

# Oraux blancs MPI

## type CCINP

Florian Bourse

### Exercice A.

1. Énoncer le théorème de Cook-Levin.
2. On considère le langage  $\mathcal{L}$  des formes normales conjonctives, par exemple  $(x_1 \vee \bar{x}_2 \vee x_3) \wedge (x_2 \vee \bar{x}_1)$   
*On pourra se contenter d'utiliser les symboles  $\wedge$ ,  $\vee$  et  $\bar{x}$ .*
3. Donner une grammaire non-contextuelle qui engendre  $\mathcal{L}$ .
4. Déterminer si  $\mathcal{L}$  est régulier ou non.
5. On considère le problème CNF-TAUT suivant :

**Instance :** Une formule  $\varphi$  sous forme normale conjonctive.

**Solution :**  $V$  si et seulement si  $\varphi$  est une tautologie.

Montrer que CNF-TAUT  $\in P$ .

6. On considère le problème DNF-TAUT suivant :

**Instance :** Une formule  $\varphi$  sous forme normale disjonctive.

**Solution :**  $V$  si et seulement si  $\varphi$  est une tautologie.

Montrer que si DNF-TAUT  $\in P$ , alors  $P = NP$ .

*On pourra utiliser sans le démontrer  $SAT \leq_P CNF-SAT$ .*

### 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

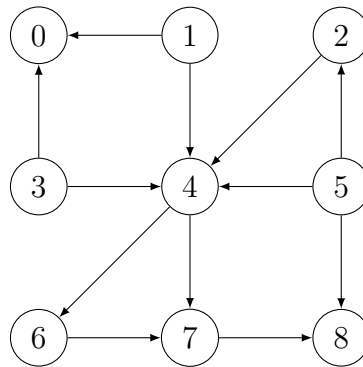
```
./test
```

Dans cet exercice, tous les graphes seront orientés. On représente un graphe orienté  $G = (S, A)$ , avec  $S = \{0, \dots, n-1\}$ , en C par la structure suivante.

```
struct graph_s {  
    int n;  
    int degre[100];  
    int voisins[100][10];  
};
```

L'entier  $n$  correspond au nombre de sommets  $|S|$  du graphe. On suppose que  $n \leq 100$ . Pour  $0 \leq s < n$ , la case `degre[s]` contient le degré sortant  $d^+(s)$ , c'est-à-dire le nombre de successeurs, appelés ici *voisins*, de  $s$ . On suppose que ce degré est toujours inférieur à 10. Pour  $0 \leq s < n$ , la case `voisins[s]` est un tableau contenant, aux indices  $0 \leq i < d^+(s)$ , les voisins du sommet  $s$ . Il s'agit donc d'une représentation par listes d'adjacences où les listes sont représentées par des tableaux en C.

Un programme en C vous est fourni dans lequel le graphe suivant est représenté par la variable `g_exemple`.



1. Compléter le code de la fonction `calcul_degres` permettant de calculer les degrés entrants  $d^-(s)$  de tous les sommets  $s$  et de stocker ces résultats dans un tableau.
2. Calculer la complexité de votre fonction.
3. Rappeler la définition d'un tri topologique et donner une condition nécessaire et suffisante pour qu'il existe.
4. Soit  $s_0, s_1, \dots, s_{n-1}$  un tri topologique de  $G$ . Donner le degré entrant  $d^-(s_0)$  de  $s_0$ .
5. En déduire un algorithme permettant de calculer un tri topologique de  $G$  et l'implémenter.
6. Calculer la complexité de votre fonction en supposant l'existence d'un tri topologique.
7. On considère l'algorithme suivant pour calculer un ensemble de plus court chemins depuis une unique source  $s$  vers tous les autres sommets dans un graphe orienté pondéré par une fonction  $w : A \mapsto \mathbb{R}$  :

Initialiser :  $d[u] \leftarrow \infty$  et  $p[u] \leftarrow \text{nil}$  pour tout  $u \in S$  ; puis  $d[s] \leftarrow 0$   
 Calculer un tri topologique des sommets :  $u_1, u_2, \dots, u_n$   
 Pour tout sommet  $u$  dans l'ordre topologique :  
   Pour tout sommet  $v$  tel que  $(u, v) \in A$  :  
     Si  $d[v] > d[u] + w(u, v)$  alors :  
        $d[v] \leftarrow d[u] + w(u, v)$   
        $p[v] \leftarrow u$

Montrer que cet algorithme est correct et donner sa complexité.