

Oraux blancs MPI

type CCINP

Florian Bourse

Exercice A.

cf annexe pour un rappel des règles de la déduction naturelle
Nous avons les faits suivants :

1. Si Informatix réussit sa preuve, il sera content.
2. S'il pleut, Informatix restera chez lui.
3. Si Informatix ne reste pas chez lui, alors il sera content.
4. Chez lui, Informatix s'entraîne.
5. Informatix réussira sa preuve s'il s'entraîne.

Nous posons les variables propositionnelles suivantes :

π : Informatix réussit sa preuve ;

C : Informatix sera content ;

P : il pleut ;

R : Informatix reste chez lui ;

E : Informatix s'entraîne.

1. Formaliser les 5 faits listés à l'aide de formules logiques propositionnelles.
2. Montrer, en déduction naturelle, qu'Informatix sera content.
3. Peut-on montrer qu'Informatix réussira sa preuve ? justifier.

Exercice B. L'exercice suivant est à traiter dans le langage C

Le code fourni peut être compilé avec la commande

```
gcc *.c -o test
```

puis exécuté avec la commande

```
./test
```

Dans cet exercice, on s'intéresse à la vérification de produits matriciels utilisant des flottants.

On représentera les matrices par le type `double **`.

Le problème de décision qui nous intéresse est le problème suivant MAT-PROD :

Instance : Trois matrices A , B , et C de dimensions respectives $n_1 \times n_2$, $n_2 \times n_3$ et $n_1 \times n_3$ pour trois entiers n_1 , n_2 et n_3 .

Solution : V si et seulement si $AB = C$.

1. Le fichier fournit implémente 4 variables flottantes : $a = 2^{60}$, $b = a - 1$, $c = b - a$ et $d = c + 1$. Quelle est la valeur de d et comment expliquer ce résultat ?
Pour la suite, on se fixera une précision de 0.001 en dessous de laquelle on considèrera qu'on ne peut pas savoir si le nombre est nul ou non.
2. Écrire une fonction `double **new_matrix(int n1, int n2)` qui prend en entrée un entier n et qui crée et renvoie une nouvelle matrice de dimension $n_1 \times n_2$.
3. Écrire une fonction `double **prod(double **A, double **B, int n1, int n2, int n3)` qui calcule le produit C des matrices A et B de dimensions respectives $n_1 \times n_2$ et $n_2 \times n_3$:

$$C_{i,j} = \sum_k A_{i,k} B_{k,j}$$

4. En déduire une fonction `verify` qui permet de résoudre le problème de décision MAT-PROD et donner sa complexité.
5. À quelle classe de complexité appartient MAT-PROD ?
6. On souhaite améliorer la complexité de notre test grâce à l'algorithme probabiliste de Freivalds :
On tire un vecteur X uniformément aléatoire de dimension n_3 à coefficients 0 ou 1.
Renvoyer le résultat du test $ABX == CX$.
À quelle famille d'algorithmes probabiliste appartient l'algorithme de Freivalds ?
7. Montrer que l'on peut effectuer le test $ABX == CX$ avec une complexité $O(n_1 n_2 + n_2 n_3 + n_1 n_3)$.
8. Implémenter cet algorithme. On rappelle que l'expression `rand() % 2` permet de tirer aléatoirement entre 0 et 1.
9. Montrer que si $AB = C$, l'algorithme de Freivalds renvoie toujours V et que sinon il renvoie V avec une probabilité inférieure à $\frac{1}{2}$.

A Annexe : Rappel des règles de la déduction naturelle

Les arbres de preuves doivent être effectués à partir de l'ensemble de règles fourni ci-dessous.

$$\begin{array}{c}
\frac{}{\Gamma, p \vdash p}(Ax) \\
\\
\frac{\Gamma \vdash p \quad \Gamma \vdash q}{\Gamma \vdash p \wedge q}(\wedge_i) \quad \frac{\Gamma \vdash p \wedge q}{\Gamma \vdash p}(\wedge_e) \quad \frac{\Gamma \vdash p \wedge q}{\Gamma \vdash q}(\wedge_e) \\
\\
\frac{\Gamma \vdash p}{\Gamma \vdash p \vee q}(\vee_i) \quad \frac{\Gamma \vdash q}{\Gamma \vdash p \vee q}(\vee_i) \quad \frac{\Gamma \vdash p \vee q \quad \Gamma, p \vdash \varphi \quad \Gamma, q \vdash \varphi}{\Gamma \vdash \varphi}(\vee_e) \\
\\
\frac{\Gamma, p \vdash q}{\Gamma \vdash p \rightarrow q}(\rightarrow_i) \quad \frac{\Gamma \vdash p \rightarrow q \quad \Gamma \vdash p}{\Gamma \vdash q}(\rightarrow_e) \\
\\
\frac{\Gamma, p \vdash \perp}{\Gamma \vdash \neg p}(\neg_i) \quad \frac{\Gamma \vdash p \quad \Gamma \vdash \neg p}{\Gamma \vdash \perp}(\neg_e) \\
\\
\frac{\Gamma, p \vdash q \quad \Gamma, \neg p \vdash q}{\Gamma \vdash q}(t.e.) \quad \frac{\Gamma \vdash \perp}{\Gamma \vdash p}(\perp_e)
\end{array}$$