Université de Lille 1 SCIENCES ET TECHNOLOGIES

Info 204 - ASD

16 mai 2014
Licence STS - Semestre 4

DS2 - documents de cours, TD, TP autorisés

Note : lorsque, dans une question, est demandée la complexité, c'est une fonction de la taille de la donnée qui est attendue. Lorsque c'est le comportement asymptotique, la réponse attendue est soit une notation $\mathcal{O}$, soit une notation Θ soit une notation Ω.

Vous pourrez utiliser toute fonction vue en cours, TD, TP à la condition expresse de spécifier correctement les entrées et les sorties.

Exercice 1 : Parcours d'arbre [3 points]

Compétence évaluée : utliliser une structure de donnée arborescente.

Avec la définition de type suivante :

```
type arbre = Vide | Cons of noeud
and noeud = {
  valeur : int;
  fils   : arbre list; /* la liste ordonnee des fils */
}

let le_noeud a =
  match a with
   | Vide -> raise ArbreVide
   | Cons n -> n
```

Ecrire une fonction en CAML `plus_profonde : arbre -> int` qui retourne la valeur associée à la feuille de profondeur maximale (si il y a plusieurs feuilles à cette profondeur maximale, on retournera celle concernant la feuille la plus à droite).

Vous pouvez (devez) écrire des sous-fonctions.

Exercice 2 : Structure de données[7 points]

Compétence évaluée : réinvestir les connaissances du cours pour répondre à un nouveau problème proche d'un problème connu.

Un utilisateur souhaite disposer d'une structure de donnée lui permettant :
- d'ajouter au fur et à mesure de nouveaux entiers à cet ensemble,
- d'être capable d'en extraire le minimum.

Un exemple d'utilisation pourrait être le suivant (`s` est la structure de données) :

```
# ajouter s 2;;
# ajouter s 5;;
# ajouter s 3;;
# minimum s;;
2
# ajouter s 1;;
# minimum s;;
1
# minimum s;;
3
```

On souhaite que les opérations d'ajout (procédure `ajouter`) et d'extraction (fonction `minimum`) s'exécutent en temps $\mathcal{O}(\log n)$ où n est le nombre d'élements contenus dans la structure au moment de l'ajout ou de l'extraction. On supposera qu'on n'ajoute pas deux fois le même entier.

Q 2.1 Expliquer la structure de données que vous allez utiliser.

Q 2.2 La structure de données sera nommée `sd`. Donner la définition des types nécessaires pour définir `sd` en CAML.

Q 2.3 Décrire en français ou pseudocode l'algorithme de la procédure d'ajout.

Q 2.4 Décrire en français ou pseudocode l'algorithme de la fonction d'extraction du minimum.

Q 2.5 Justifier de la complexité en temps en $\mathcal{O}(\log n)$ de la procédure d'ajout.

Q 2.6 Justifier de la complexité en temps en $\mathcal{O}(\log n)$ de la fonction d'extraction.

Exercice 3 : Dérécursivation [5 points]

Compétence évaluée : réfléchir sur un problème pour proposer une solution algorithmique efficace.

On fait face à un problème de calcul modélisé par l'équation de récurrence suivante :

$$f(i,j) = \begin{cases} i+j & \text{si } i=0 \text{ ou } j=0 \\ \max_{0\leq k\leq l<\min(i,j)} f(k,l) & \text{sinon} \end{cases}$$

Q 3.1 Ecrire une fonction **récursive** en CAML permettant le calcul de f.

Q 3.2 Cette fonction est-elle récursive terminale ?

En gardant une version récursive, on souhaite réaliser une modification de la fonction pour limiter le nombre d'appels récursifs tout en occupant un espace mémoire le plus faible possible. Pour répondre à ce problème :

Q 3.3 Expliquer comment réduire le nombre d'appels récursifs.

Q 3.4 Expliquer comment conserver un espace mémoire faible.

Q 3.5 Quelle est la complexité en espace de la nouvelle version ? Justifier en calculant exactement le nombre de valeurs calculées.

Exercice 4 : Programmation dynamique [5 points]

Compétence évaluée : appliquer les méthodologies apprises en cours.

On s'intéresse ici au problème du calcul de la longueur de la plus longue sous-séquence croissante d'un tableau d'entiers définie comme étant le plus grand nombre d'entiers $x_1, \ldots, x_m$ du tableau tels que :

— $x_i \leq x_{i+1} \quad \forall i \in 1..m-1$ et

— $p(x_i) < p(x_{i+1}) \quad \forall i \in 1..m-1$ où $p(x_i)$ est l'indice de x_i dans le tableau.

Par exemple, avec le tableau `[|3;1;2;6;5;7;9;8|]`, la longueur de la plus longue sous-séquence croissante est 5 (il y a plusieurs sous-séquences correspondantes : 1,2,6,7,9 ; 1,2,6,7,8 ; 1,2,5,7,9 ; 1,2,5,7,8 ; 1,2,3,7,9 ; 1,2,3,7,8).

Si on note $L(i)$ la longueur de la plus longue sous-séquence croissante incluant l'entier en position i dans le tableau, alors le problème s'exprime récursivement ainsi :

$$L(i) = \begin{cases} 1 + \max_{0\leq k<i} L(k) & \text{si il existe } k \text{ tel que } t.(k) \leq t.(i) \\ 1 & \text{sinon} \end{cases}$$

Pour trouver la solution au problème initial, il suffit de rechercher le maximum parmi les L. On supposera disposer du tableau t.

Q 4.1 Donner les valeurs de $L(0), L(1), L(2)$ pour l'exemple.

Q 4.2 Donner les instructions CAML permettant de créer (pas remplir) la table de programmation dynamique permettant le calcul de L.

Q 4.3 Donner les instructions CAML permettant d'initialiser la table L.

Q 4.4 Donner les instructions CAML permettant de remplir la table L.

Q 4.5 Donner les instructions CAML permettant d'obtenir le résultat.

Q 4.6 Quelle est la complexité en espace ? Justifier.

Q 4.7 Quelle est la complexité en temps ? Justifier.