Algorithmique procédurale avancée
Rattrapage - ING1 GI

2 juillet 2018

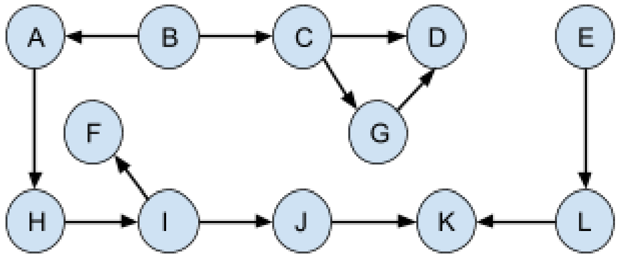
Modalités

- Durée : 2 heures
- Vous devez rédiger votre copie à l'aide d'un **stylo à encre** exclusivement.
- Toutes vos affaires (sacs, vestes, trousse, etc.) doivent être placées à l'avant de la salle.
- Aucun document autorisé.
- Aucune machine électronique ne doit se trouver sur vous ou à proximité, même éteinte.
- Aucun déplacement n'est autorisé.
- Aucune question au professeur n'est autorisé.
- Aucun échange, de quelque nature que ce soit, n'est possible.
- Le barème est donné à titre indicatif.

<u>Rappel :</u>
La **`procédure erreur(msg : Chaine de caractère)`** permet de quitter un programme en affichant le message **`msg`** à l'utilisateur.

Exercice 1 : Petits problèmes autour de graphes (6pts = 1+1+1+2+1)

1. On s'intéresse aux graphes dont tous les sommets sont de degré trois.
    a. Construire de tels graphes ayant 4 sommets, 5, 6 et 7. Quelle hypothèse pouvez-vous formuler ?
    b. Démontrer cette hypothèse déduite.
2. Le tri topologique d'un graphe orienté sans circuit `G = (S,A)` consiste à ordonner linéairement tous ses sommets de telle sorte que si l'arc `(u,v)` ∈ `A`, alors `u` apparaisse avant `v`. (Si le graphe comporte des circuits, aucun ordre linéaire n'est possible.)
    a. Donner un ordre topologique du graphe suivant :



b. Écrire l'algorithme qui permet de déterminer un ordre topologique d'un graphe. On supposera que cet ordre existe, c'est à dire qu'il n'y a pas de circuit.

3. Expliquer comment les composantes fortement connexes d'un graphe orienté peuvent être modifiées lorsqu'on ajoute *un* arc quelconque à ce graphe orienté.

Exercice 2 : Une histoire de poissons (5pts = 0.5+1+0.5+0.5+2+0.5)

Un complexe médical souhaite installer des aquariums dans l'espace d'attente. Chaque médecin souhaite avoir certains poissons, mais ils ne sont pas tous compatible entre eux. Dans le tableau ci-dessous, une croix signifie que les poissons (nommés de A à H) ne peuvent pas cohabiter dans un même aquarium.

	A	B	C	D	E	F	G	H
A		✘	✘	✘			✘	✘
B	✘				✘	✘	✘	
C	✘			✘		✘	✘	✘
D	✘		✘		✘			✘
E		✘		✘		✘	✘	
F		✘	✘		✘			
G	✘	✘	✘		✘			
H	✘		✘	✘				

1. Modéliser le problème avec un graphe. Justifier.
2. Que peut-on dire du sous-graphe {A,C,D,H} ? Proposer un double encadrement de $\chi(G)$. Justifier.
3. Combien d'aquariums au minimum faut-il ? Justifier.
4. Donner le nom de l'algorithme vu en cours qui permet de calculer une approximation du nombre chromatique d'un graphe.
5. Écrire cet algorithme.
6. Appliquer cet algorithme à l'instance donnée plus haut.

Exercice 3 : Problème de la 2-satisfiabilité (9pts = 1+0.5+1+1+0.5+1+2+1+1) repris de *Algorithms* par Dasgupta *et al.*

Dans le problème 2SAT, on a un ensemble de clauses, où chaque clause est la disjonction (*ou*) de deux littéraux (un littéral est soit une variable booléenne, soit la négation d'une variable booléenne). On cherche une affectation des variables à vrai ou faux telle que toutes les clauses soient satisfaites, c'est-à-dire qu'il y ait au moins un littéral vrai dans chaque clause.

Voici un exemple d'instance :
(*x1 ou non(x2)*) *et* (*non(x1) ou non(x3)*) *et* (*non(x3) ou x4*) *et* (*non(x1) ou x4*)

Cette instance a une solution (*x1,x2,x3,x4*) = (vrai,faux,faux,vrai).

a. Cette instance a-t-elle d'autres solutions ? Si non, justifier. Si oui les trouver toutes.
b. Donner une instance sans solution la plus simple possible.

Le but de ce problème est de résoudre 2SAT efficacement en le réduisant au calcul des composantes fortement connexes d'un graphe orienté. Soit *Inst* une instance de 2SAT avec *n* variables et *m* clauses. On construit le graphe G_{Inst} de la manière suivante :

- G_{Inst} a *2n* sommets, un pour chacune des variables et un pour chacune des négations ;
- G_{Inst} a *2m* arcs : pour chaque clause *a ou b* de *Inst* (avec *a* et *b* des littéraux), G_{Inst} a un arc de la négation de *a* à celle de *b* et de la négation de *b* à celle de *a*.

Notez que la clause *a ou b* est équivalent à l'une ou l'autre des implications *non(a) -> b* et *non(b) -> a* . En ce sens, G_{Inst} représente toutes les implications de l'instance *Inst*.

c. Construire ce graphe pour l'exemple donné et pour votre réponse à la question b.
d. Montrer que si G_{Inst} a une composante fortement connexe qui contient à la fois *x* et *non(x)* pour une variable *x* alors le problème n'a pas de solution.

Réciproquement, <u>on admettra que</u> que si aucune des composantes fortement connexes ne contient à la fois *x* et *non(x)* pour toute les variables *x*, alors le problème a nécessairement une solution.

e. Donner le nom d'un algorithme qui calcule les composantes fortement connexes d'un graphe orienté.
f. Donner le principe de cet algorithme en français.
g. Donner le (pseudo-)code correspondant.
h. Donner la complexité de cet algorithme. Justifier.
i. Donner la complexité du problème 2SAT. Justifier.