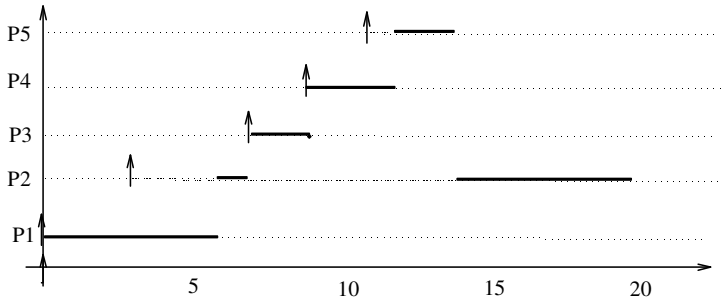
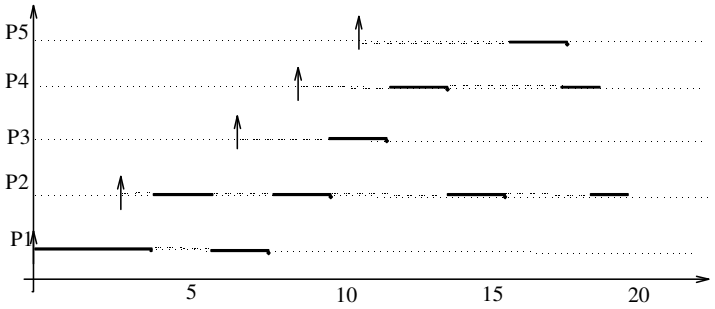
Corrigé Examen SE

Exercice1 (/4) :
Soit un système ayant un processeur de calcul préemptif. Donner l'ordonnancement et le temps d'attente des processus suivants selon les algorithmes :
1) SRTF : P1 →P2→P3→P4→P5→P2 (**1,5**)
2) Tourniquet avec quantum=2 :
P1→P2→P1→P2→P3→P4→P2→P5→P4→P2 (**2,5**)
.

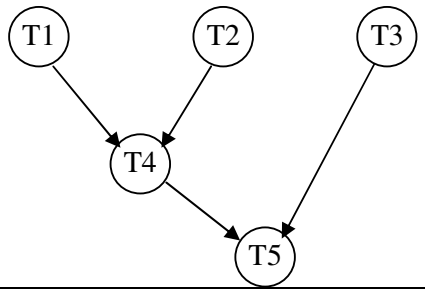
Processus	date d'arrivée	Temps d'exécution	T-attente SRTF	T-attente RR
P1	0	6	0	2
P2	3	7	10	10
P3	7	2	0	3
P4	9	3	0	7
P5	11	2	1	5





Exercice 2 (/6): Soit le système parallèle ci-dessous:
1) graphe de précédence (**0,5**)

T1 : t1=a+b (1ms)
T2 : t2=a*b (2ms)
T3 : t3=a/b (5ms)
T4 : t4= t1*t2 (2ms)
T5 : t5= t3-t4 (1ms)



2) Donner le programme parallèle en utilisant les primitives de Conway (fork/join) (**1,75**)
3) Donner le programme parallèle maximal en utilisant les primitives de Dijkstra (ParBegin/ParEnd) (**1,75**)

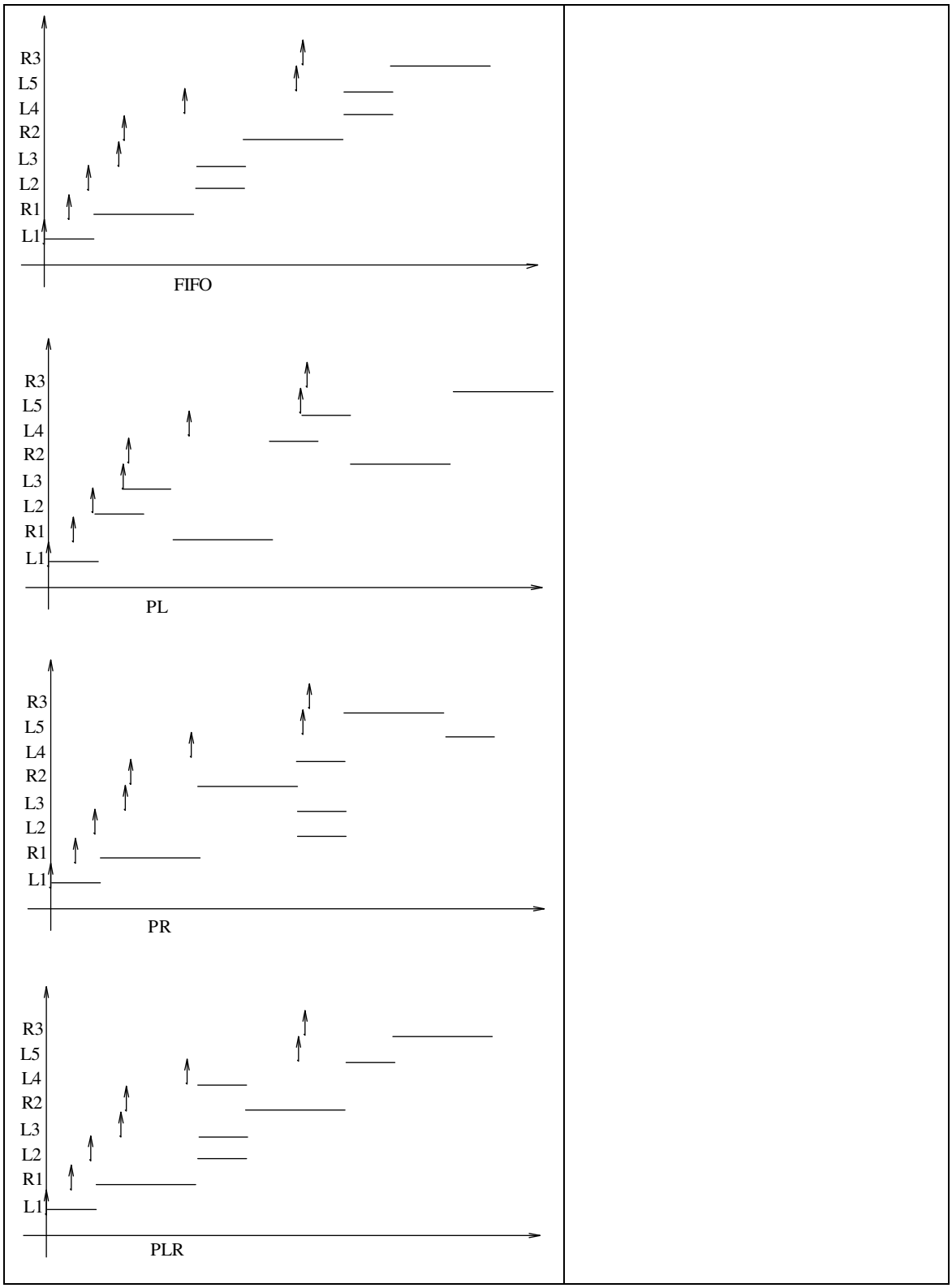
Conway :	Dijkstra :
fork L1 ; T1 ; goto L2; L1 : fork L3 ; T2 ; L2 : join n1=2 ; T4 ; goto L4 ; L3 : T3 ; L4 : join n2=2 ; T5	begin parbegin begin parbegin T1 ; T2; parend; T4; end T3; parend T5 ; end

4) Si on disposait de 3 processeurs pour exécuter les trois taches P1pour T1, P2 pour T2 et P3 pour T3 : quel processeur va executer T4 et T5 ? Justifier
P1 :T1(1ms)→(1ms attente)→**T4(2ms) (Total : 4ms) (1)**
P2 :T2(2ms)→(3ms attente) →T5(1ms) (Total : 6ms) (**1**)
P3 :T3(5ms) (Total : 5ms)

Exercice3 (/6) : Soit le modèle lecteur rédacteur:
1) Donner l'ordre d'accès à la section critique de ces processus lecteurs et rédacteurs suivant les 4 variantes en sachant que la durée d'une lecture est de 1ns et d'une écriture est de 2ns:

Processus	Lect1	Redact1	Lect2	Lect3	Redact2	Lect4	Lect5	Redact3
date d'arrivée	0ns	0.5ns	0.9ns	1.5ns	1.6ns	2.9ns	5.1ns	5.2ns

FIFO : L1 ;R1 ;L2||L3 ;R2 ;L4 ||L5 ;R3 ; (**0,25**)
Priorité aux lecteurs: L1||L2||L3;R1;L4||L5;R2 ;R3 ; (**0,5**)
Priorité aux rédacteurs : L1 ;R1 ;R2 ;L2||L3||L4 ;R3 ;L5 ; (**0,5**)
Priorité aux lecteurs sans famine des rédacteurs : L1 ;R1 ;L2||L3||L4 ;R2 ;L5 ;R3 ; (**0,75**)



2) Soit la solution suivante du modèle. Vérifier que cette solution garantit les contraintes du modèle et en déduire la variante

Shared semaphore S=1, Se=1, mutex=1;
Shared int nb=0;

Lecteur(){ while(TRUE){
P(Se);
P(mutex);
nb++;
if(nb==1)P(S);
V(mutex);
V(Se);
lire();
P(mutex) ;
nb-- ;
if(nb==0) V(S) ;
V(mutex);
}}
	Rédacteur(){while(TRUE){
	P(Se);
	P(S) ;
	V(Se) ;
	écrire() ;
	V(S) ;
	}}

La variante : FIFO ! tous les processus commencent par bloquer dans la file d'attente d'un seul sémaphore Se. Se donne l'ordre d'accès FIFO à la ressource. (**1**)
Lecture simultanée : Si une lecture est en cours, si le suivant est un rédacteur alors il sera bloqué dans S et les suivants dans Se. Si le suivant est un lecteur alors (nb=2) et va passer la section d'entrée et va pouvoir lire simultanément. (**1,5**)
Ecriture en EM : Le sémaphore S permet de garantir l'EM. Si un rédacteur est en cours d'écriture, si le suivant est un rédacteur ou un lecteur il sera bloqué dans S et les autres dans Se.(**1,5**)

Exercice4 (/4) : Soit le problème de l'interblocage:
Soit le système d'allocation de ressources suivant qui utilise le Banquier:

Processus	Alloc			Max			Bes			Disp		
	R1	R2	R3	R1	R2	R3	R1	R2	R3	R1	R2	R3
P0	1	1	1	1	2	4	0	1	3	2	3	1
P1	0	1	0	3	3	2	3	2	2			
P2	2	0	3	4	4	3	2	4	0			
P3	1	2	0	2	4	3	1	2	3			
P4	0	1	0	2	3	1	2	2	1			

1) Vérifier si l'état du système est sain.
séquence saine : P4→P2→P0→P1→P3 (seul P4 puis P2 peuvent débuter cette séquence) (**1**)
2) Proposer une requête possible qui sera acceptée par le banquier
P4 qui demande les ressources dont il a besoin ou P2 qui demande une instance de R2(**1,5**)
3) Proposer une requête possible qui sera refusée par le banquier pour des raisons d'état virtuel non sain.
si P0 ou P1 ou P3 demande 1 instance de R3, cela va bloquer P4 , ou une instance de R2 ce qui va bloquer P2 et donc pas de séquence saine. (**1,5**)