G. HUNAULT
Licence MASS

Angers, 2003

Mini-Projet en Algorithmique et Programmation :

Propriétés métriques de fonctions

1. Position du problème

Soit f une fonction définie sur $E \times E$ et q_f la fonction définie sur E^4 par $q_f(a,b,c,d) = f(a,b) + f(c,d)$.

On définit les propriétés suivantes :

nom	f est	si et seulement si f vérifie (pour tous x, y, z et t)
C1	d-nulle	$f(x,x) = 0$
C2	symétrique	$f(x,y) = f(y,x)$
C3	d-triangulaire	$f(x,y) \leqslant f(x,z) + f(z,y)$
C4	séparée	$f(x,y) = 0 \Rightarrow y = x$
C5	sous-maximale	$f(x,y) \leqslant max(\ f(x,z), f(y,z)\ )$
C6	bunemanienne	$q_f(x,y,z,t) \leqslant max(\ q_f(x,z,y,t), q_f(x,t,y,z)\ )$
C7	tripunctive	$f(x,z) = f(y,z) = c \Rightarrow f(x,y) \leqslant c$

Lorsque plusieurs propriétés sont vérifiées simultanément, on donne à f des noms particuliers, à savoir :

nom	conditions	remarque
indice de dissimilarité	C1 C2	
pseudo-métrique	C1 C2 C3	on dit aussi *écart*
métrique	C1 C2 C3 C4	on dit aussi *distance*
ultramétrique	C1 C2 C3 C5	ce qui implique C4
métrique d'arbres	C1 C2 C3 C4 C6	on dit aussi *métrique additive*

Le but du projet est de tester ces conditions pour quelques fonctions définies par des fichiers-textes à l'aide d'un programme Rexx et des fonctions Apl.

2. Travail demandé

2.1) On décide de définir la fonction f à travers les valeurs $f(i,j)$ pour i et j valant 1, 2, 3... Pour cela on utilise un fichier d'extension .f ou .F qui contient n lignes telles que chaque ligne contient exactement n mots, la valeur $f(i,j)$ correspondant au mot numéro j de la ligne i.

Ecrire l'algorithme du module `litFonc` qui admet un paramètre à savoir le nom du fichier, qui lit ce fichier et remplit au passage le tableau TF défini par $TF[i,j] = f(i,j)$; on prendra comme valeur pour n le nombre de mots de la première ligne du fichier. On utilisera bien sûr exclusivement le vocabulaire algorithme vu en cours et accepté par *Galg*.

2.2) Ecrire ensuite l'algorithme du module `testeFonc` qui teste si la fonction f induite par le fichier lu vérifie une ou plusieurs des propriétés C_i. On pourra définir d'autres modules pour faciliter la lecture de l'algorithme.

2.3) Ecrire enfin un algorithme principal qui demande un nom de fichier, le passe en paramètre à `litFonc` puis qui utilise à `testeFonc` pour caractériser la fonction.

2.4) Trouver l'origine de ce problème, son ou ses auteurs et leur biographie succinte, les références et les théorèmes liés à ce problème. On pourra se contenter de références bibliographiques et de pages *Web*.

On s'intéresse maintenant à la traduction des algorithmes en Rexx et en Apl.

2.5) Traduire tous les algorithmes en Rexx. On utilisera pour tester le programme en ligne de commande les 5 fichiers suivants :

fnc1.F	*fnc2.F*	*fnc3.F*	*fnc4.F*	*fnc5.F*
0 8 8 1 8 0 2 2 8 2 0 3 0 2 3 0	0 8 8 1 8 0 2 2 8 2 0 3 1 2 3 0	0 2 4 1 2 0 2 2 4 2 0 3 1 2 3 0	0 11 10 9 11 0 3 12 10 3 0 11 9 12 11 0	0 8 8 14 8 0 2 14 8 2 0 14 14 14 14 0

Ces fichiers sont disponibles dans les salles informatiques, sous la forme d'une archive nommée FONC_ALS03.ZIP située sur le disque K: répertoire stat_ad soit le fichier K:\stat_ad\FONC_ALS03.ZIP.

2.6) Traduire tous les algorithmes en Apl ; on remplacera la lecture de fichier par l'utilisation d'une matrice de la zone de travail. La fonction principale utilisera cette matrice comme paramètre.

Pour tester les fonctions associées, on pourra utiliser les matrices définies dans la zone de travail FONC_ALS.W3 située sur le disque K: répertoire stat_ad soit le fichier K:\stat_ad\FONC_ALS.W3.

3. Consignes pour le mini-projet

Ce miniprojet est à rédiger seul(e) même si vous travaillez à plusieurs pour la programmation ou les recherches bibliographiques. Vous devez envoyer en attaché par mail un fichier .zip qui contient tout votre travail, à savoir :

- un ou plusieurs fichiers algorithmes .alg avec leur analyse par *Galg* (fichiers .lst et .lvm),
- une zone de travail Apl (fichier .ws = "*Workspace*" ou .apl ou .w3), avec les fonctions et si possible aucune variable sauf un texte d'explication,
- un fichier .rex texte du programme Rexx qui fonctionne sous *Dos/Windows* comme sous *Unix/Linux* avec l'interpréteur *regina*,
- un document .doc au sens de *Word* pour *Windows* (de préférence au format RTF), un fichier .tex pour LateX, un fichier *Acrobat* au format PDF ou enfin un fichier PostScript qui contiendra les la documentation des algorithmes, les références du problème, les résultats des progammes et et vos commentaires concernant l'analyse, la programmation et la traduction.

Ces fichiers auront pour nom vos initiales : Jomes Band fournira donc JB.WS, JB.REX et JB.DOC le tout dans JB.ZIP ; par contre Nona Higane rendra NH.WS, NH.REX et NH.DOC, le tout dans NH.ZIP. Lors de l'archivage, vous ne mettrez aucun sous-répertoire.

Ce travail de recherche, programmation et rédaction est à rendre pour le lundi 8 décembre 2003 à 11h.

Aucun retard ne sera toléré.

Pour envoyer les fichiers, il faut impérativement utiliser la page *Web*

`http://www.info.univ-angers.fr/pub/gh/wals/envoiproj.php`

4. Remarques concernant la rédaction

Il vaut mieux rendre un travail incomplet que de ne rien rendre. La qualité des algorithmes, en particulier leur simplicité et leur lisibilité sera appréciée. Indiquez clairement vos choix : si un terme vous parait ambigu, précisez sa définition pour vous. Justifiez vos modes de programmation ; récursivité ou linéarité, appel d'opérateurs, de fonctions prédéfinies ou écriture de boucles explicites. Nommez distinctement vos variables (en particulier, vous utiliserez systématiquement au moins 4 lettres par identificateur). N'hésitez pas à commenter sans dépasser toutefois un commentaire toutes les 4 lignes en moyenne (sauf pour l'en-tête).

Vous n'êtes pas obligé(e) de tout tester dans tous les sens. Effectuez donc des tests élémentaires. L'affichage des résultats à l'écran devra être très propre, très soigné. Ainsi en cas d'affichage numérique multi-lignes, on mettra bien les unités sous les unités, les dizaines sous les dizaines etc. Soignez aussi la rédaction, l'orthographe (utilisez des correcteurs orthographiques et grammaticaux).

La note sera principalement fonction de la qualité de l'analyse du travail à effectuer, de l'exposé de la méthode et de la clarté de la rédaction. La valeur attribuée au programme ne représentera que 25 % de la note ; aucune interface n'est demandée.

Aucun langage autre que cité ne devra être utilisé, aucun traitement de textes autre que cité ne devra être utilisé. Tout fichier qui ne correspondra pas aux initiales de l'élève ne sera pas lu. En particulier, les fichiers nommés minip, algo, proj, proj, projet, doc ne seront pas lus.

Un programme même incomplet mais commenté et bien indenté sera au moins autant apprécié qu'un programme qui "marche" mais sans commentaire et illisible quand aux structures de flot (boucles, tests etc.).

5. Références

Le logiciel *APL* pour *Dos/Windows* (ancienne version 'spéciale Université d'Angers') peut être récupéré à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/getapl.htm`

on y trouvera aussi comment récupérer un *APL* gratuit pour *Linux*.

Le logiciel *REGINA (REXX)* pour *Dos/Windows* et *Linux* peut être récupéré à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/getrex.htm`

Plus généralement, tous les documents pour le cours Algorithmique, Langages et Styles de Programmation dont un manuel élémentaire d'*Algorithmiques raisonnée*s, les consignes pour le projet et ce texte sont disponibles à l'adresse

`http://www.info.univ-angers.fr/pub/gh/wals/`

Pour être sur(e) que les algorithmes sont correctement écrits, on utilisera le programme galg et on rendra le listage des instructions (fichier `.lst` plutôt que l'algorithme initial `.alg`) ainsi que le fichier `.lvm` des variables, tableaux et modules. Le programme galg est disponible à l'adresse

`http://www.info.univ-angers.fr/pub/gh/Galg/`

De plus, il est possible de tester si le projet a été correctement reçu grâce à la page

`http://www.info.univ-angers.fr/pub/gh/wstat/zipdirmpm.php3`