Model of Computation - Kahn Process Network

1. **1**

   - 算法1是不确定的，因为其输入的时序会影响算法的输出。

     证明：

     当 $X1 = ([x1, x2], [\phi, \phi]) X2 = ([x1, x2], [y1, y2])$

     此时$F(X1) = ([x1, x2])$ $F(X2) = ([x1, y1, x2, y2])$

     此时 $\forall i X1_i \subseteq X2_i$ 即$X1_i \subseteq X2_i$ ，但是$F(X1) \not\subseteq F(X2)$ 所以是不确定的

     算法1 是公平的，因为即使输入序列的长度不同，也依旧遵循FCFS的原则。

   - 算法2是确定的，其输入时序不会影响系统的输出，但算法2是不公平的因为两个输入的序列中较长的那个需要等待较短的那个，此时也就是会出现较长的序列饥饿，需要等待较短序列中新的输入出现，如果当我们用完了较短序列，此时就会出现需要等待一个不会出现的数据的情况

   2

   - 公式为

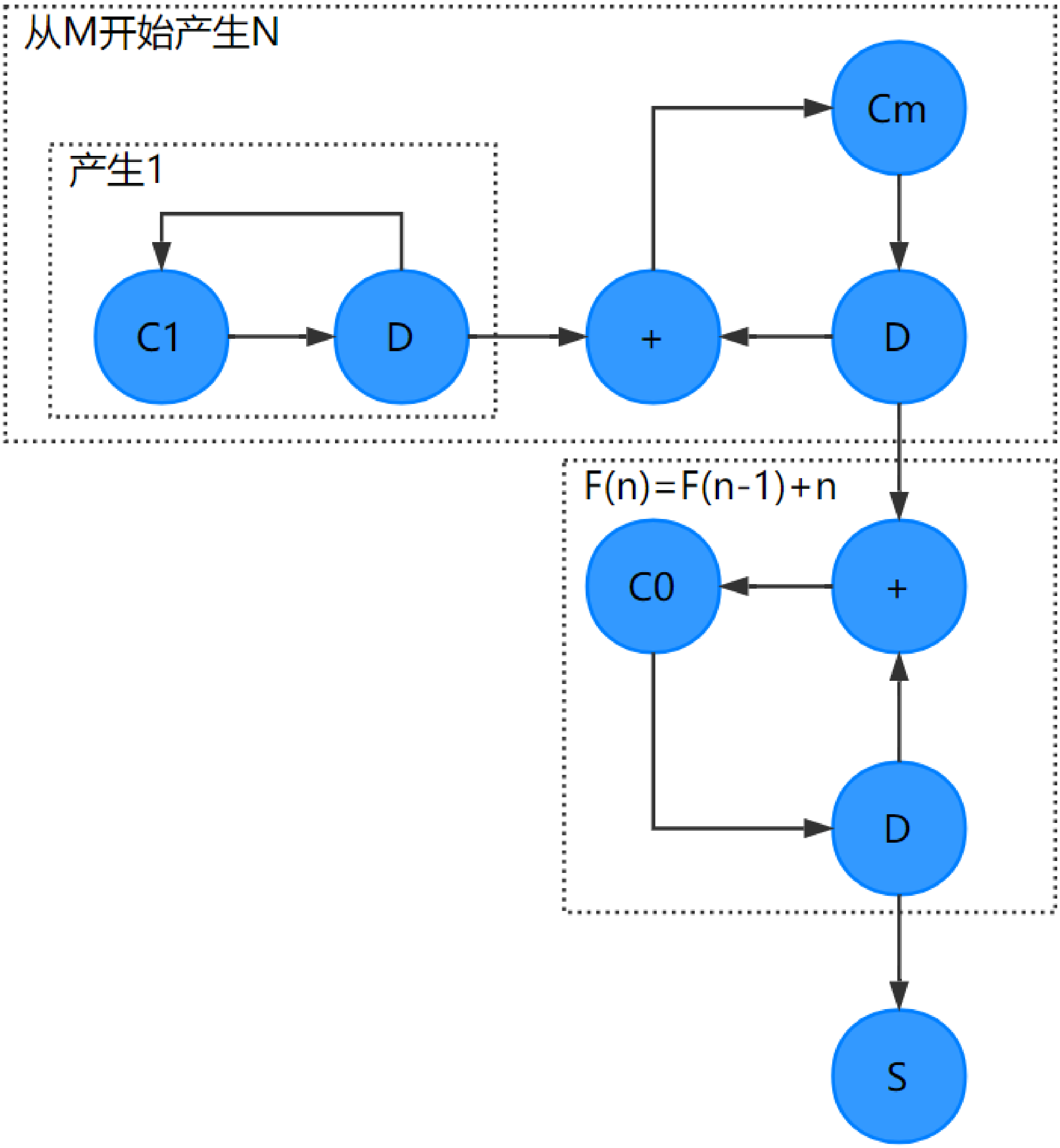
     $$F(n) = \frac{n(n+1)}{2} = \sum_{i=1}^{n} i \qquad (1)$$

     所以可以状转化为

     $F(0) = 0 \quad (2)$

     $F(n) = F(n-1) + n (n \geq 1) \quad (3)$

     Kahn Process Network图如下



其中Ci模块为Constant 一个输入端一个输出端，一开始输出i之后，之后的输入即为输出。

```
out(i);
for(::){
  a = wait(in);
  out(a);
}
```

上述图中m=1 主要是为了使得运行之后能按顺序输出n={0,1,..,2}，开始的时候C0先给出一个0，之后Cm给出1到达加法处，加上0使得C0处输出1。此时的Cm输入的值为2，然后就是可以理解的正常循环了。

D模块为Duplicating a number拷贝数据一个输入拷贝两份输出

```
for(::){
  a = wait(in);
  out1(a);
  out2(a);
}
```

加法模块

```
for(::){
  a = wait(in1);
  b = wait(in2);
  c = a + b;
  out(c);
}
```

S模块Sink process，无限等待数据输入，一旦有输入，就将原数据丢弃。

2. **1**

左图:

$$\begin{cases} a-b=0 \\ b-a=0 \end{cases} \tag{4}$$

拓扑矩阵为:
$$M=\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{5}$$

左图是consistent,因为 $r=rank(M)=1=n-1$; n为进程数=2

所以左图至少需要节点 a：1 b：1

右图:

$$\begin{cases} 2a-b=0 \\ b-a=0 \end{cases} \tag{6}$$

拓扑矩阵为:
$$M=\begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix} \tag{7}$$

右图是inconsistent,因为$r=rank(M)=2=n$;n为进程数=2

2

拓扑矩阵为:

$$M=\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -77 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 0 & 0 & 77 \end{bmatrix} \tag{8}$$

化简为

$$M = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -77 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{9}$$

rank(M)=5;进程数量n=6;此时$r = n - 1$所以矩阵是一致的（consistent）

需要的节点数量量为Quelle：77 ; DCT：77 ; Q ： 77 ; RLC：77; C：1; R：1。