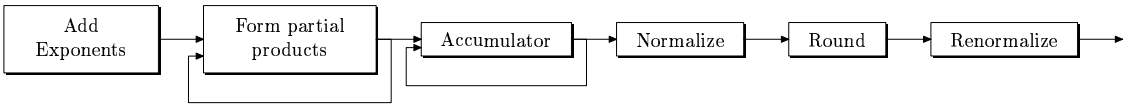
Pipeline

Résumé: Dans ce TD, nous allons, à travers des exemples simples, concevoir des pipelines, modéliser leurs performances et les améliorer.

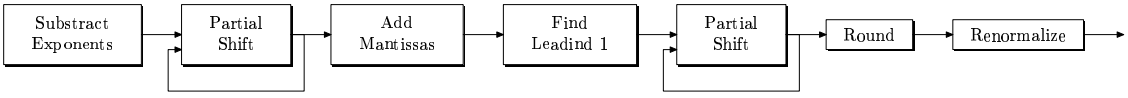
L'idée de base du pipeline est assez naturelle : elle consiste à découper une tâche complexe en une suite de micro-opérations afin de permettre de commencer à traiter une tâche alors qu'une autre est encore en cours de traitement. Le temps de traitement d'une tâche donnée ne diminue donc pas (il peut même être un peu plus long à la suite du découpage) mais le rendement des unités de calcul est augmenté. Le nombre de tâche traitées en un temps donné n'est plus fonction du temps de traitement d'une tâche mais de la vitesse à laquelle on peut introduire une nouvelle tâche dans le pipeline.

1 Contrôle

Dans cette section, nous nous intéressons aux pipelines représentés en Figure 1(a) et 1(b).



(a) Multiplicateur pipeliné avec boucles arrières

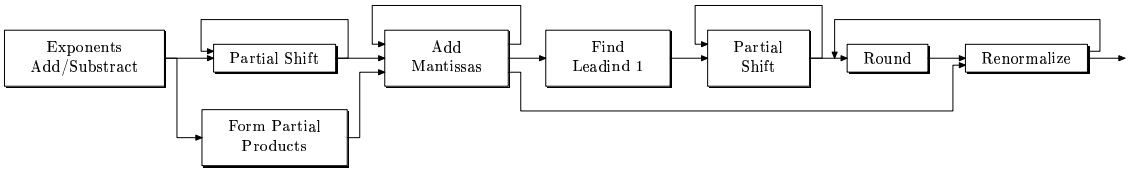


(b) Additionneur pipeliné avec boucles arrières

FIG. 1 – Pipeline

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

Réponse.



□

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 si, quand on introduit une opération au temps $t = 1$, quelles sont les unités occupées au temps t.

▷ **Question 2.** Construire les tables de réservations des différentes unités de calculs pour chacune des deux opérations.

Réponse. On peut construire les deux tables suivantes :

	1	2	3	4	5	6	7
Ex Add	×						
Mult		×	×				
Man Add			×	×			
Renorm					×		×
Round						×	
Shift A							
Lead 1							
Shift B							

Table de réservation pour la multiplication

	1	2	3	4	5	6	7	8	9
Ex Add	×								
Mult									
Man Add				×					
Renorm									×
Round								×	
Shift A		×	×						
Lead 1					×				
Shift B						×	×		

Table de réservation pour l'addition

□

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

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

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

Réponse. 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. Il est par exemple facile de voir qu'il n'est pas possible de commencer une multiplication un top après l'entrée d'une multiplication dans le pipeline :

	1	2	3	4	5	6	7
Ex Add	×	△					
Mult		×	×△	△			
Man Add			×	×△	△		
Renorm					×	△	×
Round						×	△
Shift A							
Lead 1							
Shift B							

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

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

□

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

Réponse. 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[1] 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.

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

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. L'opération reste bloquée à l'entrée du pipeline tant que le premier bit du registre à décalage vaut 1. □

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

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

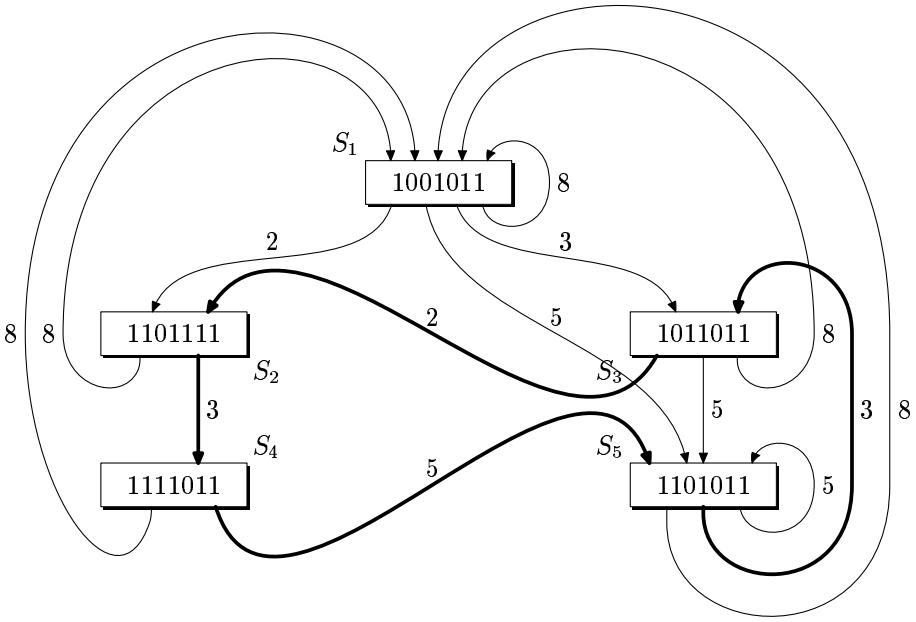
Quand une opération est en attente, on regarde dans le registre qui lui correspond si elle peut entrer ou non dans le pipeline. Le cas échéant, les deux registres sont mis à jours en calculant un OU bit-à-bit avec le vecteur de collision de l'opération. □

2 Efficacité

Définition 3 *(Graphe d'états).* Le graphe d'états d'un vecteur de collision est un graphe orienté dont les sommet 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 si on insère une opération dans le pipeline i tops après que le registre ait été dans l'état S_1.

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

Réponse. Le diagramme ci-dessous 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 cycle s'écoulant entre deux états.



□

▷ **Question 7.** Quel est le débit optimal du pipeline associé à ce vecteur de collision ?

Réponse. 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 cycles et la somme des poids des arcs. Par exemple le cycle constitué de S_5 est de longueur 5 et a donc un rendement de $1/5 = 0.2$. Le cycle le plus performant du graphe précédent est le cycle $S_2 - S_4 - S_5 - S_3$ dont le rendement vaut $4/13 = 0.31$. Il se trouve que c'est le cycle sur lequel tombe l'algorithme glouton dont le chemin dans le graphe est $S_1 - (S_2 - S_4 - S_5 - S_3)*$. □

▷ **Question 8.** Donner un vecteur de collision pour lequel l'algorithme glouton de la question4 n'est pas optimal.

Réponse. À chercher!!! □

▷ **Question 9.**

- Donner un diagramme 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
×	×								
×					×				
×							×		
×									×

Réponse.

- 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 dans une question précédente que son débit optimum était de $4/13 = 0.31$.
- 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 optimum 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 et que malgré ces retards, le débit du pipeline est grandement accru.

□

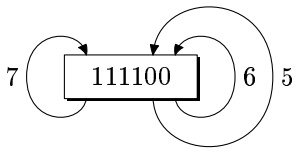
3 Retarder pour aller plus vite

Dans cette section, nous allons montrer pourquoi et comment le rendement de certains pipelines peut être amélioré en retardant certaines opérations. Pour cela, nous allons nous intéresser au pipeline dont la table de réservation est donné Figure 3.

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

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

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



□

Étant donné qu'il y a au plus 3 × dans chaque ligne de la table de réservation, le rendement est d'au mieux 1/3. Il est possible d'atteindre cette valeur en introduisant des délais[2]. Pour le montrer, on va essayer de voir s'il est possible d'introduire des retards afin de lancer les opérations avec comme intervalle entre chaque lancement $(d_1, d_2, \ldots, d_p)$.

▷ **Question 11.** Montrer qu'il n'est pas possible d'initier les opérations de la table 3, même en insérant des délais, avec les intervalle de temps 2 et 4.

Réponse. Étant donné que l'on s'intéresse au régime permanent, il suffit de regarder les unités de temps modulo 6 (la longueur du cycle que l'on souhaite créer). 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) puisque l'on veut lancer les opérations avec comme intervalle de temps 2 et 4 entre chaque lancement. 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 impaire, alors une opération paire utilisera l'unité A au temps $t + 2 \equiv 0[6]$, et entrera en conflit avec la première étape d'une opération impaire. Si elle l'est par une opération paire, alors une opération impaire utilisera l'unité A au temps $t + 4 \equiv 2[6]$, et entrera en conflit avec la première étape d'une opération paire. 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.** 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.

Réponse. 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 $1, 0, 2$ 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.** Généraliser le résultat précédent.

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

Réponse. Si D est le nombre maximum 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 le rallongement du temps de traitement des opérations. □