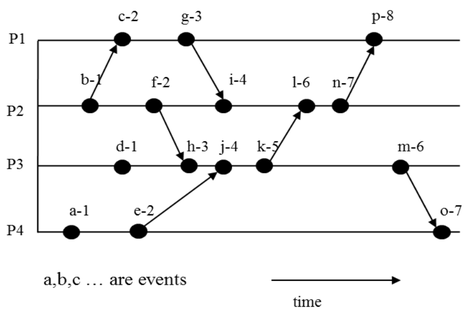
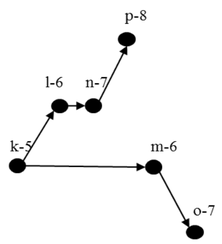
solutions available at http://mathieu.delalandre.free.fr/teachings/dcomputing.html

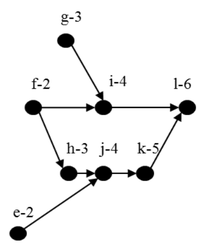
Q11





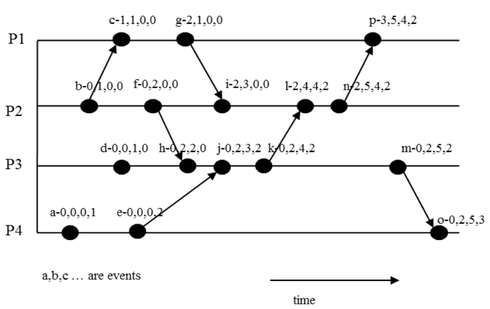
Happens-before	k → l → n → p k → m → o
Concurency	l||m, n||o

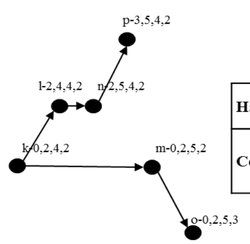
The event order is k, l, m, n, o, p



Happens-before	g → i f → i → l f → h, k → l h → j → k e → j
Concurency	f||e, g||h, i||j

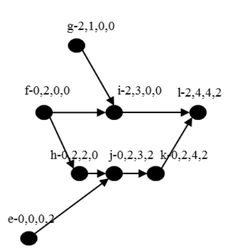
The event order is f, e, g, h, i, j, k, l





Happens-before	k → l → n → p k → m → o
Concurency	l||m, l||o n||m, n||o p||m, p||o

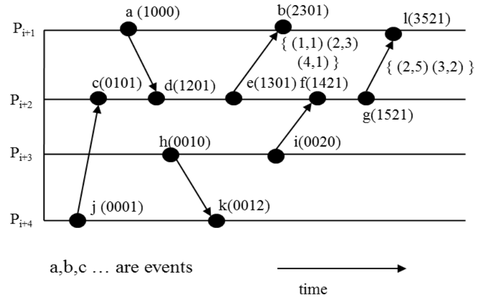
The event order is k, l, n, p, m, o



Happens-before	g → i f → i → l f → h, k → l h → j → k e → j
Concurency	g||(f, h, j, k, e) f||(g, e) h||(g, i, e) e||(g, i, f, h) i||(h, j, k, e) j||(g, i) k||(g, i)

The event order is g, f, i, h, e, j, k, l

Q12



		(c)	(d)	(e)	(f)	(g)
LS_{i+2}	0 0 0 0			3 0 0 0		5 0 0 0
LU_{i+2}	0 0 0 0	0 1 0 1	2 2 0 1	2 3 0 1	2 4 4 1	2 5 4 1

At (e) $\begin{pmatrix}2\\3\\0\\1\end{pmatrix} \overset{?}{>} (LS_{i+2}(1)=0) = \begin{pmatrix}1\\1\\0\\1\end{pmatrix}$

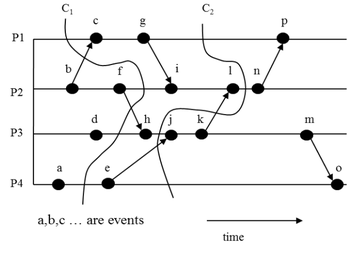
The { (1,1) (2,3) (4,1) } timestamp will be sent.

At (g) $\begin{pmatrix}2\\5\\4\\0\end{pmatrix} \overset{?}{>} (LS_{i+2}(1)=3) = \begin{pmatrix}0\\1\\1\\0\end{pmatrix}$

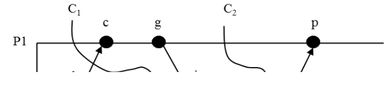
The { (2,5) (3,2) } timestamp will be sent.

Q2

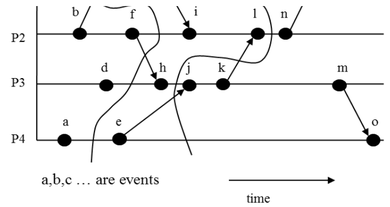
history	**H**	a b c d e f g h i j k l m n o p			
	h1	c g p			
	h2	b f i l n			
	h3	d h j k m			
	h4	a e o			
Cuts	**C1**	b f d a	**frontier**	f d a	**concistent**
	C2	c g b f i l d h a e	**frontier**	g l h e	**no consistent**



run and linearization	run	linearization	
c g f d b h	no	no	f is not happens-before of b in local execution of P2
f e h i g j	yes	no	i is not happens-before of g
d b c a h e j	yes	no	we have f happens-before of h but f is not in the event cut



f g i h e j k l yes yes



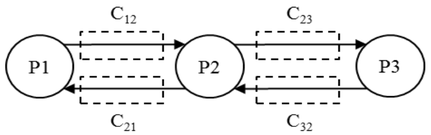
Q3

Time	Events	Rules
	initial state	
t0	p1 sends order 30 to p2	
t1	p2 sends order 90 to p3	
t2	p2 starts the snapshot	(1)(2)
t3	p3 receives 90 from p2	
t4	p2 receives 30 from p1	
t5	p3 sends order 210 to p2	
t6	P1 receives a marker	(3)(4)(1)(2)
t7	P2 sends order 80 to P1	
t8	P3 receives a marker	(3)(4)(1)(2)
t9	P2 receives a marker	(5)
t10	P2 receives 210 from P3	
t11	P2 receives a marker	(5)
t12	p1 receives 80	

P1		
€	S1	C21
600	0	0
570		
	570	0
650		
	570	0

P2			
€	state S2	C12	C32
150	0	0	0
60			
	60		
90			
10			
		30	
220			
			210
	60	30	210

P3		
€	state S3	C23
350	0	0
440		
230		
	230	0
	230	0

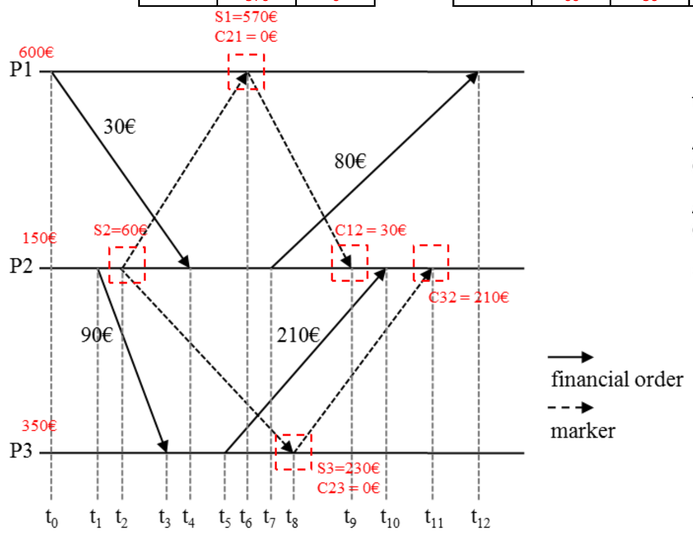


Marker Sending Rule ($\mathbf{M_{SR}}$) at process $\mathbf{P_i}$
(1) Record the state $\mathbf{S_i}$ of $\mathbf{P_i}$
(2) **For** each outgoing channel $\mathbf{C_{ij}}$ on which a marker has not been sent, send a marker along $\mathbf{C_{ij}}$ before any other message

Marker Sending Rule

Marker Receiving Rule ($\mathbf{M_{RR}}$) at process $\mathbf{P_j}$
On receiving a marker along channel $\mathbf{C_{ij}}$
If $\mathbf{P_j}$ has not recorded its state
(3) Record the state of $\mathbf{C_{ij}}$ as empty
(4) Execute the $\mathbf{M_{SR}}$
Else
(5) Record the state of $\mathbf{C_{ij}}$ as the set of messages received over $\mathbf{C_{ij}}$ since the last state record of the process

Marker Receiving Rule



The snapshot stops at t9 then t11

At t9, the local recored state for P2, P1 is different from local execution state (60€+30€, 570€+0€) ≠ (10€, 570€)

At t11, the local recored state for P2, P3 is different from the local execution state (60€+210€, 230€+0€) ≠ (220€, 230€)

P1, P2, P3 change their states asynchronously compared to the snapshot