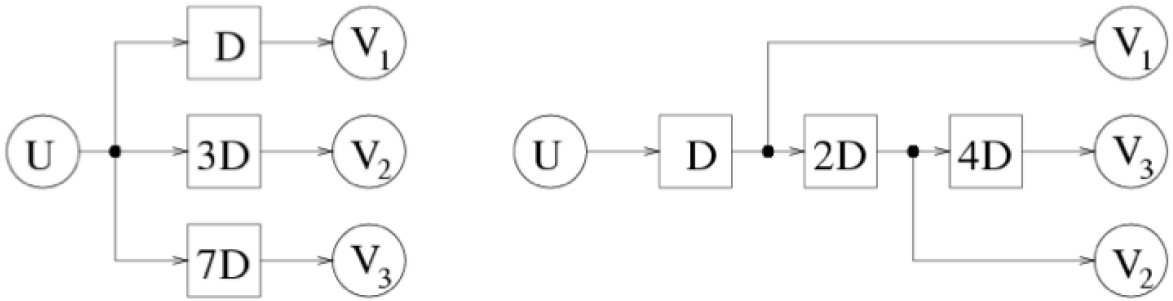
$w_r(1 \to 2) = 0 + r(2) - r(1) = 0$

$w_r(2 \to 3) = 0 + r(3) - r(2) = 1$

$w_r(3 \to 0) = 0 + r(0) - r(3) = 0$

Retiming for Register Minimization

When a node has multiple fan-out with different number of delays, the registers can be shared so that only the branch with max. number of delays will be needed



$$\max_{V \xrightarrow{e} ?} \{1, 3, 7\}$$

Retiming for Register Minimization

The number of registers required to implement the output edges of the node V in the retimed graph is:

$$R_V = \max_{V \xrightarrow{e} ?} \{w_r(e)\}$$

The total register cost in the retimed circuit is

$$COST = \sum R_V$$

Minimize $COST$ subject to

- fanout constraint:

$$R_V \geq w_r(e),\ \forall V\ \&\ \forall e\ V \xrightarrow{e} ?$$

- feasibility constraint:

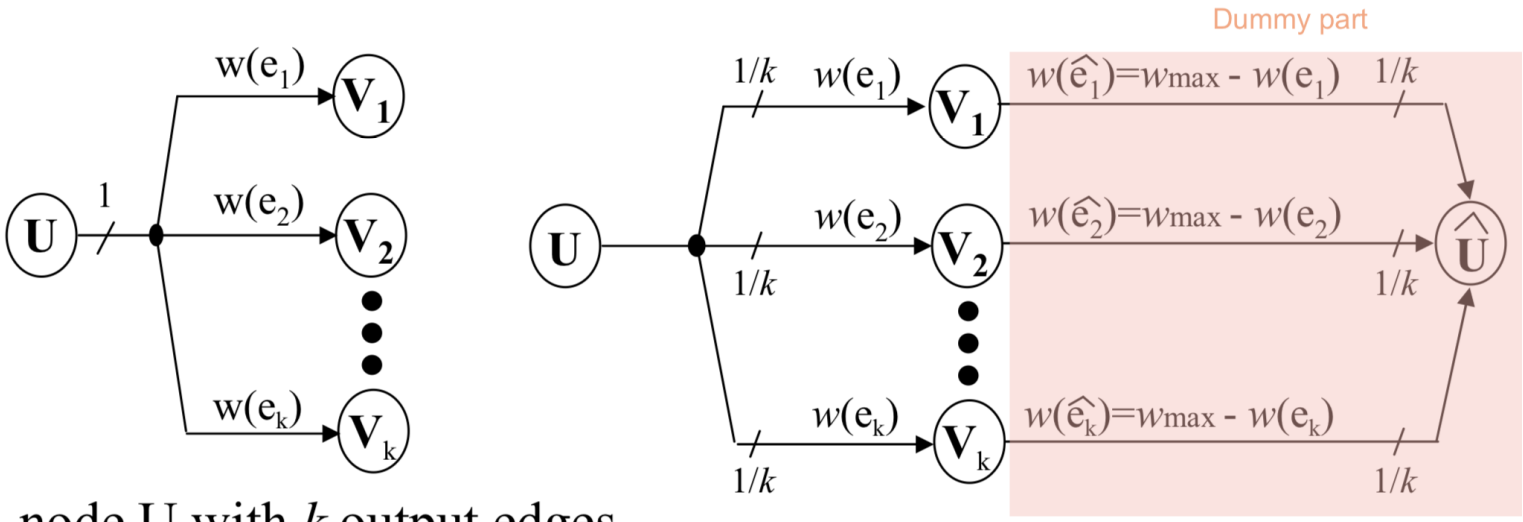
$$r(U) - r(V) \leq w(e),\ \forall U \xrightarrow{e} V$$

- clock period constraint:

$$r(U) - r(V) \leq W(U,V) - 1,\ \forall U \& V\ suchth.\ D(U,V) > c$$

Retiming for Register Minimization

It can not be solved using linear programming (LP) techniques for some solutions may not be integer. Thus, a "gadget" is used to represent nodes with multiple output edges to solve the above problem.



A node U with k output edges

$$w_{max} = \max_{1 \le i \le k} \{w(e_i)\}$$

Dummy edges introduced so the retiming for register min. problem can be modelled as LP.

Retiming for Register Minimization

The breadth of each of the edges (e_i and $\hat{e}_i$) is $\beta = 1/k$ (fanout node with k output edge). Minimize

$$COST = \sum_e \beta(e) w_r(e) = \sum_V r(V) \left(\sum_{? \xrightarrow{e} V} \beta(e) - \sum_{V \xrightarrow{e} ?} \beta(e) \right)$$

subject to

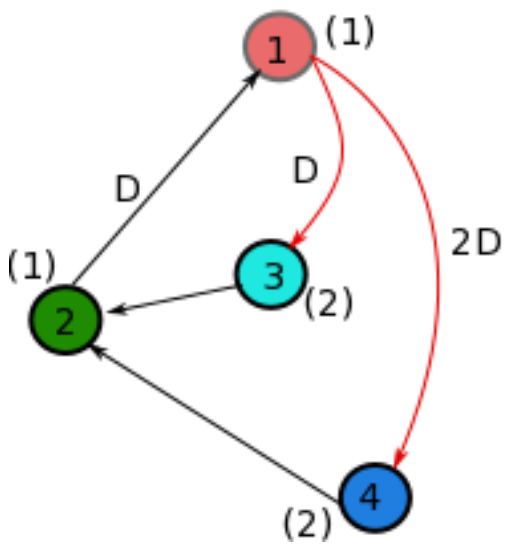
- feasibility constraint:

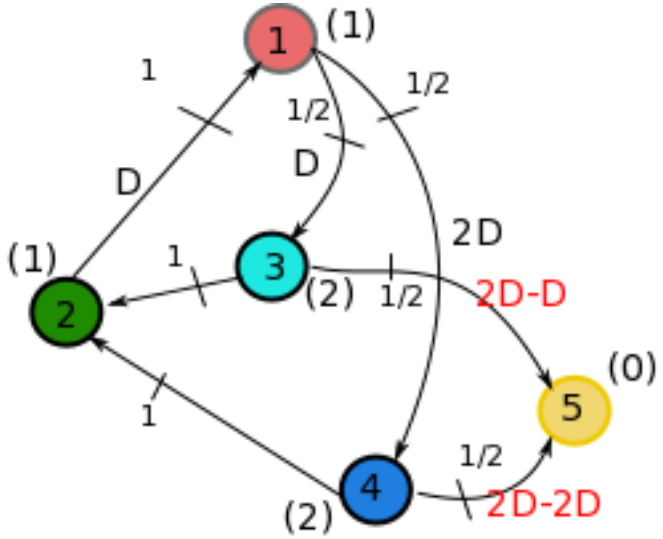
$$r(U) - r(V) \leq w(e), \ \forall U \xrightarrow{e} V$$

- clock period constraint:

$$r(U) - r(V) \leq W(U,V) - 1, \ \forall U \& V \, s.t. \, D(U,V) > c$$

Retiming for Register Minimization





$$COST = \sum_{V} r(V) \left(\sum_{? \xrightarrow{e} V} \beta(e) - \sum_{V \xrightarrow{e} ?} \beta(e) \right)$$

$$\min COST =$$
$$r(1)(1-1) + r(2)(2-1) + r(3)(1/2-3/2) + r(4)(1/2-3/2) + r(5)(1-0)$$

	W					D				
	V_1	V_2	V_3	V_4	V_5	V_1	V_2	V_3	V_4	V_5
V_1	0	1	1	2	2	1	4	3	3	3
V_2	1	0	2	3	3	2	1	4	4	4
V_3	1	0	0	3	1	4	3	2	6	2
V_4	1	0	2	0	0	4	3	6	2	2
V_5	-	-	-	-	-	-	-	-	-	-

subject to

- feasibility constraint:$c = 3$
  $r(1) - r(3) \leq 1, r(1) - r(4) \leq 2, r(2) - r(1) \leq 1, r(3) - r(2) \leq 0, r(3) - r(5) \leq 1, r(4) - r(2) \leq 0, r(4) - r(5) \leq 0$
- clock period constraint:
  $r(1) - r(2) \leq 0, r(1) - r(5) \leq 1, r(2) - r(5) \leq 2, r(3) - r(4) \leq 2, r(4) - r(3) \leq 1, r(1) - r(3) \leq 0, r(2) - r(3) \leq 1, r(3) - r(1) \leq 0, r(4) - r(1) \leq 0, r(1) - r(4) \leq 1, r(2) - r(4) \leq 2, r(3) - r(2) \leq -1, r(4) - r(2) \leq -1$

Retiming Technique for Clock Period Minimization using Shortest Path Algorithm