Rapport de TP 4MMAOD : Patch optimal entre deux fichiers

BOYER Quentin (groupe étudiant$_1$)

6 novembre 2019

1 Graphe de dépendances (1 point)

Dessinez le graphe de dépendances des appels.

L'algorithme est alors de trouver le plus court chemin depuis un point $(i, j) tq i = 0 et/ou j = 0$. Cela donne donc en fonction du chemin le patch optimal

2 Principe de notre programme (1 point)

Pour pouvoir trouver le plus court chemin je commence par creer une liste de tout les chemins au depart d'un des bords avec $i = 0$ ou $j = 0$. Ensuite on fait un tas avec ces chemins. A chaque iteration suivante on prends le chemin le plus court, on l'enleve du tas. Grace a ce chemin on produit trois chemin en incrementant i, j puis i et j. Si un chemin $c1$ arrive sur une case sur laquelle un autre chemin est deja passe, on sait par notre invariant qu'il existe un autre moyen d'arriver a ce point, et qu'il est plus court. On peut donc supprimer $c1$, puisque il sera pire que le chemin qui est deja passe par la. Sinon on ajoute le chemin dans notre tas

On s'arrete des que le plus court chemin arrive sur la case (m, n). On a alors trouve le plus court chemin pour arriver au patch optimal

De plus pour calculer les couts on stock non pas un tableau des lignes mais un tableau qui donne l'indice de la fin de la ligne i, cela permet ainsi d'avoir un cout faible pour trouver la longeure d'une ligne

3 Analyse du coût théorique (3 points)

On commence par calculer la liste des indices de fin de ligne, ca coute $m + n$ dans tout les cas.

3.1 Nombre d'opérations en pire cas :

Justification : Dans le pire cas il faut calculer tout les chemins possibles, c'est a dire passer par chaque case exactement une fois, cela fait donc $m \times n$ iterations. A chaque iteration on produit environ 1 chemin, puisque si rentre en collision avec les autres chemins, au total on est donc en $\Theta(m \times n)$

3.2 Place mémoire requise :

Justification : Dans chaque case on doit stocker de quelle cases viens t'on, pour pouvoir refaire le chemin dans le sens inverse. Cela fait donc dans le pire cas autant de place requise que de chemins, soit $\Theta(m \times n)$.

3.3 Nombre de défauts de cache sur le modèle CO :

Justification : Pour les defauts de cache, dans le pire cas a chaque fois qu'on essaye d'acceder a une case elle n'est pas dans le cache, si par exemple les chemins les plus courts sont dans des parties totalement differentes de la grille. Cela coute donc $\Theta(m \times n)$

3.4 Quand arrive le pire cas

Si les fichiers ne sont pas tres differents on va rester majoritairement autour de la diagonale. Cela change donc beacoup la donne, puisque on va faire des defauts de cache que lorsque on a trop avance dans la diagonale, on va calculer et garder en memoire qu'environ la diagonale.

Si $m \approx n$ on a la diagonale de longeure m, cela donne donc un algorithme quasiment lineaire en la taille des fichiers, et m/L defauts de cache environ

4 Compte rendu d'expérimentation (2 points)

4.1 Conditions expérimentaless

Décrire les conditions permettant la reproductibilité des mesures : on demande la description de la machine et la méthode utilisée pour mesurer le temps.

4.1.1 Description synthétique de la machine :

indiquer ici le processeur et sa fréquence, la mémoire, le système d'exploitation. Préciser aussi si la machine était monopolisée pour un test, ou notamment si d'autres processus ou utilisateurs étaient en cours d'exécution.

4.1.2 Méthode utilisée pour les mesures de temps :

préciser ici comment les mesures de temps ont été effectuées (fonction appelée) et l'unité de temps ; en particulier, préciser comment les 5 exécutions pour chaque test ont été faites (par exemple si le même test est fait 5 fois de suite, ou si les tests sont alternés entre les mesures, ou exécutés en concurrence etc).

4.2 Mesures expérimentales

Compléter le tableau suivant par les temps d'exécution mesurés pour chacun des 6 benchmarks imposés (temps minimum, maximum et moyen sur 5 exécutions)

	coût du patch	temps min	temps max	temps moyen
benchmark_00				
benchmark_01				
benchmark_02				
benchmark_03				
benchmark_04				

FIGURE 1 – Mesures des temps minimum, maximum et moyen de 5 exécutions pour les benchmarks.

4.3 Analyse des résultats expérimentaux

Donner une réponse justifiée à la question : les temps mesurés correspondent ils à votre analyse théorique (nombre d'opérations et défauts de cache) ?

5 Coût du patch en octet (1 point)

Quelle méthode utiliseriez vous pour résoudre le problème si le coût d'un patch était défini comme sa taille en nombre d'octets (i.e. taille du fichier patch). On ne demande pas de programme ni d'algorithme, mais juste de préciser le principe de la résolution choisie (parmi celles vues en cours) ; on précisera soit les équations de base pour la résolution, soit les modifications à apporter à votre programme s'il peut être adapté à cette fonction de coût.

6 Bonus : Pourrait-on utiliser une technique de blocking ? (1 point)

Si oui, quel serait asymptotiquement le nombre de défauts de cache avec un cache de taille Z chargé par par lignes de taille L ?