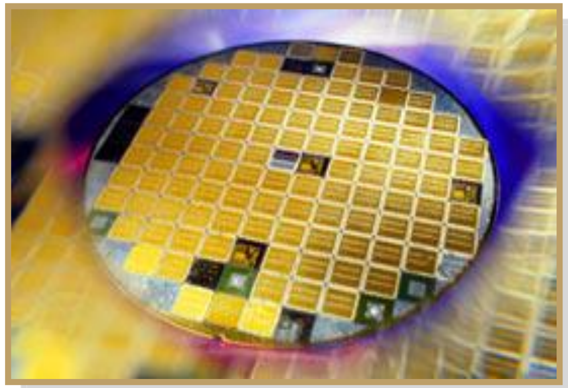
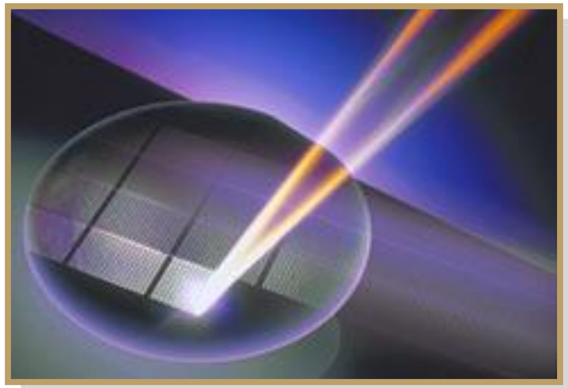
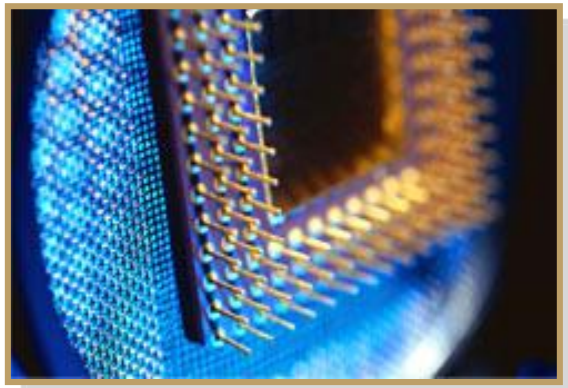
《VLSI数字通信原理与设计》课程

主讲人 贺光辉

第五章：展开







回顾

重定时技术

- 在保持系统功能不变的前提下，改变系统延时数目和分布的方法
- 可以不改变电路结构而提高频率，缩短设计周期

重定时方法

- 割集重定时：
  - 在一个方向的边上增加延时，在另外方向的边上减少同样的延时
  - k倍降速（k-slow）技术

重定时的重要性质

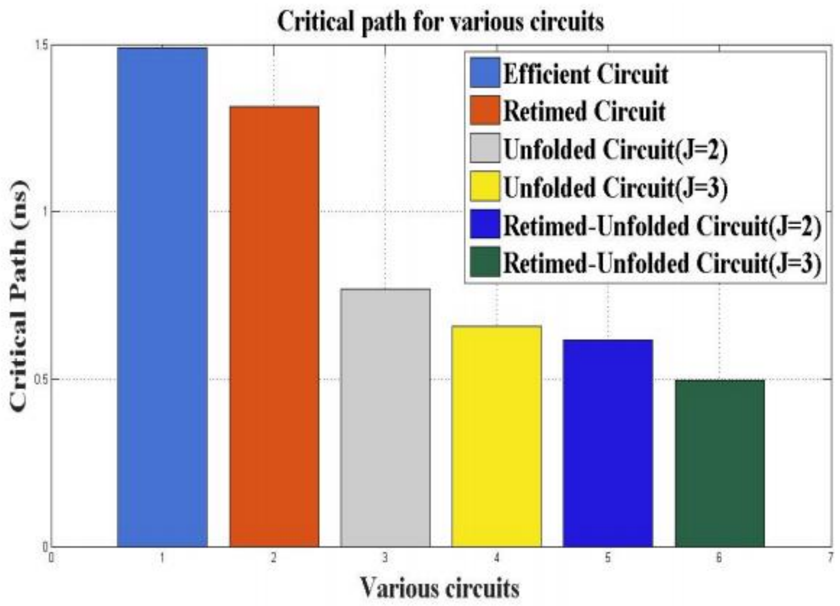
- 不改变环路中的总延迟数
- 不改变DFG的迭代边界T_∞

提升系统吞吐率

- 流水线
- 并行处理
- 重定时

展开

进一步提升系统吞吐率?



Critical path for various circuits [1]

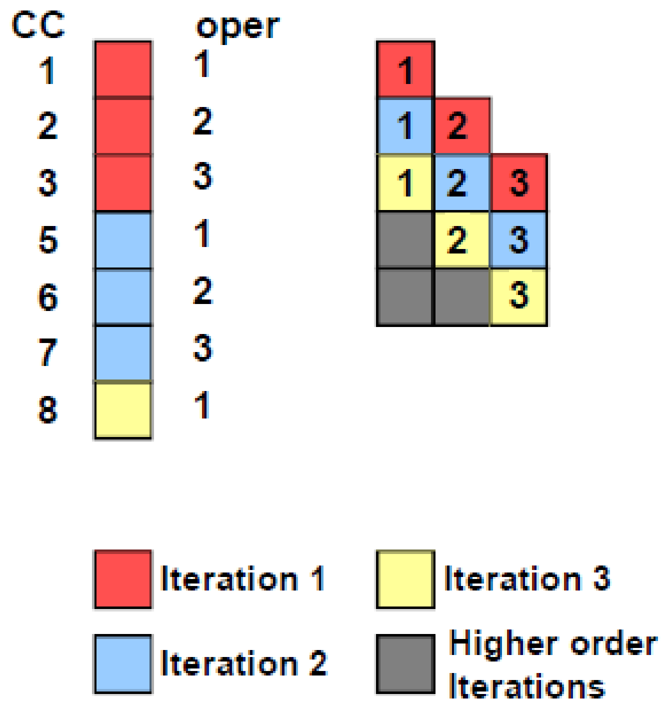
[1] Kumar, Kagitha Naga Lakshmana, Pathipati Srihari, Gnane Swarnadh Satapathi, and Gollakota Venkata Krishna Sharma. "A high speed complementary pulse compressor and its implementation of FPGA." In *2017 IEEE Radar Conference (RadarConf)*, pp. 1379-1382. IEEE, 2017.

目录

01. 展开的基本概念 —— 定义

展开(Unfolding)：

- 展开会创建一个包含多个迭代的程序
- 展开是实现并行处理的结构化方式
- Unfolding = Loop unrolling
  - Assembly programming
  - Compiler theory



应用

- 缩短样本周期
- 并行处理
  - Bit-serial and Digital-serial

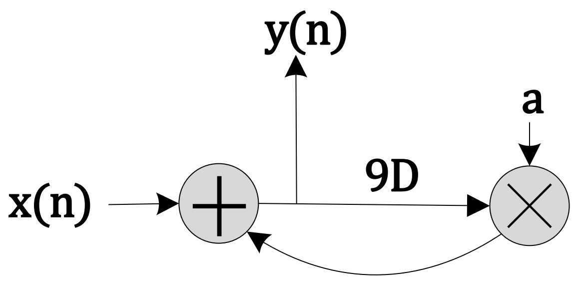
01. 展开的基本概念 —— 示例1

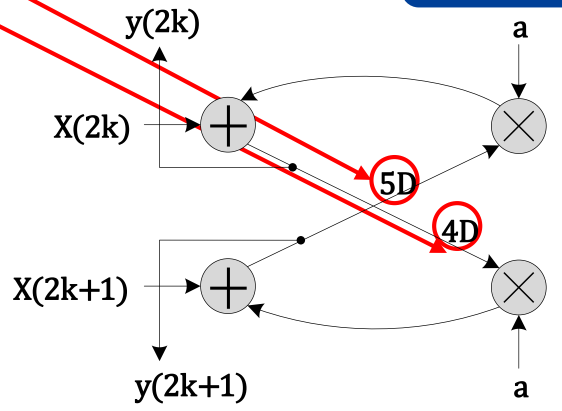
- **展开(Unfolding)：是一种转换技术，它产生一个新的程序来描述原有程序的多次迭代，J称为展开因子，表示迭代次（阶）数**

- **例1：对DSP程序 $y(n) = ay(n-9) + x(n)$ 进行2阶展开**

$$y(2k) = ay(2k-9) + x(2k) = ay(2(k-5)+1) + x(2k)$$
$$y(2k+1) = ay(2k-8) + x(2k+1) = ay(2(k-4)+0) + x(2k+1)$$

二阶展开 (2-unfolding)





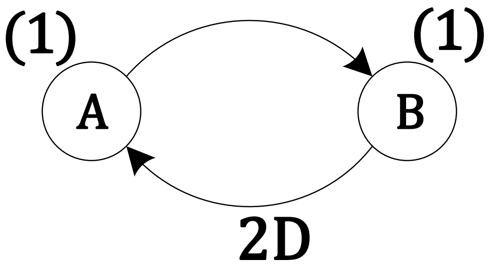
a) **描述$y(n) = ay(n-9) + x(n), n = 0$到∞的DSP原始程序**

b) **描述$y(2k) = ay(2k-9) + x(2k), n = 0$ 和 $y(2k+1) = ay(2k-8) + x(2k+1), n = 0$到∞的二阶展开DSP程序**

在J阶展开系统中，每个延迟元件是J倍降速(J-slow)的

- 如果输入到一个延迟单元的信号是 $x(kJ+m)$，则该延迟单元的输出是

$$x\big((k-1)J+m\big)=x(kJ+m-J)$$



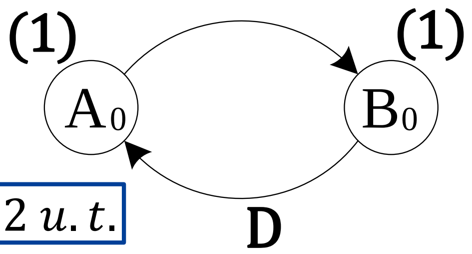
$$A_0 \rightarrow B_0 \Rightarrow A_2 \rightarrow B_2 \Rightarrow A_4 \rightarrow B_4 \Rightarrow \cdots$$
$$A_1 \rightarrow B_1 \Rightarrow A_3 \rightarrow B_3 \Rightarrow A_5 \rightarrow B_5 \Rightarrow \cdots$$

2个节点和两条边

$$T_\infty=(1+1)/2=1\,u.t.$$

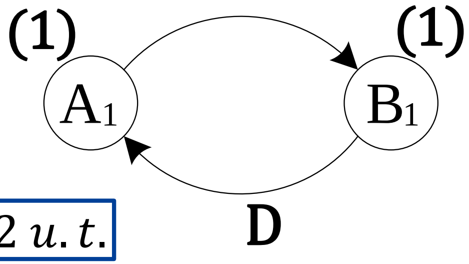
可见：展开 ≡ 并行处理

0,2,4,......



$T_\infty = 2\,u.t.$

1,3,5,......



$T_\infty = 2\,u.t.$

二阶展开

01. 展开的基本概念 ——应用

发掘算法潜在的并发性

- 采用并行处理来降低迭代（采样）周期，（对环路来说）向迭代边界T_∞逼近

得到位/字级并行架构

- 位串行→位并行或字串行
- 字串行→字并行

展开 = 环路展开（Loop Unrolling）

- 应用于汇编编程、编译理论

01. 展开的基本概念 —— 展开方法

符号

- $\lfloor x \rfloor$：表示对x向下取整，即取小于或等于x的最大整数
- $\lceil x \rceil$：表示对x向上取整，即取大于或等于x的最小整数
- $a\%b$(或$a \bmod b$）：表示a除以b的余数，其中a和b是整数

J阶展开DFG的节点与边

- 节点U：有J个具有相同功能的节点$U_i(i = 0, 1, \ldots, J-1)$
- 边：有J条相应的边
  - 即：J阶展开后的DFG总是包含了相当于原始DFG的J倍数量的节点和边

构建一个J阶展开DFG

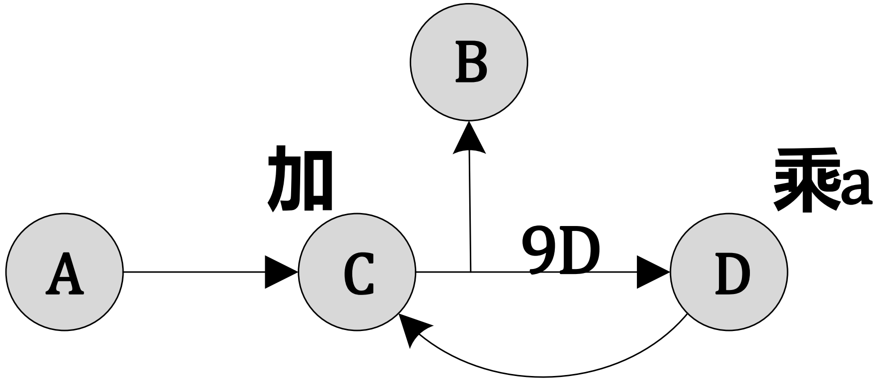
- 对原始DFG中的每个节点U，画J个节点$U_0, U_1, \ldots, U_{J-1}$
- 对在原始DFG中的每个延迟为w的边$U \to V$，画延迟为

  $w_{unf} = \lfloor (i+w)/J \rfloor$的J个边$U_i \to V_{(i+w)\%J}(i = 0, 1, \ldots, J-1)$

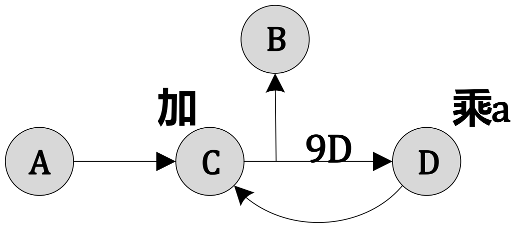
例2： $y(n) = ay(n-9) + x(n)$

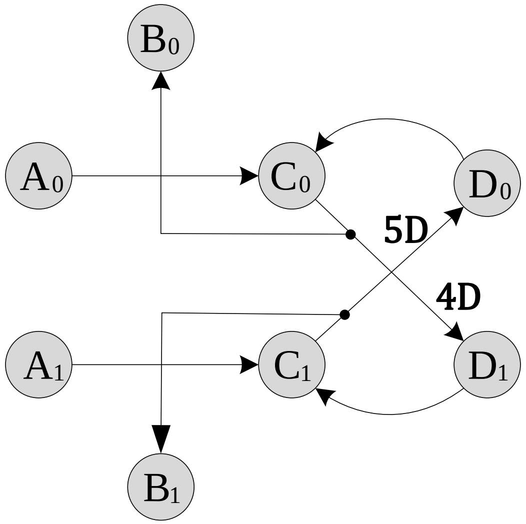
原始图

- 节点A和B：输入和输出
- 节点C和D：相加和乘 a



例2： $y(n) = ay(n-9) + x(n)$



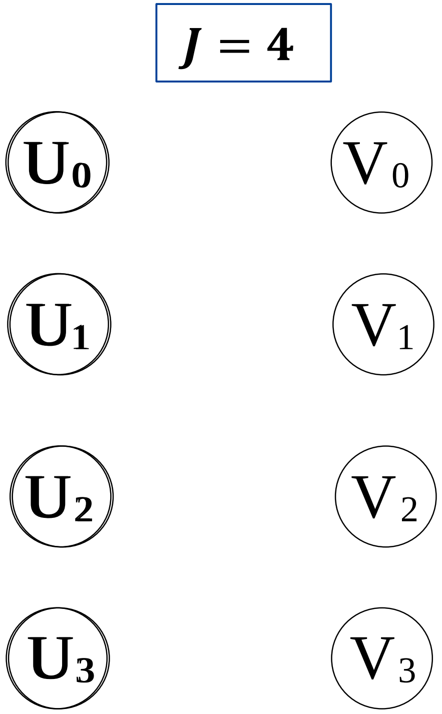


二阶展开图—节点&边

- 节点A_0： 输入，x(2k+0) 节点A_1： 输入，x(2k+1)
- 节点B_0： 输出，y(2k+0) 节点B_1： 输出，y(2k+1)
- 无延迟边A →C、 C→ B、 D→C都展开为连接相应节点的两条边
- 延迟9D(w=9)的边$C \rightarrow D$展开为两条边：
  - 延迟4D($w_{unf} = \left\lfloor \frac{0+9}{2} \right\rfloor = 4$)的边$C_0 \rightarrow D_1\left(D_{(0+9)\%2=1}\right)$
  - 延迟5D($w_{unf} = \left\lfloor \frac{1+9}{2} \right\rfloor = 5$)的边$C_1 \rightarrow D_0\left(D_{(1+9)\%2=0}\right)$
  - 当w<J时，展开原始DFG中延迟为w的边，相当于在J阶展开DFG中生成了J-w条无延迟的边和w条延迟为1的边

例3：$J = 4$

- 对于原始DFG中的每个节点U，绘制J个节点$U_0, U_1, U_2, \dots U_{J-1}$

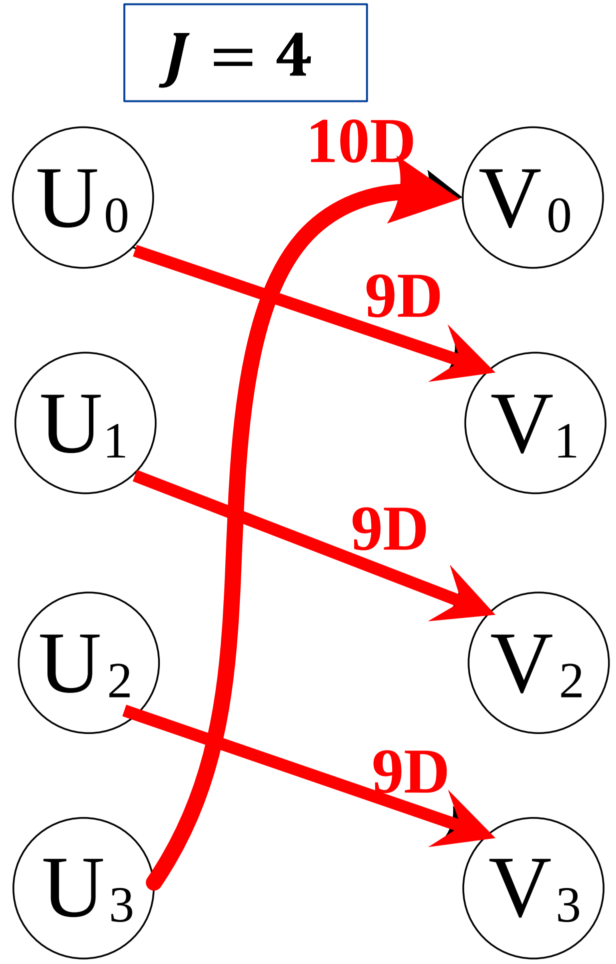


- 对于原始DFG中带有w延迟的每条边 $U \to V$, 用$\lfloor (i+w)/J \rfloor$ 延迟绘制边$U_i \to V_{(i+w)\%J}$ ，$i = 0, 1, \dots, J-1$

$$\left\lfloor \frac{(i+w)}{J} \right\rfloor = \left\lfloor \frac{(i+37)}{4} \right\rfloor = \begin{cases} 9\ , i = 0,1,2 \\ 10\ , i = 3 \end{cases}$$

例3：$J = 4$

- $U_0 \rightarrow V_{(0+37)\%4=1}$ with $\left\lfloor\frac{0+37}{4}\right\rfloor = 9$ delays
- $U_1 \rightarrow V_{(1+37)\%4=2}$ with $\left\lfloor\frac{1+37}{4}\right\rfloor = 9$ delays
- $U_2 \rightarrow V_{(2+37)\%4=3}$ with $\left\lfloor\frac{2+37}{4}\right\rfloor = 9$ delays
- $U_3 \rightarrow V_{(3+37)\%4=0}$ with $\left\lfloor\frac{3+37}{4}\right\rfloor = 10$ delays



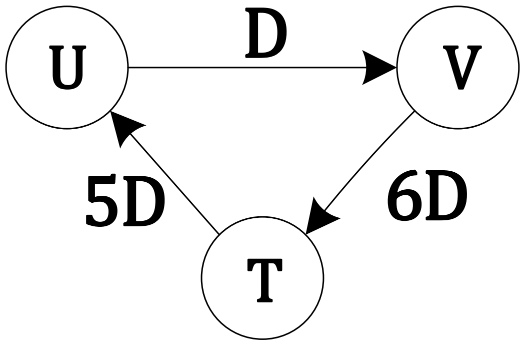
例4：$J=3$

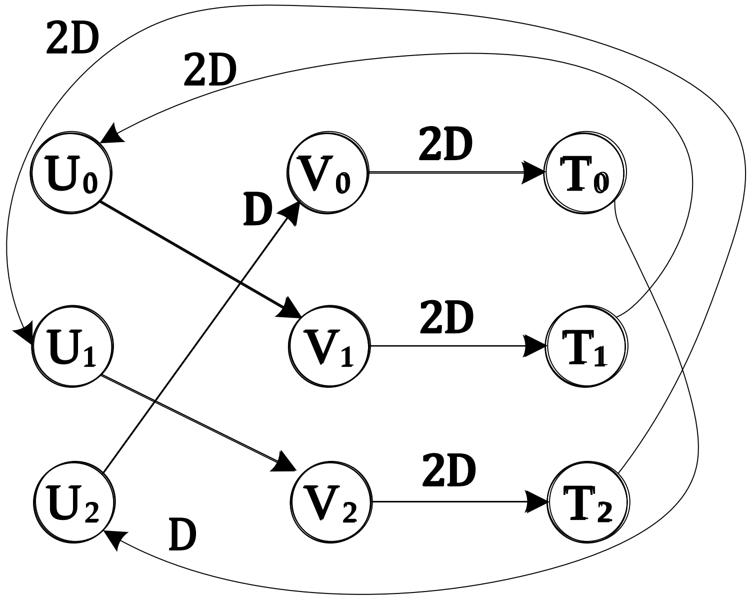
- $w_{unf}=\lfloor (i+w)/J \rfloor$, $\quad U_i \to V_{(i+w)\%J}, i=0,1,2$

$U_0 \to V_{(0+1)\%3=1}$ [0D]
$U_1 \to V_{(1+1)\%3=2}$ [0D]
$U_2 \to V_{(2+1)\%3=0}$ [D]

$T_{(1+6)\%3=1}$ [2D]
$T_{(2+6)\%3=2}$ [2D]
$T_{(0+6)\%3=0}$ [2D]

$U_{(1+5)\%3=0}$ [2D]
$U_{(2+5)\%3=1}$ [2D]
$U_{(0+5)\%3=2}$ [D]





目录

02. 展开的性质

展开保留原DFG中各边的优先约束

- J 阶展开DFG中延迟为$\left\lfloor \frac{i+w}{J} \right\rfloor (i = 0, 1, \dots, J-1)$ 的 J 条边$U_i \to V_{(i+w)\%J}$对应于原始DFG中延迟为w的边$U \to V$

展开保持原DFG中各边的延迟数

- J 阶展开DFG中的边$U_i \to V_{(i+w)\%J}, i = 0, 1, \dots, J-1$ 的延迟数之和与原始DFG中边$U \to V$的延迟数相同

$$\boxed{\lfloor w/J \rfloor + \lfloor (w+1)/J \rfloor + \cdots + \lfloor (w+J-1)/J \rfloor = w}$$

展开对关键路径的影响

- 原图G中w<J的边在J阶展开G_J中生成J-w条无延迟的边和w条延迟为1的边。

  注意：无延迟边的增加意味着关键路径可能增加，可能导致时钟周期增加、频率降低

- 原图G中$w \geq J$ 的边在J阶展开G_J 中生成J条延迟≥ 1的边，不会生成无延迟边，则不会增加关键路径

02. 展开的性质

展开环路的规则

- 原图G中延迟为w_l的环路l，在J阶展开G_J中有$gcd(w_l,J)$个环路，每个环路包含了$w_l/gcd(w_l,J)$个延迟以及原环路l中每个节点的$J/gcd(w_l,J)$个拷贝

展开环路使迭代边界增加

- 迭代边界为T_∞的原图G的J阶展开G_J的迭代边界为JT_∞

$$\boxed{T'_\infty = \max_l \left\{\frac{(J/\mathrm{gcd}(w_l,J))t_l}{w_l/\mathrm{gcd}(w_l,J)}\right\} = J\max_l\left\{\frac{t_l}{w_l}\right\} = JT_\infty}$$

02. 展开的性质 —— 示例5

例5：一个环的3阶展开

展开前

延迟$w_l = 12$的环

关键路径 $T_{critical} = 1ut$

迭代边界T_∞ =3/12ut

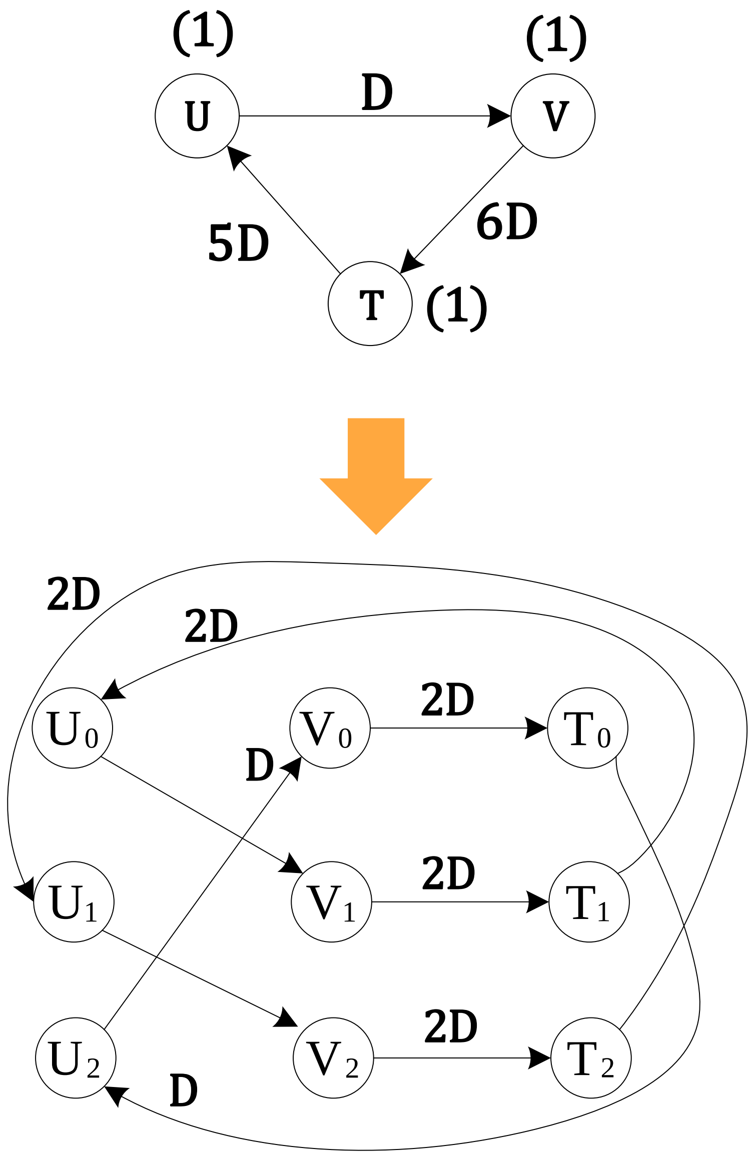
3阶展开图

环数$gcd(12,3) = 3$

环延迟12/3= 4

关键路径$T_{critical} = 2ut$, 增加!

迭代边界3T_∞ =3/4ut



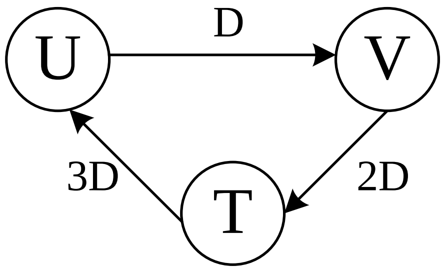
例6：一个环的3阶、4阶展开

展开前

延迟$w_l = 6$的环

关键路径 $T_{critical} = 1ut$

迭代边界T_∞ =3/6ut



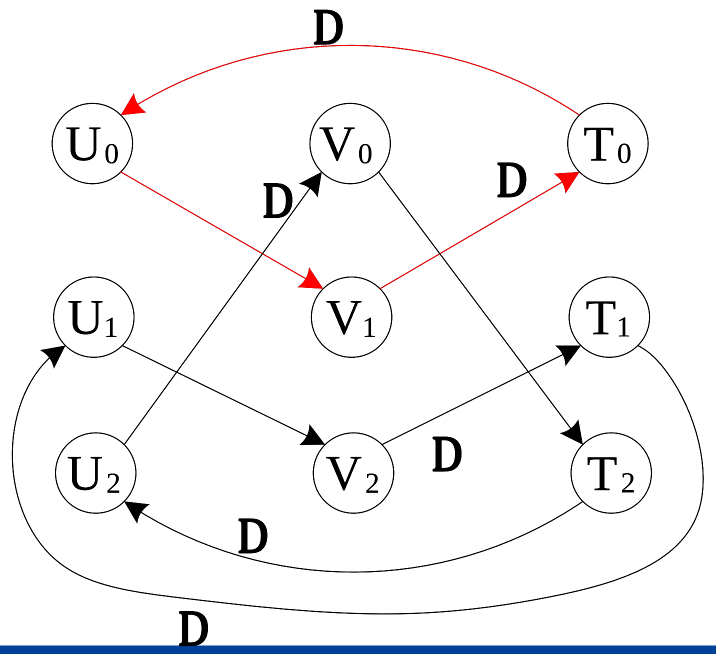
3阶展开图

环数$gcd(6,3) = 3$

环延迟6/3= 2

关键路径$T_{critical} = 2ut$, 增加!

迭代边界$3T_\infty$ =3/2ut



例6：一个环的3阶、4阶展开

展开前

延迟$w_l = 6$的环

关键路径 $T_{critical} = 1ut$

迭代边界T_∞ =3/6ut

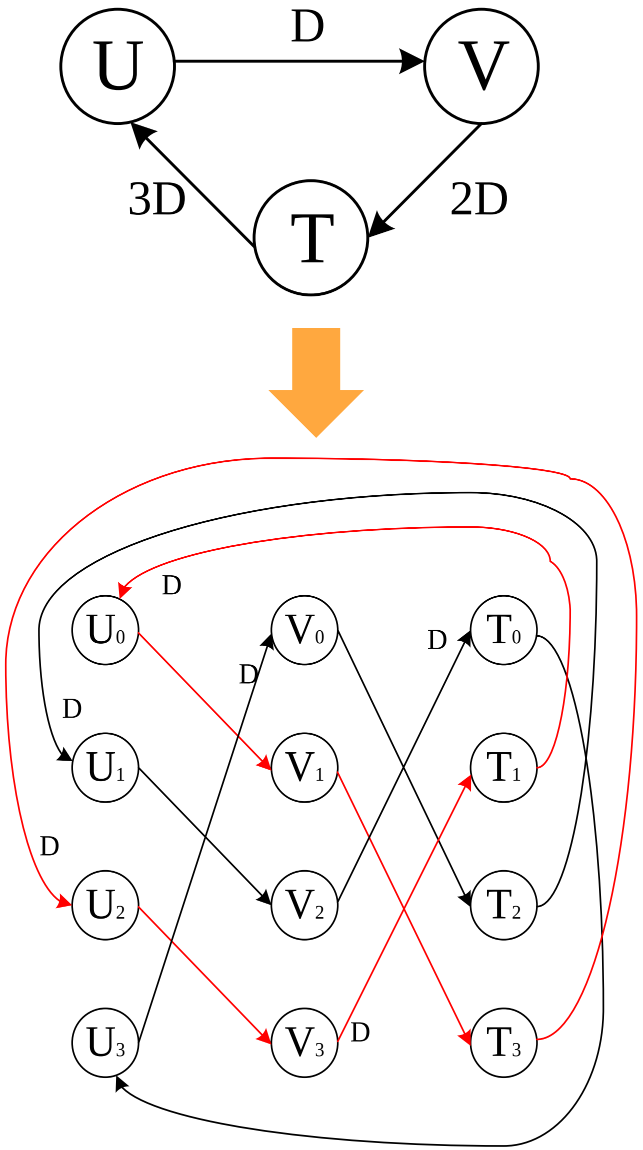
4阶展开图

环数$gcd(6,4) = 2$

环延迟6/2= 3

关键路径$T_{critical} = 3ut$, 增加!

迭代边界4$T_\infty = 6/3 = 2$ut



例7：$y(n) = ay(n-9) + x(n)$

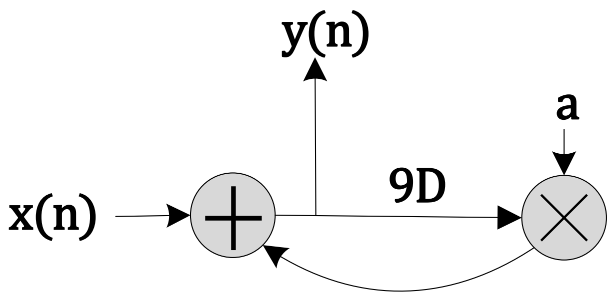
$T_A = 3, T_M = 6$
$T_\infty = 9/9 = 1$

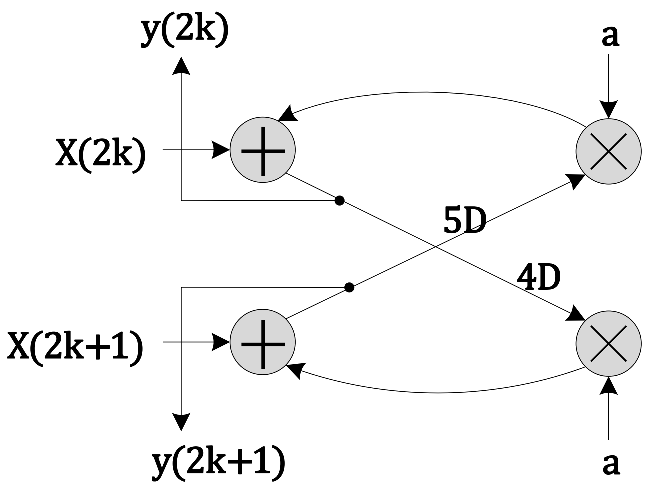
展开前

延迟$w_l = 9$的环

关键路径 $T_{critical} = 9ut$

迭代边界T_∞ = 9ut/9=1ut





$gcd(9,2) = 1$
$T_\infty = 18/9 = 2$

2阶展开图

环数$gcd(9,2) = 1$

环延迟= 9

每节点2个拷贝

关键路径$T_{critical} = 9ut$, 不变!

迭代边界$2T_\infty$ = 2ut

But we process 2 samples

例8：$J = 3$

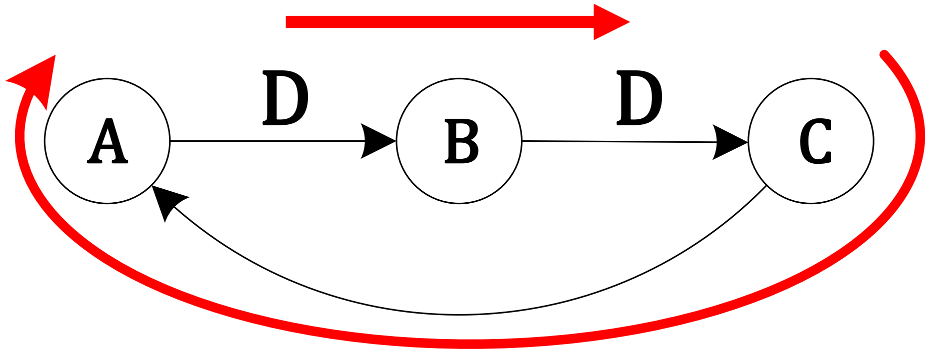
展开前

延迟$w_l = 2$的环

关键路径 $T_{critical} = T_A + T_C$

迭代边界$T_\infty = (T_A + T_B + T_C)/2$

Can lead to increased critical path!



Edge with $w \geq J$ will not create new critical path!

例8：$J = 3$

3阶展开图

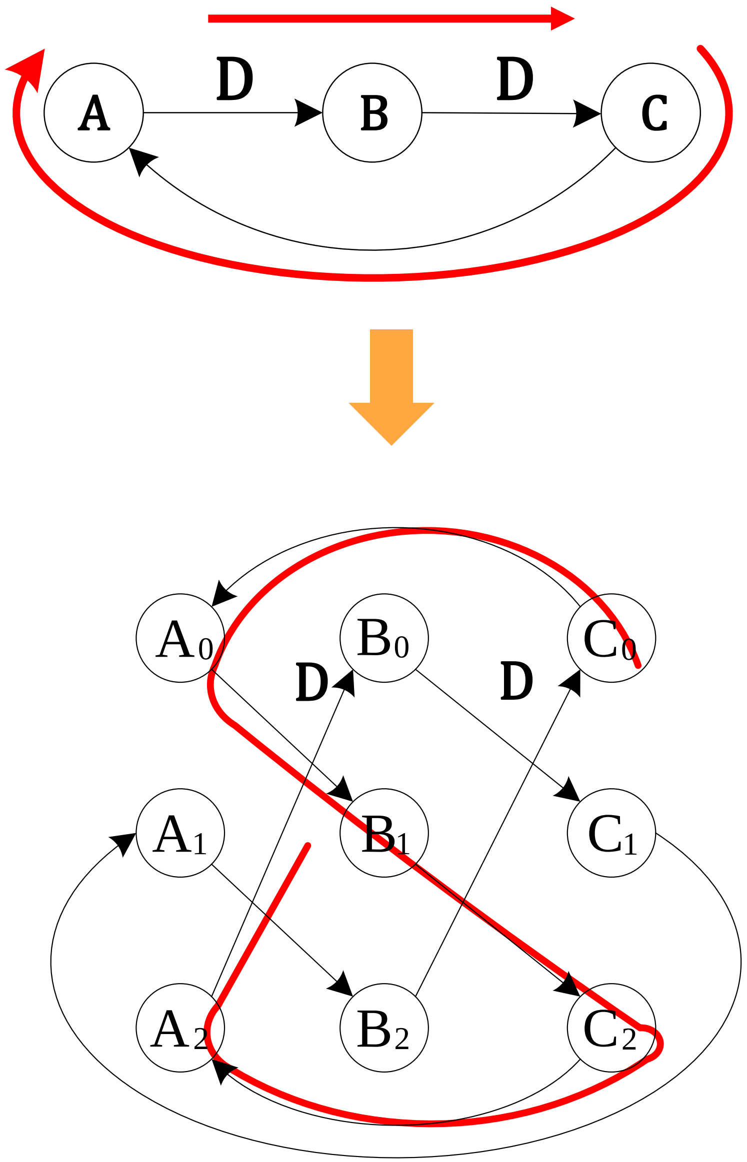
环数$gcd(2,3) = 1$

环延迟$2/1 = 2$

每节点3个拷贝

关键路径$T_{critical} = T_{A2} + T_{C2} + T_{B1} + T_{A0} + T_{C0}$，增加！

迭代边界$3T_{\infty}$



例9：$J = 3$

展开前

延迟$w_l = 12$的环

$T_{critical} = 1ut$

迭代边界T_∞ =3/12ut

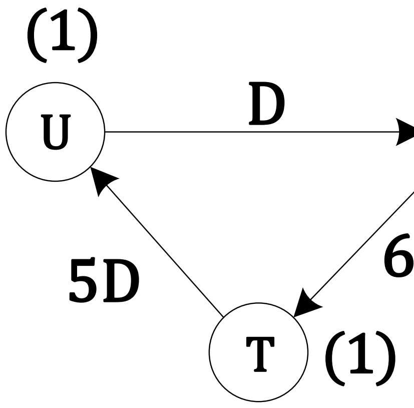
3阶展开图

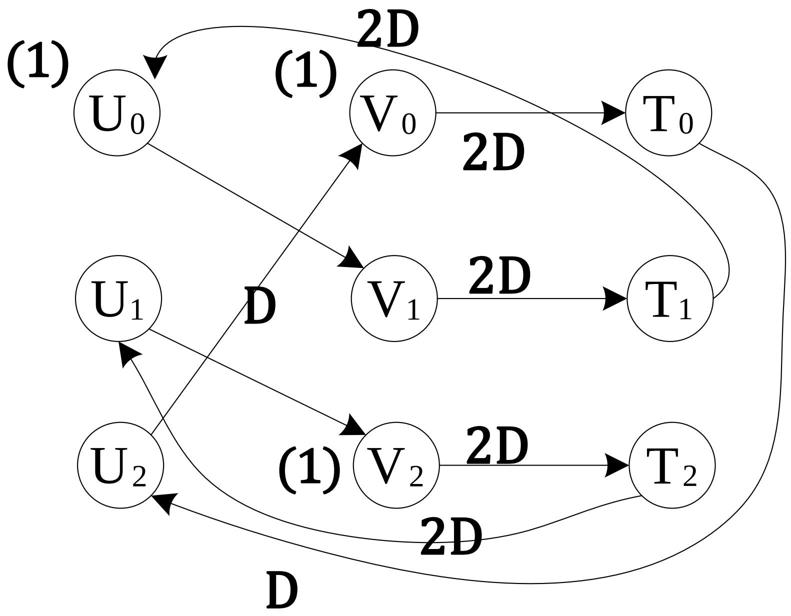
环数$gcd(12,3) = 3$

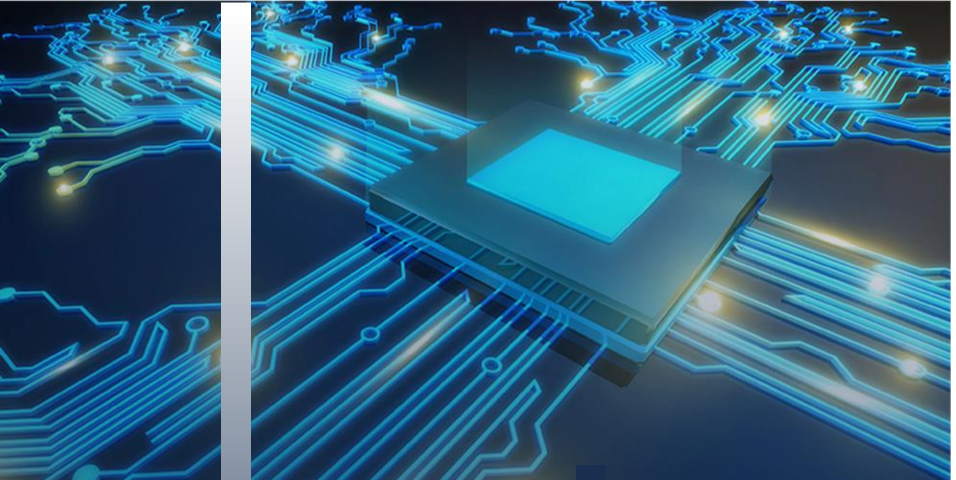
环延迟12/3= 4

关键路径$T_{critical} = 2ut$, 增加!

迭代边界$3T_\infty$ =3/4ut



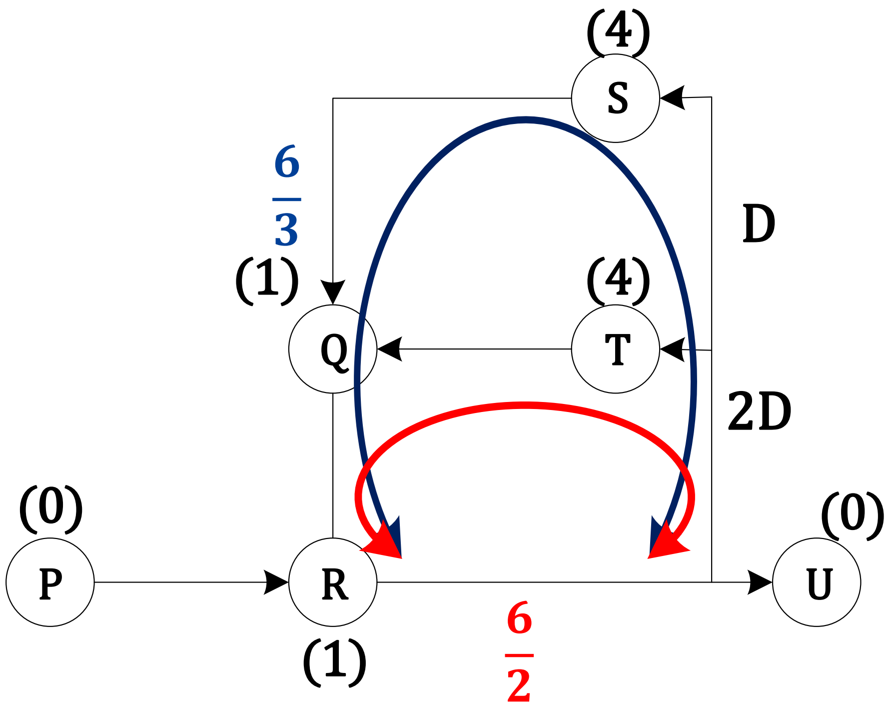




目录

情况1：原始DFG中存在计算时间$T_U > T_\infty$的节点

- **原图：节点S、T的计算时间4ut，$T_{critical} = 6\text{ut}, T_\infty = 3ut$**



因为$T_S = T_T = 4ut$, 则直接重定时不能解决问题，需要展开

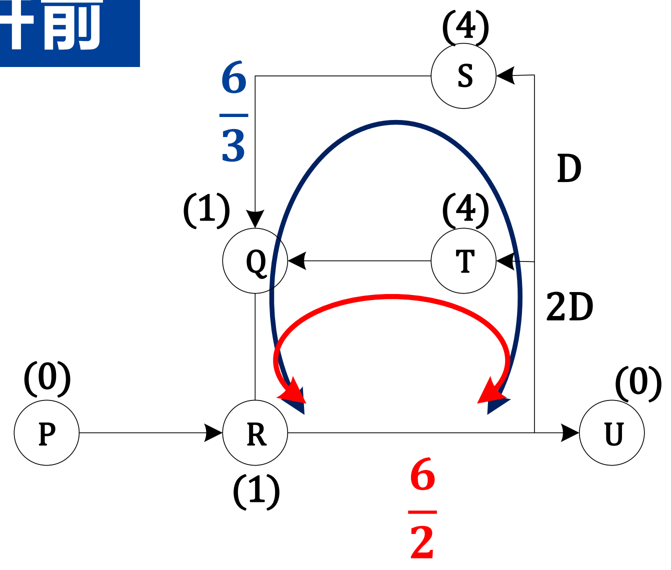
$$T_\infty = \max_{l \in L}\left\{\frac{t_l}{w_l}\right\} = \max_{l \in L}\left\{\frac{6}{3}, \frac{6}{2}\right\} = 3$$

< 4, max node time

情况1：原始DFG中存在计算时间$T_U > T_\infty$的节点

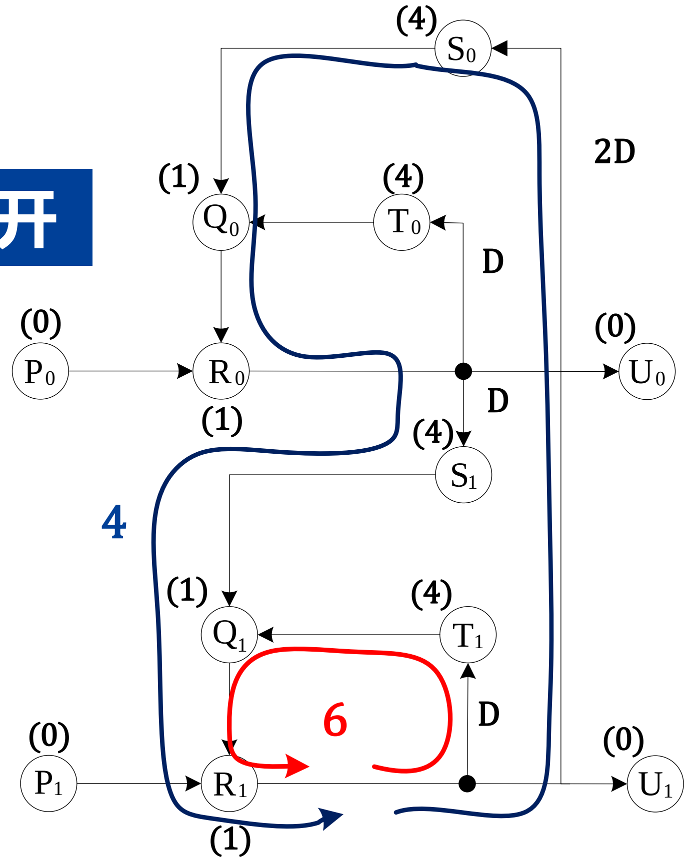
- 原图：节点S、T的计算时间4ut，$T_{critical} = 6\text{ut}, T_\infty = 3ut$
- 2阶展开图：$T_{critical} = 6\text{ut}$, 不变! 迭代边界为$2T_\infty = 6ut$，但是每次迭代处理2个样点，则$T_S$ =3ut，采样速率可以提高

展开前



一般情况下最小需要取$J = \lceil T_U/T_\infty \rceil$

2阶展开



$T_U = 4, T_\infty = 3$

$\left\lceil \frac{4}{3} \right\rceil = 2 - unfolding$

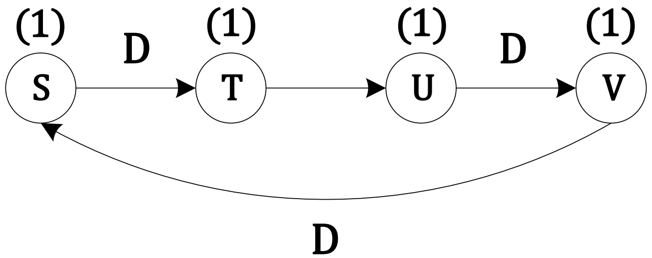
两个样本

3

情况2：迭代边界T_∞不是整数

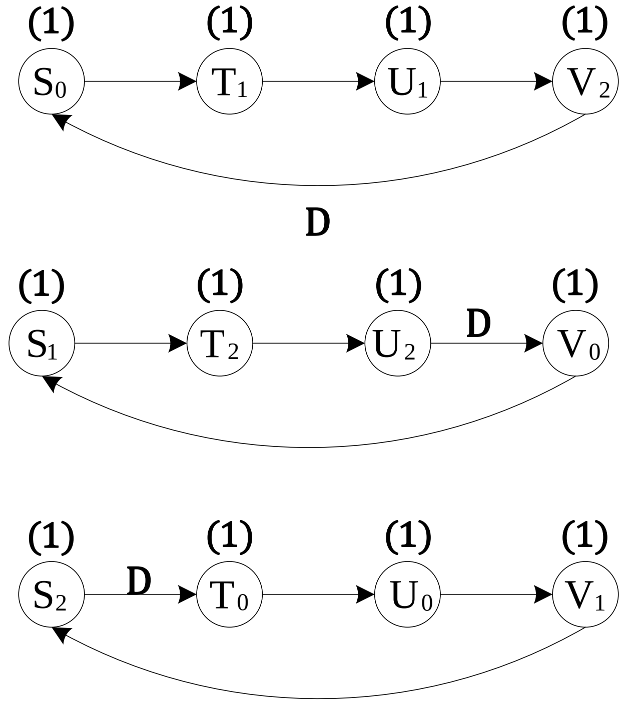
- 原图：$T_l = 4\text{ut}$，$w_l = 3$，$T_\infty = 4/3$，$T_{critical} = 2$，$T_S \geq 2$
- 3阶展开图：$T_\infty = 4$，$T_{critical} = 4$,增加! 每次迭代处理3个样点，则$T_S = 4/3$

一般情况下最小需要取$J = w_l/gcd(T_l, w_l)$



$$T_\infty = \max_{l \in L}\left\{\frac{t_l}{w_l}\right\} = \frac{4}{3}$$

展开前



3阶展开

情况3：最长的节点计算时间大于迭代边界T_∞，同时T_∞也不是整数（是情况1、2的混合）

- 例如：若$T_\infty = 4/3$，且$T_U > 6$，

  满足：

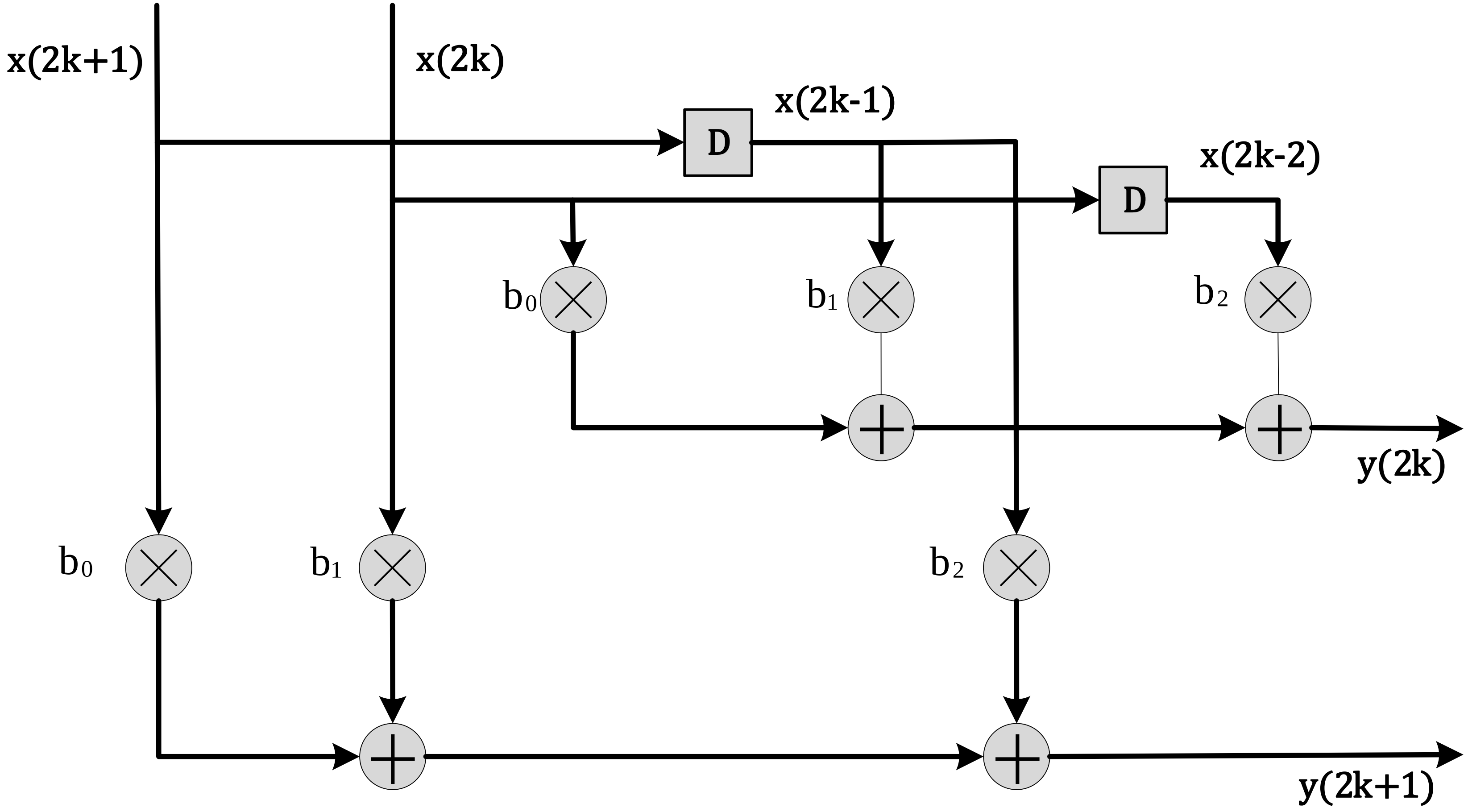
J(4/3)为整数	J(4/3)≥ 6	最小J值

↓

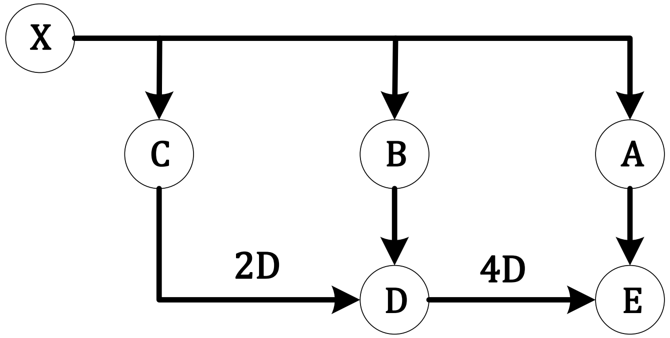
使得迭代周期等于迭代边界的最小展开因子为$J=6$

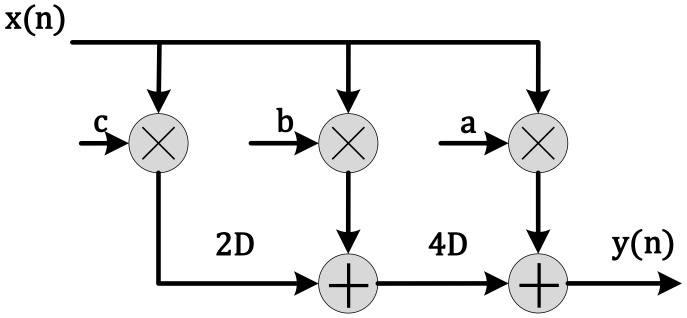
- 使得迭代周期能够等于迭代边界的最小展开因子是能使JT_∞为整数且≥最长节点计算时间的J的最小值

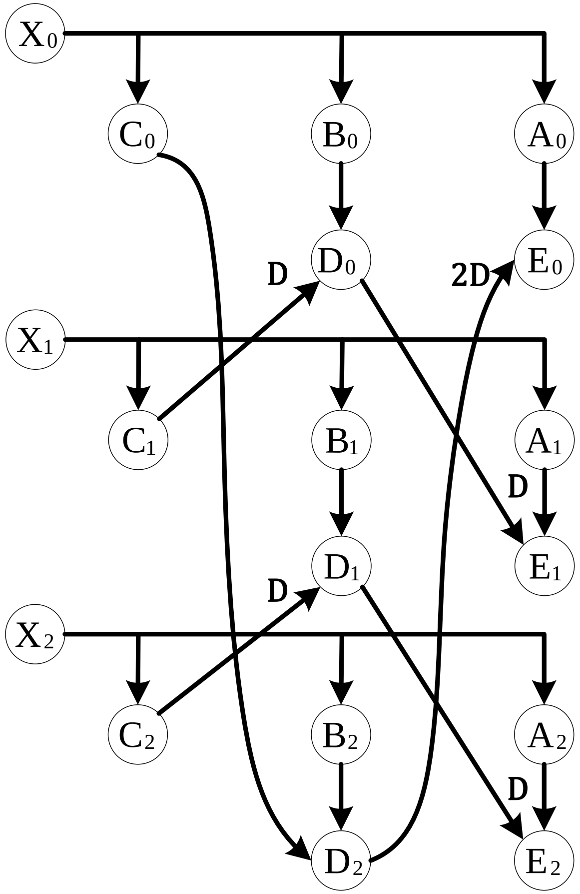
Word level & bit level

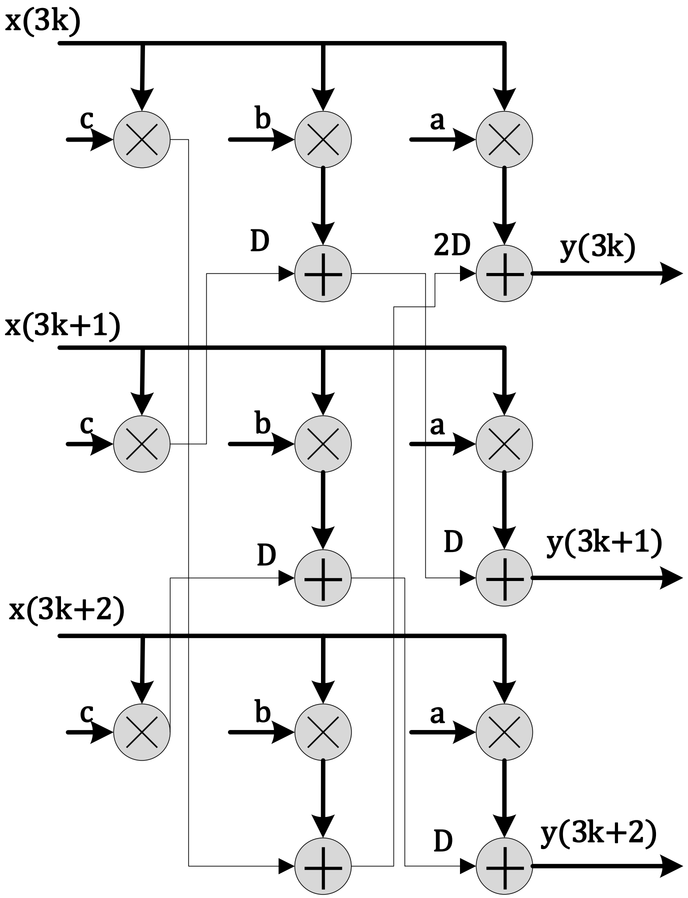


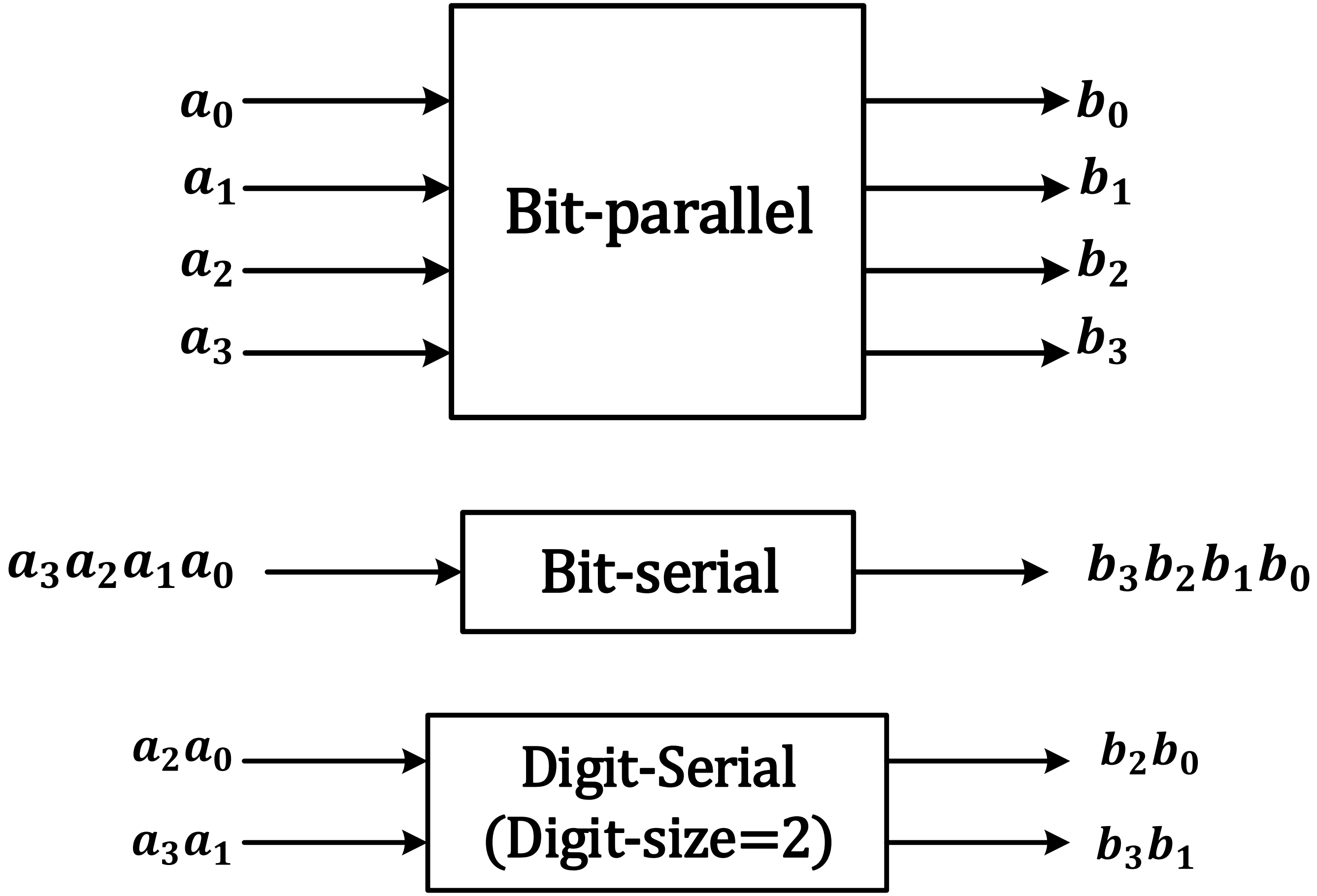
03. 展开的应用 —— Word-Level Parallel Processing



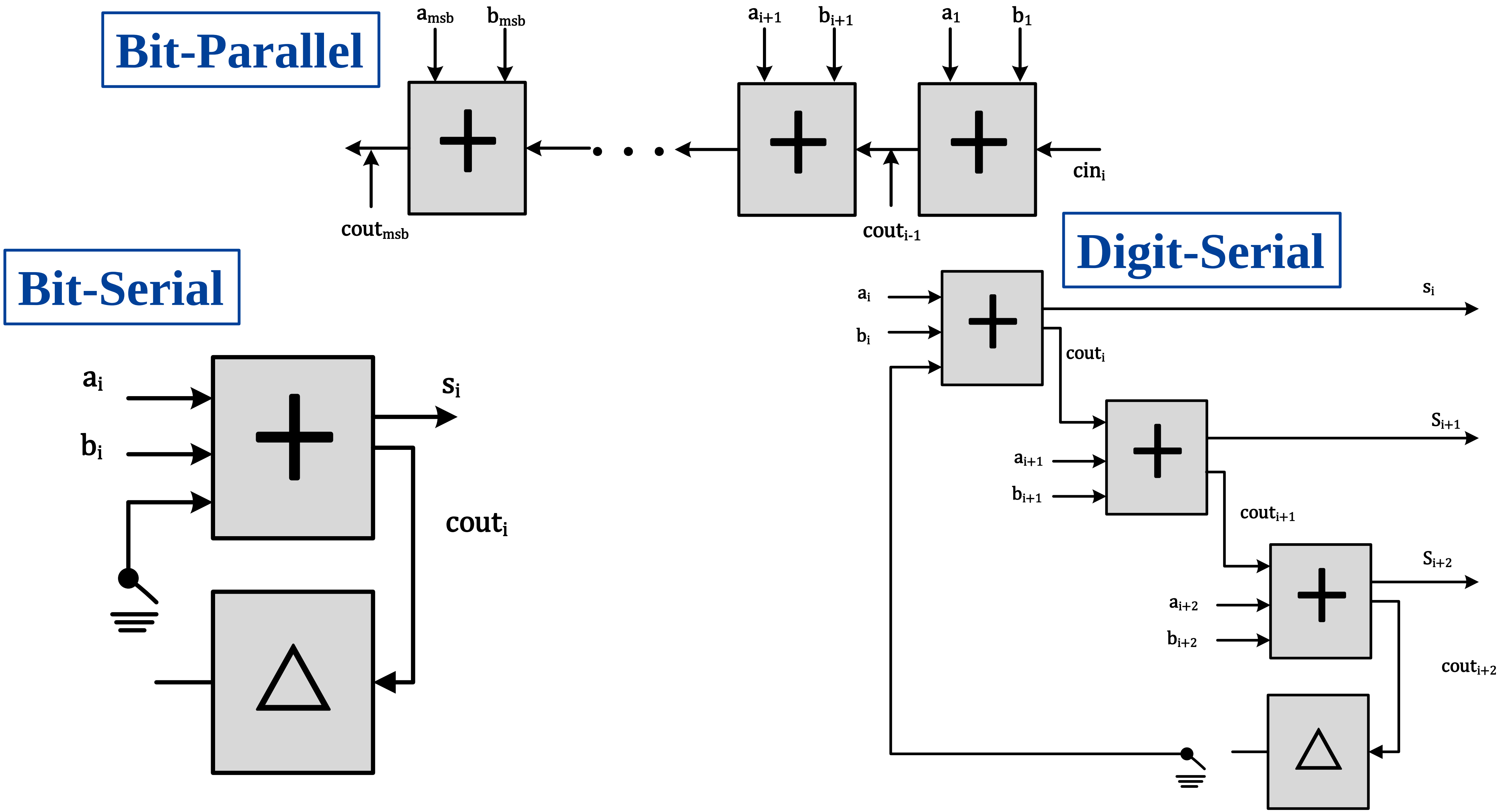







a_0
a_1
a_2
a_3
Bit-parallel
b_0
b_1
b_2
b_3
$a_3a_2a_1a_0$
Bit-serial
$b_3b_2b_1b_0$
a_2a_0
a_3a_1
Digit-Serial
(Digit-size=2)
b_2b_0
b_3b_1

03. 展开的应用



Bit-Parallel
a_{msb}
b_{msb}
a_{i+1}
b_{i+1}
a_1
b_1
cin_i
$cout_{msb}$
$cout_{i-1}$
Bit-Serial
a_i
b_i
s_i
$cout_i$
Digit-Serial
a_i
b_i
s_i
$cout_i$
S_{i+1}
a_{i+1}
b_{i+1}
$cout_{i+1}$
S_{i+2}
a_{i+2}
b_{i+2}
$cout_{i+2}$

目录

展开与并行处理

算法

- 节点和边：原图G的J倍数量
- J条边的分配：$i = 0, 1, \dots, J-1$
  - $U_i \rightarrow V_{(i+w)\%J}$
  - 延迟为$w_{unf}(i) = \lfloor (i+w)/J \rfloor$
- 环路：
  - 原图G中延迟为w_l的环路l，在J阶展开G_J中有$gcd(w_l, J)$个环路
  - 每个环路包含了$w_l/gcd(w_l, J)$个延迟

性质

- 保留原图G中各边的优先约束
- 保持原图G中各边的延迟数
- 对关键路径的影响
  - 原图G中$w < J$的边在J阶展开G_J中生成$J - w$条无延迟的边可能增加关键路径

谢谢！