Questionnaire TP AOD à compléter et rendre sur teide avant le 13/11/2014 - 23h00

Binôme (NOM_1 Prénom_1 – NOM_2 Prénom_2) : Raphaël Laguerre, Alexandre Rupp

1. QCM: A quel découpage correspond le coût $\phi(i)$?
   ☐ A/ $m_0, \ldots, m_n$ ☐ B/ $m_0, \ldots, m_{i-1}$ ☐ C/ $m_0, \ldots, m_i$
   ☑ D/ $m_i, \ldots, m_n$ ☐ E/ $m_{i+1}, \ldots, m_n$ ☐ F/ aucune des réponses A..E

2. QCM: Que vaut $C^\star$ du paragraphe?
   ☑ A/ $\phi(0)$ ☐ B/ $\phi(1)$ ☐ C/ $\phi(n-1)$ ☐ D/ $\phi(n)$ ☐ E/ aucune des réponses A..D

3. Analyser le coût de votre implémentation en répondant aux questions ci-dessous. Soit un paragraphe comportant t caractères (imprimables et séparateurs) formant n mots ayant chacun au plus m caractères.

   (a) Analyser asymptotiquement (notation O ou Θ):

      i. le nombre d'opérations effectuées (travail) $= 0(init+phi+impression) = O((3t+n)+(t+\frac{n(n+1)}{2})+(t)) = O(n^2)$

      ii. la profondeur (temps parallèle minimal ou chemin critique) $= O(n^2)$ (Calcul des Phi, chacun dépend des précédents).

   (b) Analyser précisément en donnant un équivalent (donc pas en notation O ou Θ):

      i. l'espace mémoire requis par votre programme$= O(2n + t + n)$ (Phi(coupure + cout), tableau de mots, tableau de sommes des tailles de mots)

      ii. le nombre de défauts de localité (cache) $= O(\frac{t}{B} + \frac{n}{B} + \frac{n(n+1)}{2B})$ (tableau de mot, tableau de tailles, phi)

4. Pour un fichier de longueur s caractères ($s \gg Z$) dont le plus long paragraphe contient t caractères, donner:

   (a) le nombre d'opérations effectuées (notation O ou Θ) $= 0(s)$

   (b) la profondeur (temps parallèle minimal ou chemin critique) $= O(t)$

   (c) le nombre de défauts de cache $= O(\frac{3s}{B}) = O(\frac{s}{B})$ (chargement du fichier + tableau de mots + tableau de somme des tailles de mots).

5. Expérimenter votre programme sur le fichier F=`ALaRechercheDuTempsPerdu.txt` en sortie texte (pas pdf). Mesurer sur 10 exécutions le temps CPU et le nombre de défauts de localité et indiquer les valeurs moyennes, minimales et maximales des 10 exécutions.
   Indiquer la machine (nom si Ensimag; processeur et fréquence sinon) : pcserveur

Valeur mesurée	Minimum	Maximum	Moyenne
$C^\star(F)$	1268482	1268482	1268482
Temps CPU	0.79s	0.919s	0.8318s
Défaut de localité	0.1%	0.1%	0.1%

6. La fonction `mmap` projette un fichier ou un périphérique en mémoire. Ainsi on peut parcourir le fichier à partir d'un pointeur; on accède les octets en déréférençant le pointeur, comme un tableau. Les octets du fichier sont chargés en mémoire par blocs (pages) et seulement lorsqu'ils sont accédés. En utilisant `mmap` (et `unmmap`), peut-on diminuer le nombre de défauts de localité? ☑ A/ Oui ☐ B/ Non, il reste le même ☐ C/ Non, il augmente.
   Si A ou B, indiquer le nombre de défauts de localité en fonction de Z et $B = O(\frac{2s}{B})$ (comme avant - défauts de cache liés au tableau de mot)
   Justifier brièvement (3 lignes max) :
   Lorsqu'on utilise mmap, ce dernier nous fournit un pointeur permettant d'accéder directement aux données du fichier. On s'épargne donc les défauts de cache obtenus lors de la recopie du contenu du fichier vers un tableau. Lorsqu'on a besoin d'un mot, on accède dirèctement à la zone couplée.

7. En UTF8, la césure douce est représentée par le caractère SHY (*Soft Hyphen*) de code hexadécimal `0xAD`, affiché par le signe '-'; en LaTeXpar la séquence '\-'. Votre programme actuel ne prenant pas en compte les césures douces, il les affiche comme des '-'. Pour un plus bel affichage, on considère ici une extension prenant en compte les césures (que l'on ne demande pas de programmer). Lorsqu'on insère un retour à la ligne entre deux mots séparés par une césure douce, un caractère moins '-' est ajouté à la fin du premier mot. Mais si les deux mots sont sur la même ligne, alors ils sont affichés collés l'un à l'autre. La fonction de pénalité est modifiée pour prendre en compte, outre la taille d'espace, la présence d'une césure en fin de ligne. Soit $T(f)$ le temps pris par votre programme actuel sur un fichier f (comportant des césures douces qui ne sont pas prises en compte).
   Question : expliquer clairement et brièvement comment vous adapteriez votre programme pour que le nombre d'opérations $T'(f)$ du nouveau programme étendu pour les césures vérifie : $T'(f) = O(1) \cdot T(f)$.
   Réponse (10 lignes max) : Les mots contenant une césure douce sont stockés dans le tableau de mot comme si il n'en contenaient pas.
   À coté du tableau de mot on conserve un tableau d'integer de taille <nombre de mots du paragraphe> qui conserve pour chaque mot la position de la coupure, ou -1 si le mot n'est pas coupé.
   On utilise également un tableau de booleens de taille <nombre de mots du paragraphe> qui stocke si le mot a effectivement été coupé ou non.

Dans la fonction Phi (“justifypar” dans notre code), on prend la décision de couper oun non le mot, et on met à jour le tableau de booleens en conséquence.
On doit également mettre à jour la fonction “L” qui donne la somme des tailles des mots car la taille est différente si il y a coupure de mot. Dans L on regardera simplement si le dernier mot de la ligne doit être coupé ou non, si oui la taille est plus grande de 1).
Lors du drawparagraphe, à l’endroit de la coupure (= fin de ligne), on vérifie si le dernier mot de la ligne doit être coupé ou non (tableau de boolen), si il doit l’être, on rajoute un “hyphen” à l’endroit de la coupure (cf tableau de position des coupures).