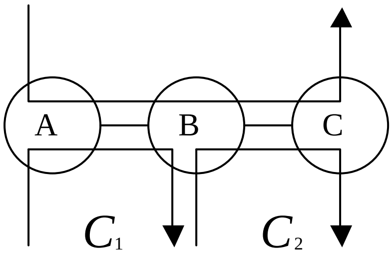
Multi queues　**電話系統**



C_1 、 C_2　truck Capacity

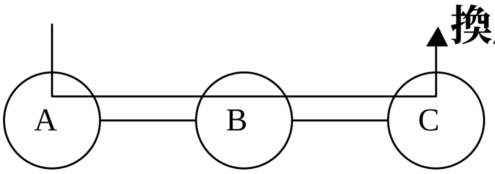
Sol:①Model the whole system

(# of　connection’s　between A,B)
(# of　connection’s　between B,C)
(# of　connection’s　between A,C)
disadvantage: State space too large
②Independent Assumption

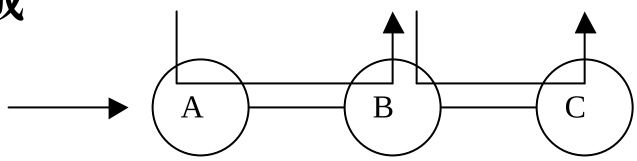
-Poisson arrival (**在複雜的系統不可能是** Poisson**，因為可能被** block **住**)
但為了要用 M/M/?**分析又不得不用** Poisson **分析**。

-Link independent
A call carried on a n-link path behaves like n dependent calls on each link.

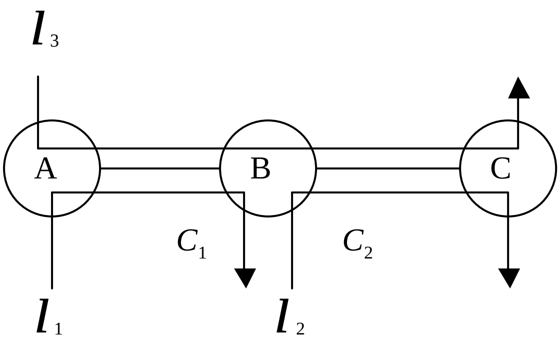
將



換成



如果上面的 Assumption **成立的話**
=>Link isolation (decomposition)



分析 C_1： M/G/C/C

Erlang B 公式: $EB_1 = \dfrac{\dfrac{r_1^{C_1}}{C_1!}}{\sum_{i=0}^{c_1} \dfrac{r_1^i}{i!}}$, $r_1 = \dfrac{l_1 + l_3}{m}$ (node A)

$$C_2 \ : EB_2 = \frac{\dfrac{r_2^{C_2}}{C_2!}}{\sum_{i=0}^{c_2} \dfrac{r_2^i}{i!}}$$

這樣的分析不準，以 l_3 **為例有可能在到達** B **時被** block **住。**

- Reduce Load approximation

Arrival rate to C_1 should be reduce to $l_3(1 - EB_2)$

(**先確定在後半段的** link **不會被** block **住**)

In general $l \prod_{\substack{j=1 \\ j \neq i}}^{m} (1 - EB_j)$

To: C_2 : $l_3(1 - EB_1)$

$$EB_1 = \frac{\dfrac{r_1^{C_1}}{C_1!}}{\sum_{i=0}^{c_1} \dfrac{r_1^i}{i!}}, \quad r_1 = \frac{l_1 + l_3(1 - EB_2)}{m}$$

$$EB_2 = \frac{\dfrac{r_2^{C_2}}{C_2!}}{\sum_{i=0}^{c_2} \dfrac{r_2^i}{i!}}, \quad r_2 = \frac{l_2 + l_3(1 - EB_1)}{m}$$

To compute EB_1 needs EB_2 and compute EB_2 needs EB_1?

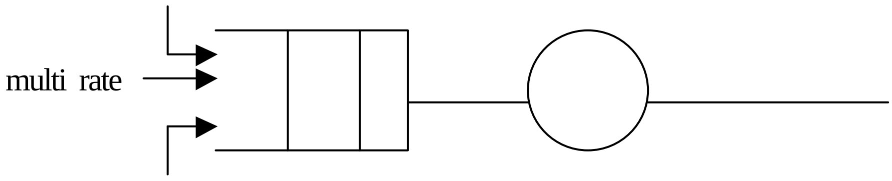
Solution: Iteration (fix-point)

Initial: $EB_1 = EB_2 = 0$

Iteration i : use EB_1^{i-1}, EB_2^{i-1}

- Multi-dimension approximation

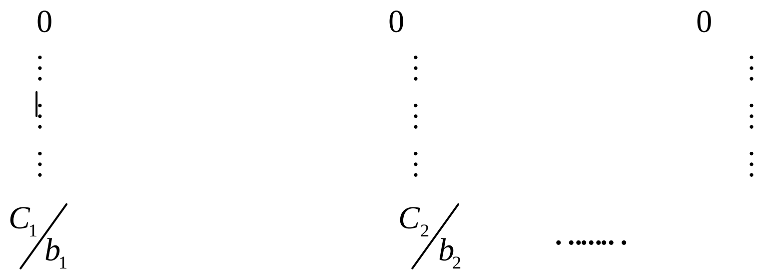
$l_1(b_1)$



$l_n(b_n)$ M/M/C/C

Approach1: state descriptor

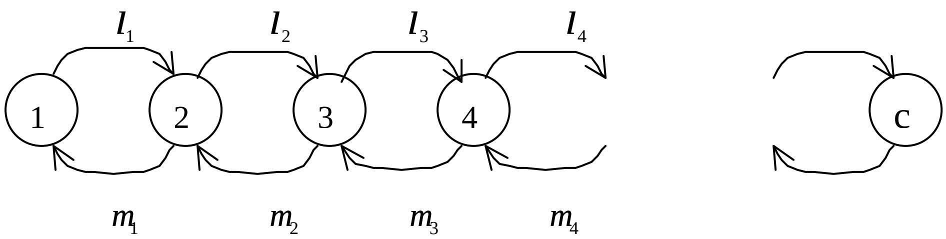
(# of class 1 call,# of class 2 call, ...)



缺點 matrix **成長太快，** dimension **太大難以計算。**

Approach2: Single dimension state?(**失去**Marcov property，**需要額外的**history**記錄**)

(# of busy circuits)



PASSAL approximation

$$l_i = \frac{e^2}{s^2} + i(1 - \frac{e}{s^2})$$

$$m_i = im$$

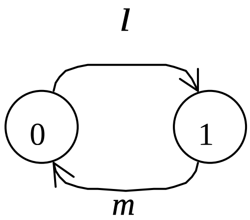
$$r_i = \frac{l_i}{m_i}$$

$$e = \sum_{i=1}^{n} b_i r_i$$

$$s^2 = \sum_{i=1}^{n} b_i^2 r_i$$

CTMC

$$P_{ij}^{'}(t) = \sum_{k=0}^{\infty} P_{ik} Q_{kj} \qquad , P(t) = S + T(t)$$

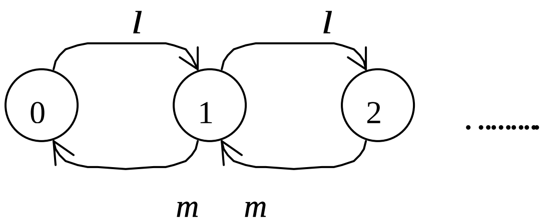


$$P(t)=\begin{bmatrix}\frac{m}{l+m} & \frac{l}{l+m}\\ \frac{l}{l+m} & \frac{m}{l+m}\end{bmatrix}+e^{-(l+m)t}\begin{bmatrix}\frac{l}{l+m} & \frac{-l}{l+m}\\ \frac{-m}{l+m} & \frac{m}{l+m}\end{bmatrix}$$

CTMC with reward

做 transition **或待在某個** state **會有收入。**

(reward for transition is constant, reward for staying at a state is rate)



r_{ij} : reward for transition from i to j

r_{ii} :reward rate for staying at state i

$V_i(t)$:reward earned during time t when the process starts at i

$$\frac{dV_i(t)}{dt}=r_{ii}+\overbrace{\sum_{j\neq i}q_{ij}r_{ij}}^{q_i}+\sum_{j=0}^{\infty}q_{ij}V_j(t)$$

transition rate

$V'(t)=q+QV(t)$

$r_{11}=6, r_{23}=-3, r_{12}=r_{21}=0$

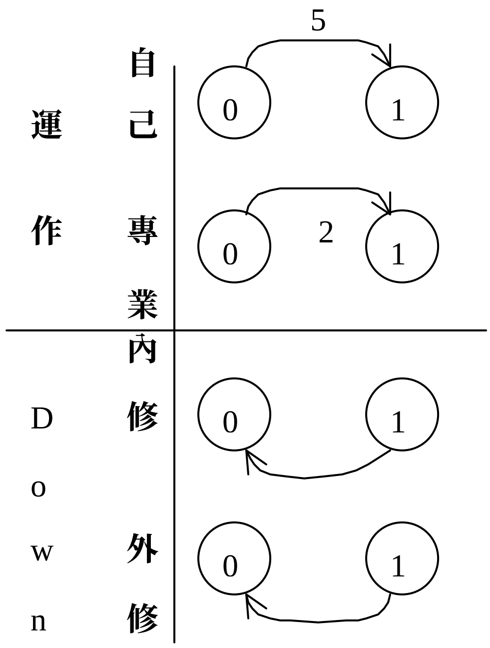
$$V(t)=\begin{bmatrix}1\\1\end{bmatrix}t+\begin{bmatrix}\frac{-5}{9}\\ \frac{-4}{9}\end{bmatrix}+e^{-9t}\begin{bmatrix}\frac{-5}{9}\\ \frac{4}{9}\end{bmatrix}$$

$$V_1(t)=t+\frac{5}{9}-\frac{5}{9}e^{-9t}$$

$$t->\infty, V(t)=tg+V$$

$$V_1(t)=t*1+\frac{5}{9}$$

example :Markov decision process



$$r_{00}=6, r_{01}=0$$

$$q_0=r_{00}+\sum_{i\neq j}q_{ij}r_{ij}=6+0$$

$$r_{00}=4, r_{01}=0$$

$$q_0=4+0=4$$

$$r_{11}=-1, r_{10}=-0.5$$

$$q_1=-1+4*-0.5=-3$$

$$r_{11}=-1.5, r_{10}$$

$$q_1=-1.5+7*-0.5=-5$$

A_i:{set of actions that state i can take}

Policy: Each state takes an given action

算法是每個 policy **代入** CTMC with reward **但太麻煩了，一種要算一次。**

Markov decision Theory

Policy Iteration Procedure

-Value determination step

For a given policy,solve

V(t)=tg+v

Where $g_i=q_i+\sum_{j=0}^{\infty}q_{ij}V_j$

If the CTMC is complete Ergodic

g_i=g $\forall i$

then solve $g=q_i+\sum_{j=0}^{\infty}q_{ij}V_j$ get g and V_j

-Policy Improvement step

Use V_j,for each state I find action a_k such that

$$q^k + \sum_{j=0}^{\infty} q_{ij}^k V_j \ \text{ is } \max \forall k$$

Get an improved policy

Repeat the iteration until converge to an optimal policy.

Ex: initial **都選自己**

-Value determination step

$$g = 6 - 5V_0 + 5V_1$$

$$g = -3 + 4V_0 - V_1$$

let V_1=0

g=1,V_0=1

-Policy improvement step

state 0:

$$\left.\begin{array}{l} a_1^1 : 6 - 5V_0 + 5V_1 = 6 - 5 = 1 \\ a_2^1 : 4 - 2V_0 + 2V_1 = 4 - 2 = 2 \end{array}\right. \Rightarrow \text{choose } a_2^1$$

state 1:

$$\left.\begin{array}{l} a_1^2 : -3 + 4V_0 - 4V_1 = -3 + 4 = 1 \\ a_2^2 : -5 + 7V_0 - 7V_1 = -5 + 7 = 2 \end{array}\right. \Rightarrow \text{choose } a_2^2$$

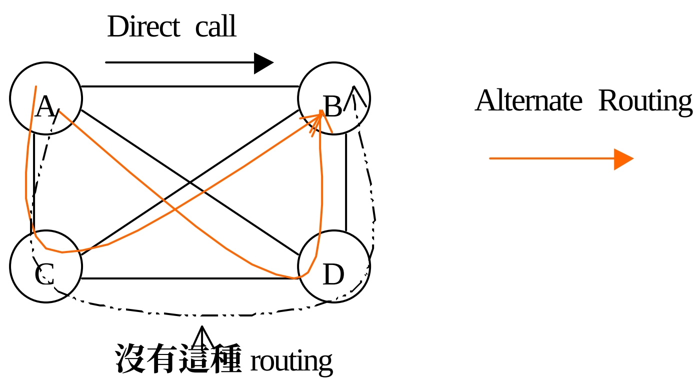
-Value determination step

$$\begin{cases} g = 4 - V_0 + 2V_1 \\ g = -5 + 7V_0 - 7V_1 \end{cases} \text{let } V_1=0, g=2, V_0=1$$

V_1 、V_0 **一樣**

=>optimal policy a_2^1, a_2^2

Alternate Routing in circuit-switched Networks



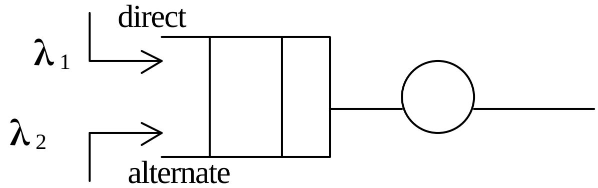
Link independent

==>Single Link single rate multiple classes

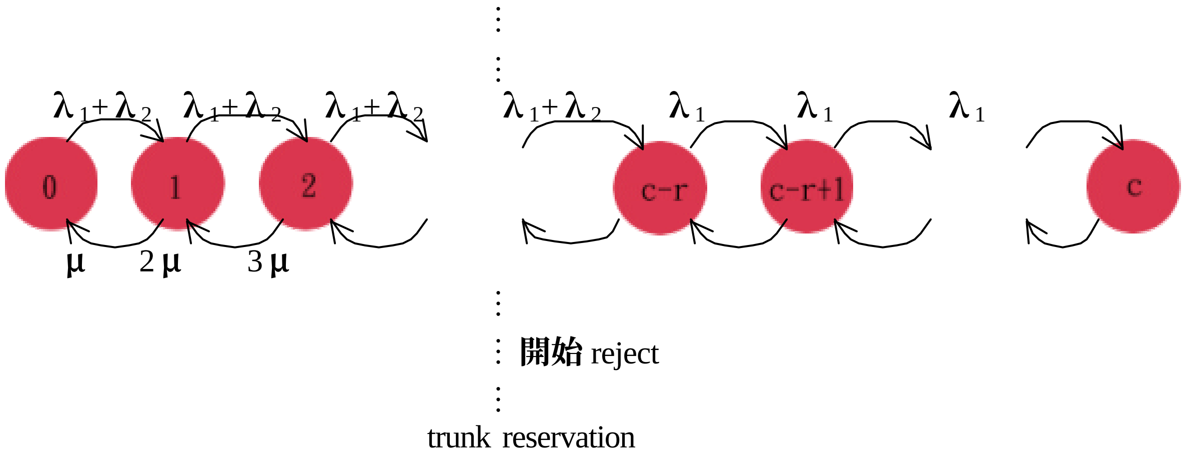
reward $r_1 > r_2$

r_1

r_2



Trunk reservation to prevent excess alternate calls



用 reward 的觀點看不出是 direct call 還是 alternate call，改以 lose 的觀點若有拒絕會 lose 多少。

$$q_i = \begin{cases} 0 & if\ i\ accept\ alt.\ call \\ l_2 r_2 & if\ i\ reject\ alt.\ call \\ l_1 r_1 + l_2 r_2 & i = C \end{cases}$$

-Value determination step

let $W_i = V_{i+1} - V_i$

$$i = 0 \quad g = 0 - (l_1 + l_2)V_0 + (l_1 + l_2)V_1 = (l_1 + l_2)W_0$$

$$i = 1 \quad g = 0 + mV_0 - (l_1 + l_2 + m)V_1 + (l_1 + l_2)V_2 = (l + l_2)W_1 - mW_0$$

$$i = 2 \quad g = l_2 r_2 + 2mV_1 - (l_1 + 2m)V_2 + l_1 V_3 = l_2 r_2 + l_1 W_2 - 2mW_1$$

$$i = 3 \quad g = l_1 r_1 + l_2 r_2 + 3mV_2 - 3mV_3 = l_1 r_1 + l_2 r_2 - 3mW_2$$

$$W_0 = \frac{g}{(l_1 + l_2)}$$

$$W_1 = \frac{(l_1 + l_2 + m)g}{(l_1 + l_2)^2}$$

$$\vdots$$

$$W_i = \frac{g}{\overline{l}_{i-1} E(\overline{l}, i-1)}$$

$$E(\overline{l}, 0) = \frac{\frac{l_1 + l_2}{m}}{1 + \frac{l_1 + l_2}{m}} = \frac{l_1 + l_2}{l_1 + l_2 + m}$$

有了 W_i 就可以不用解 linear eq. 讓 routing 可以即時。