

TD d'architecture des ordinateurs V

Performance des caches

Défaut de page

Exercice 1 : Caches et temps CPU

Rappel : Le temps CPU peut se décomposer en deux parties indépendantes :
 – le nombre de cycles horloge nécessaires à l'exécution du programme (N_X);
 – le nombre de cycles horloge à attendre une donnée de la mémoire (N_W).
 Donc on a : $T_{CPU} = (N_X + N_W) * T_{cycle}$.

1. Exprimer N_W en fonction de :
 - N_{at} , le nombre total d'accès mémoire,
 - M_{rate} , le taux d'échec lors d'une requête au cache (Miss Rate),
 - $M_{penalty}$, le coût (en cycles horloge) d'un échec (Miss Penalty).

N.B. : On suppose que le nombre de cycles horloge nécessaire à la recherche dans le cache est inclus dans N_X .

Réponse _____
 $N_W = N_{at} * M_{rate} * M_{penalty}$.



2. En déduire l'expression de T_{CPU} en fonction de :
 - IC , le nombre d'instructions exécutées;
 - CPI_X , le nombre moyen de cycles pour exécuter une instruction;
 - N_a , le nombre moyen d'accès au cache par instruction;
 - M_{rate} ;
 - $M_{penalty}$.

Réponse _____
 $T_{CPU} = IC * (CPI_X + N_a * M_{rate} * M_{penalty}) * T_{cycle}$.



3. Soient les caractéristiques suivantes :
 - Temps d'un accès mémoire sans passer par le cache = 2 cycles horloge;
 - $M_{penalty} = 6$ cycles horloge;
 - $M_{rate} = 11\%$;
 - $CPI_X = 8.5$ cycles horloge;
 - $N_a = 3.0$ accès mémoire par instruction.

Quel est l'impact sur les performances de l'utilisation d'un tel cache?

Réponse _____

$$T_{CPUaveccache} = IC * (8.5 + 3.0 * 0.11 * 6) * T_{cycle},$$

$$\text{soit } T_{CPUaveccache} = IC * 10.5 * T_{cycle}.$$

Si on suppose que l'on n'a pas de cache et qu'un accès mémoire nécessite 2 cycles horloge d'attente, alors on obtient :

$$T_{CPUsanscache} = IC * (8.5 + 3.0 * 2) * T_{cycle} = IC * 14.5 * T_{cycle}.$$

Et le temps CPU est donc accru de 40% environ (38).



Exercice 2 : Caches et espace mémoire

Afin d'étudier les coûts relatifs de différents modèles de mémoire cache, nous considérons dans cet exercice 3 modèles de mémoire cache n-way-associative, composés de N, 2N et 4N ensembles. Ces ensembles contiennent respectivement 8, 4 et

2 blocs (La taille du cache est donc la même dans les trois modèles.). Supposons pour la suite que $N=128$ et que les adresses sont représentées sur 24 bits et que chaque bloc contient 4 octets.

1. Indépendamment de l'implantation matérielle, indiquer le plus petit nombre de bits nécessaire au contrôle de chaque ensemble pour implanter une stratégie FIFO pour le remplacement des blocs. Réponse _____

- Pour un ensemble à 2 blocs, il suffit de connaître le premier bloc entré. Comme le numéro de bloc dans l'ensemble tient sur un bit (0 ou 1), un unique bit suffit pour gérer un ensemble à 2 blocs.
 - Pour un ensemble à 4 blocs, il suffit de connaître les trois premiers blocs entrés. Comme le numéro de bloc dans l'ensemble tient sur deux bits (00, 01, 10 ou 11), $3*2 = 6$ bits suffisent pour gérer un ensemble à 4 blocs.
 - Pour un ensemble à 8 blocs, il suffit de connaître les sept premiers blocs entrés. Comme le numéro de bloc dans l'ensemble tient sur trois bits (000, 001, 010...111), $7*3 = 21$ bits suffisent pour gérer un ensemble à 8 blocs.
- Remarque* : Il est évident que, si ce nombre de bits suffit à la gestion de tels ensembles, il ne garantit sûrement pas son efficacité.



2. Donner l'expression permettant de déterminer la taille totale du cache.

Réponse _____

$$Taille = Nb_{ensembles} * (Nb_{bitscontrole} + Nb_{Blocs} * Taille_{Bloc})$$



3. Estimer le nombre total de bits nécessaire pour chaque modèle.

Réponse _____

Étudions d'abord comment calculer la taille d'un bloc. Un bloc contient les bits nécessaires au stockage de la donnée ($4*8 = 32$ dans notre cas), plus un bit de validité pour savoir si le contenu d'un bloc est valide à un instant donné et enfin le nombre de bits utiles pour le codage de son Tag. Le tag doit contenir suffisamment de bits pour identifier les blocs susceptibles de se ranger dans l'ensemble. Considérons une adresse sur 24 bits :

Tag	Set	Offset
		2 bits

La taille de bloc est de 4 octets pour tous les modèles, donc les 2 bits de droite suffisent pour constituer l'offset. Ce qui va évoluer, c'est la répartition des 22 bits restants entre Tag et Set, l'un repérant le bloc et l'autre l'ensemble susceptible de le contenir. En effet le choix de l'ensemble se fait par un modulo sur ces 22 bits.

Le numéro d'un ensemble est codé sur p bits si le nombre d'ensembles est 2^p . Donc pour le cas du cache à $4N$ ensembles de 2 blocs, il faut 9 bits pour repérer l'ensemble correspondant à un bloc car $2^9 = 512 = 4*128$. Donc le Tag sera composé des $22 - 9 = 13$ bits restants. Dans le cas du cache à $2N$ ensembles, il faut 8 bits pour le Set donc 14 pour le tag. Et dans le cas du cache à N ensembles on a 15 bits de tag.

- Cas du cache à N ensembles : $T_{B1} = 32 + 1 + 15 = 48$ bits, donc :
 $T_{cache1} = 128 * (21 + 8 * 48) = 51840bits$
- Cas du cache à $2N$ ensembles : $T_{B2} = 32 + 1 + 14 = 47$ bits, donc :
 $T_{cache2} = 256 * (6 + 4 * 47) = 49664bits$
- Cas du cache à $4N$ ensembles : $T_{B3} = 32 + 1 + 13 = 46$ bits, donc :
 $T_{cache3} = 512 * (1 + 2 * 46) = 47616bits$

-
4. En supposant que le coût de la mémoire cache est uniquement fonction du nombre de bits, calculer les coûts relatifs des caches par rapport au modèle à 4N ensembles.

Réponse _____

$$\frac{T_{cache1}}{T_{cache3}} = 1.1, \text{ donc } 10\% \text{ plus cher.}$$

$$\frac{T_{cache2}}{T_{cache3}} = 1.05, \text{ donc } 5\% \text{ plus cher.}$$

Exercice 3 : Défauts de page et temps d'exécution

Si une instruction représente 1 micro-seconde de temps d'exécution, et qu'un défaut de page nécessite n micro-secondes supplémentaires, donner l'expression permettant de calculer le temps d'exécution moyen d'une instruction en supposant qu'un défaut de page survient toutes les k instructions.

Réponse _____

$$T_{moyen} = (1 + n/k) \text{ micro-secondes}$$

Exercice 4 : Défauts de page et temps d'exécution (suite)

Le nombre d'instructions exécutées entre deux défauts de pages est proportionnel au nombre de pages réelles allouées à un programme (sauf exceptions...). De ce fait, si le nombre de cadres (pages réelles) est doublé, l'intervalle moyen entre deux défauts de page est lui aussi doublé.

Supposons qu'une instruction normale s'exécute en 1 micro-secondes, mais qu'un défaut de page porte ce temps à 2001 micro-secondes. Soit, de plus, un programme qui s'exécute en 60 secondes durant lesquelles se produisent 15000 défauts de page.

1. Quel est l'intervalle moyen entre deux défauts de page?

Réponse _____

2000 instructions.

2. Combien de temps le programme nécessitera-t-il pour s'exécuter si on double le nombre de cadres?

Réponse _____

45 secondes

Exercice 5 : Stratégie de remplacement

Considérons une machine possédant une mémoire organisée en 4 pages. Soient les données suivantes :

Page	dernier chargement	dernière référence	R	M
0	126	279	0	0
1	230	260	1	0
2	120	272	1	1
3	160	280	1	1

Quelle page sera remplacée en utilisant la stratégie :

1. FIFO ;
2. LRU ;
3. NRU

Réponse _____

1. 2 ;
2. 1 ;
3. 0



Rappel sur la stratégie NRU :

On associe deux bits à chaque page ; R et M. Ces bits sont initialisés à zéro lors du chargement de la page. Ensuite R prend la valeur 1 lors d'un accès (lecture ou écriture) à la page et M prend la valeur 1 uniquement en cas d'écriture.

Périodiquement, le bit R est remis à 0 à chaque période d'horloge (toutes les 20 ms par exemple). On peut ainsi aisément distinguer les pages utilisées récemment des autres. Il y a 4 états possibles pour une page.

- état 1 : $R = M = 0$;
- état 2 : $R = 0, M = 1$;
- état 3 : $R = 1, M = 0$;
- état 4 : $R = M = 1$.

La stratégie NRU consiste à remplacer la page ayant le plus petit numéro d'état. Si plusieurs pages remplissent les mêmes conditions, le choix de l'une d'elles est aléatoire.

Cette méthode est facile à implanter et efficace ; les performances sont souvent intéressantes.

Exercice 6 : Stratégie de remplacement (suite)

Considérons une machine utilisant 4 pages réelles. Un processus utilise 8 pages et les référence dans l'ordre suivant : 0 1 7 2 3 2 7 1 0 3.

En considérant les 4 pages réelles initialement libres, combien de défauts de pages surviendront si l'on utilise :

1. la stratégie FIFO ;
2. la stratégie LRU.

Réponse _____

1. FIFO : 4 défauts pour les 4 premières pages, puis deux autres en cours = 6 défauts.
2. LRU : 4 défauts pour les 4 premières pages, puis trois autres = 7 défauts.



Remarque : Anomalie de Belady.

Alors qu'intuitivement il semble que le nombre de défauts de page décroisse avec l'augmentation du nombre de pages réelle, il existe des contre-exemples dont le suivant :

On utilise une stratégie FIFO avec 3 et 4 pages et la séquence de références suivante :

0 1 2 3 0 1 4 0 1 2 3 4.

On constate 9 défauts pour le cas où l'on dispose de trois pages réelles et 10 défauts pour 4 pages !