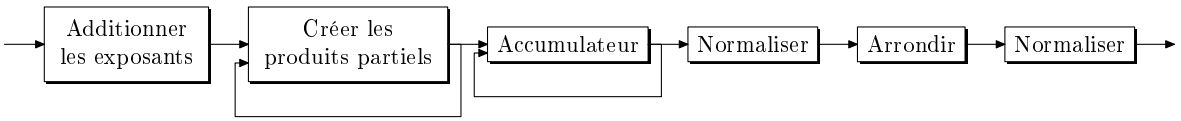
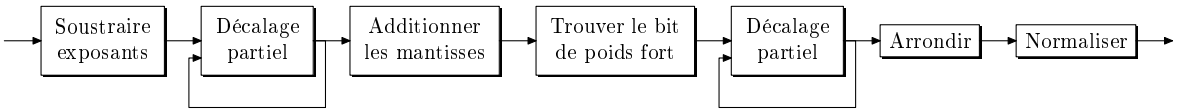
Pipeline

1 Contrôle d'un pipeline multifonctions

Nous nous intéressons aux pipelines représentés figures 1(a) et 1(b).



(a) Multiplicateur pipeliné avec boucles arrières



(b) Additionneur pipeliné avec boucles arrières

FIG. 1 – Pipelines arithmétiques.

▷ **Question 1** *Un grand nombre d'unités de calcul sont communes aux deux pipelines figures 1(a) et 1(b). Construire un pipeline multifonctions permettant d'accomplir ces deux opérations.*

Définition 1 (Table de réservation)**.** La table de réservation d'une opération dans un pipeline est un tableau à deux dimensions dont la première est indexée par les différentes unités du pipeline et la seconde par le temps. Elle permet de savoir, quand on introduit une opération au top $t = 1$, quelles seront les unités occupées aux tops suivants $t = 2$, $t = 3$, etc.

▷ **Question 2** *Construire les tables de réservation des différentes unités de calcul pour chacune des deux opérations en supposant qu'une unité effectuant une addition est aussi capable d'effectuer une soustraction ou de servir d'accumulateur.*

Définition 2 (Vecteur de collision)**.** On définit le vecteur de collision de deux opérations op_1 et op_2 par :

$$col_{op_1,op_2}(i) = \begin{cases} 1 & \text{si une collision se produit quand } op_2 \text{ est exécutée } i \text{ tops après } op_1 \\ 0 & \text{sinon} \end{cases}$$

▷ **Question 3** *En s'aidant des tables précédentes, calculer les vecteurs de collision d'une multiplication suivie d'une multiplication, d'une addition suivie d'une addition, d'une multiplication suivie d'une addition et d'une addition suivie d'une multiplication.*

▷ **Question 4** *Proposer un algorithme* simple *de contrôle pour un pipeline monofonction utilisant le vecteur de collision.*

▷ **Question 5** *Proposer un algorithme de contrôle pour le pipeline adapté à l'addition et à la multiplication.*

2 Efficacité du pipeline

À la suite de l'exercice précédent, on étudie les relations entre le vecteur de collision et le rendement du pipeline.

Définition 3 (Graphe d'états)**.** Le graphe d'états d'un vecteur de collision est un graphe orienté dont les sommets sont étiquetés par chacun des états possibles du registre à décalage lorsqu'une opération entre dans le pipeline. Dans ce graphe, deux sommets S_1 et S_2 sont reliés par un arc étiqueté par i si S_2 est l'état du registre à décalage quand on insère une opération dans le pipeline i tops après que le registre a été dans l'état S_1.

▷ **Question 6** *Construire le graphe d'états du vecteur de collision* 1001011*.*

▷ **Question 7** *Quel est le débit optimal du pipeline associé à ce vecteur de collision ?*
▷ **Question 8** *Donner un vecteur de collision pour lequel l'algorithme glouton (question 4 exercice 1) n'est pas optimal.*

▷ **Question 9**
- *Donner le diagramme d'une table de réservation possible pour le vecteur de collision* 10011*.*
- *Quel est le débit maximal pour la table de réservation suivante ?*

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*
×	×						
×				×			
×						×	
×							×

- *Quel est le débit maximal pour la table de réservation suivante ?*

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
×	×								
×					×				
×							×		
×									×

3 Retarder pour aller plus vite

Dans cet exercice, nous allons montrer pourquoi et comment le rendement des pipelines peut être amélioré en retardant certaines opérations. Pour cela, nous allons nous intéresser au pipeline dont la table de réservation est donnée figure 2.

	1	2	3	4	5	6	7
A	×		×	×			
B		×				×	
C					×		×
D				×			

Fig. 2 – Table de réservation.

▷ **Question 10** *Donner le vecteur de collision ainsi que son graphe d'états. Quelle est la valeur maximale du rendement de ce pipeline ?*

La ligne de la table de réservation qui comporte le plus de cases réservées × en comporte trois, donc le rendement sera au mieux 1/3. Il est possible d'atteindre cette valeur en introduisant des délais[1]. Pour le montrer, on va essayer de voir s'il est possible d'introduire des retards afin de lancer les opérations avec, entre chaque lancement, la suite d'intervalles $(d_1, d_2, \ldots, d_p)$.

[1] On peut retarder certaines réservations du moment qu'on respecte les dépendances.

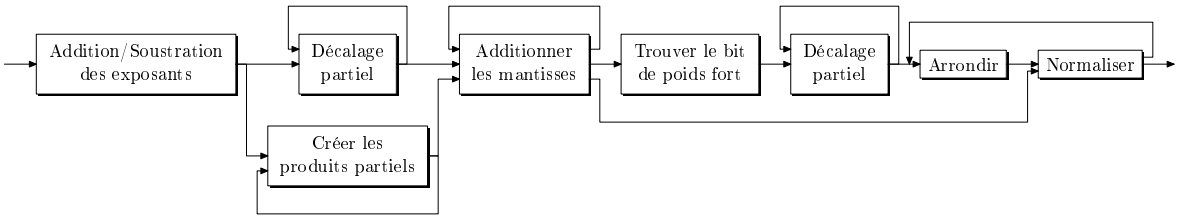
▷ **Question 11** *Montrer qu'il n'est pas possible d'initier les opérations de la table figure 2, même en insérant des délais, avec les intervalles de temps* 2 *et* 4.

▷ **Question 12** *Est-il possible d'insérer des délais dans la table de réservation de façon à avoir un débit de* 1 *opération tous les* 3 *tops ? Donner une table de réservation plus efficace que la précédente.*

▷ **Question 13** *Généraliser le résultat précédent.*

4 Réponses aux exercices

▷ Question 1, page 1



▷ Question 2, page 1

On construit les deux tables suivantes :

	1	2	3	4	5	6	7
Add. exp.	×						
Prod. partiels		×	×				
Add. man.			×	×			
Normaliser					×		×
Arrondir						×	
Shift A							
Poids fort							
Shift B							

Table de réservation pour la multiplication

	1	2	3	4	5	6	7	8	9
Add. exp.	×								
Prod. partiels									
Add. man.				×					
Normaliser									×
Arrondir								×	
Shift A		×	×						
Poids fort					×				
Shift B						×	×		

Table de réservation pour l'addition

▷ Question 3, page 1

Il suffit pour calculer le vecteur de collision d'une opération op_2 avec une opération op_1 de décaler la table de réservation de op_2 et de la superposer avec la table de réservation de op_1. Ainsi, il n'est pas possible de commencer une deuxième multiplication (notée △) un top après l'entrée d'une multiplication (notée ×) dans le pipeline :

	1	2	3	4	5	6	7
Add. exp.	×	△					
Prod. partiels		×	×△	△			
Add. man.			×	×△	△		
Normaliser					×	△	×
Arrondir						×	△
Shift A							
Poids fort							
Shift B							

Fort de cette remarque, il devient facile de calculer les vecteurs de collision suivants :

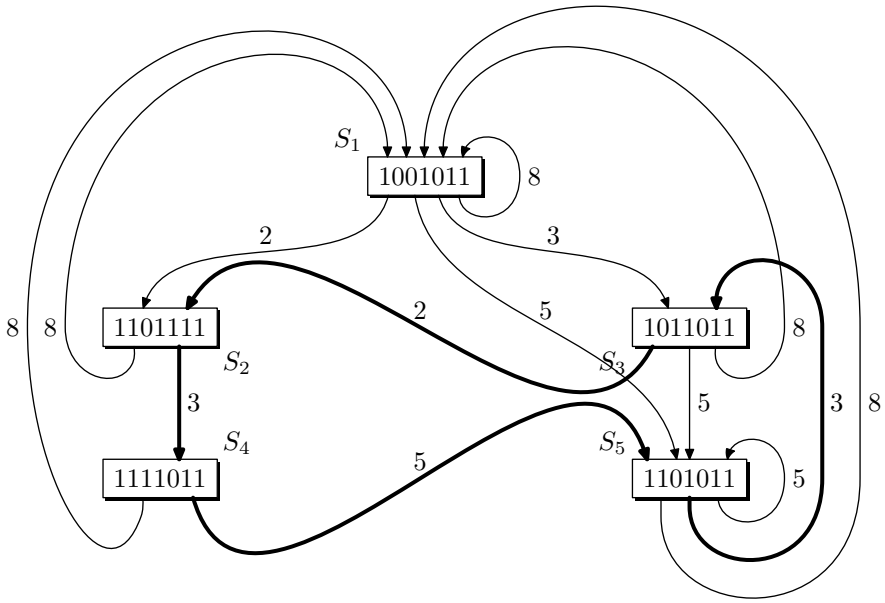


FIG. 3 – Graphe d'états du vecteur de collision 1001011.

- Mult-Mult : 110000
- Add-Add : 10000000
- Mult-Add : 00000000
- Add-Mult : 11010000

▷ Question 4, page 1

Comment contrôler un pipeline dynamiquement et rapidement ? On ne peut pas se permettre d'utiliser un algorithme compliqué, étant donné que la réponse doit pouvoir être calculée en moins d'un cycle.

Le contrôleur du pipeline utilise un simple registre à décalage[2] pour savoir quand l'opération peut commencer. On va tenir à jour le contenu de ce registre en calculant un OU bit à bit avec le vecteur de collision de la nouvelle opération.

Si une opération doit entrer dans le pipeline, on regarde le premier bit du registre à décalage. Si c'est un 0, alors il est possible de lancer une opération. Dans ce cas, on met à jour le contenu du registre à décalage en calculant un OU bit à bit avec le vecteur de collision et l'opération rentrera dans le pipeline au cycle suivant. Sinon, l'opération reste bloquée à l'entrée du pipeline tant que le premier bit du registre à décalage vaut 1.

▷ Question 5, page 1

Il suffit d'utiliser deux registres à décalages, le premier pour savoir quand une addition peut entrer dans le pipeline, et le second pour savoir quand une multiplication peut entrer dans le pipeline.

Quand une opération est en attente, on regarde dans le registre qui lui correspond si elle peut entrer ou non dans le pipeline. Le cas échéant, les deux registres sont mis à jours en calculant un OU bit à bit avec le vecteur de collision de l'opération.

▷ Question 6, page 2

Le diagramme figure 3 représente chacun des états possibles du registre à décalage lorsqu'une opération entre dans le pipeline. Les nombres sur les arcs représentent le nombre de cycles s'écoulant entre deux états. Par exemple, pour passer de l'état S_1 à l'état S_2, on a décalé le vecteur de collision 1001011 de deux rangs vers la gauche, ce qui donne 0101100 ; quand on fait le OU bit à bit avec 1001011, on obtient 1101111.

[2]Un registre à décalage est un registre qui décale son contenu sur la gauche à chaque top d'horloge.

▷ Question 7, page 2

Le débit d'un pipeline s'obtient en cherchant des cycles dans le graphe d'état du vecteur de collision qui lui est associé. Le rendement pour un cycle donné est le rapport entre le nombre d'états du cycle et la somme des poids des arcs. Par exemple, le cycle (la boucle en fait) constitué autour de S_5 est de longueur 5 et a donc un rendement de $1/5 = 0,2$: on initie en moyenne une opération tous les cinq tops. Le cycle le plus performant du graphe précédent est le cycle $S_2 - S_4 - S_5 - S_3$ dont le rendement vaut $4/13 = 0,31$.

▷ Question 8, page 2

Sur l'exemple précédent, l'algorithme glouton initie les opérations dès que possible. Partant de l'état initial S_1, l'algorithme glouton effectue le chemin $S_1 - (S_2 - S_4 - S_5 - S_3)^*$ et trouve donc le cycle optimal.

Construisons donc un autre exemple : avec le vecteur de collision 100110 (état S_1), l'algorithme glouton va en deux tops à l'état S_2 de vecteur 111110, puis revient en six tops à l'état S_1. On obtient un cycle de rendement $2/8 = 0,25$. Par contre, en partant de S_1, on peut aller en trois tops à l'état S_3 de vecteur 110110 et boucler autour de S_3 tous les trois tops pour un rendement $1/3 = 0,33$.

▷ Question 9, page 2

– Il est aisé de construire de façon automatique une table de réservation qui crée les conflits d'un vecteur de collisions données :

1	2	3	4	5	6
×	×				
×				×	
×					×

– Le vecteur de collision de cette table est le vecteur 1001011, et nous avons vu plus haut que son débit optimal est égal à $4/13 = 0,31$.

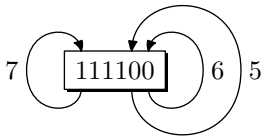
– Le vecteur de collision de cette table est le vecteur 100010101. Étant donné que deux opérations de ce type ne peuvent pas entrer dans le pipeline à moins de deux tops d'horloge d'intervalle, son débit optimal est inférieur à 1/2. Le tableau suivant montre qu'il est parfaitement possible d'initialiser une opération tous les deux tops d'horloge.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	...
1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	...
1		2		3	1	4	2	5	3	6	4	7	5	8	6	...
1		2		3		4	1	5	2	6	3	7	4	8	5	...
1		2		3		4		5	1	6	2	7	3	8	4	...

On peut remarquer que cette table de réservation correspond à la précédente où certaines opérations ont été retardées. Malgré ces retards, le débit du pipeline est grandement accru.

▷ Question 10, page 2

Le vecteur de collision est 111100 et son graphe d'états est le suivant :



Son rendement vaut donc $\frac{1}{5} = 0,2$.

▷ Question 11, page 2

On s'intéresse au régime permanent dans le cas où les instructions sont insérées dans le pipeline au temps $t \equiv 0[6]$ (on parlera des opérations d'indice impair) et $t \equiv 2[6]$ (on parlera des opérations d'indice pair). Il suffit donc de regarder les unités de temps modulo 6 pour détecter les conflits possibles.

L'unité de calcul A est utilisée pour tout $t \equiv 0[6]$ (pour la première étape de calcul des opérations impaires) et pour tout $t \equiv 2[6]$ (pour la première étape de calcul des opérations paires). Comme l'unité A doit être utilisée à temps plein si on souhaite produire 2 opérations tous les 6 cycles, en particulier, elle doit être utilisée au temps $t \equiv 4[6]$. Si elle l'est par une opération d'indice impair, alors une opération d'indice pair utilisera l'unité A au temps $t + 2 \equiv 0[6]$, et entrera en conflit avec la première étape d'une opération d'indice impair. Si elle l'est par une opération d'indice pair, alors une opération d'indice impair utilisera l'unité A au temps $t + 4 \equiv 2[6]$, et entrera en conflit avec la première étape d'une opération d'indice pair. Il est donc impossible d'initier des opérations avec les intervalles de temps 2 puis 4, et ce même en insérant des délais.

▷ Question 12, page 3

Il suffit qu'une unité donnée ne soit utilisée qu'à des moments distincts modulo la longueur du cycle. Par exemple, dans la table suivante :

	1	2	3	4	5	6	7	8
A	×		×		×			
B		×					×	
C						×		×
D				×				

- l'unité A est utilisée aux temps 1, 3 et 5, c'est-à-dire aux temps 1, 0, et 2 modulo 3 (la longueur du cycle) ;
- l'unité B est utilisée aux temps 2 et 7, c'est-à-dire aux temps 2 et 1 modulo 3 ;
- l'unité C est utilisée aux temps 6 et 8, c'est-à-dire aux temps 0 et 2 modulo 3 ;
- l'unité D est utilisée au temps 4, c'est-à-dire au temps 1 modulo 3.

Ainsi, on est sûr qu'il n'y aura pas de conflit d'utilisation d'une même ressource.

▷ Question 13, page 3

Si D est le nombre maximal de fois où une même opération utilise une ressource, il est possible de modifier la table de réservation en introduisant des délais de façon à obtenir une production de 1 opération tous les D tops d'horloge en régime permanent. Il suffit de prendre les réservations (toutes ressources confondues) en respectant leur dépendances et de retarder celles qui tombent sur un emplacement qui, modulo D, a déjà été utilisé. L'inconvénient de cette technique est l'allongement du temps de traitement des opérations.