SESSION 2011

Filière : 2ème concours ENS Lyon

INFORMATIQUE

Durée : 3 heures

Les calculatrices sont interdites.

Le sujet comporte deux parties et neuf pages numérotées de 1 à 9. Le barême sera adapté à la longueur du sujet.

Bien que les deux parties aient un rapport entre elles, elles sont largement indépendantes et peuvent être traitées comme telles. La première partie traite d'algorithmique sur les séquences d'entiers, la deuxième concerne les graphes parfaits. Il est toujours possible d'admettre les résultats énoncés dans les questions antérieures pour répondre à une question. Toutes vos réponses doivent être justifiées rigoureusement, et prouvées formellement lorsque cela est demandé ou que vous l'estimez nécessaire.

Définitions et notations communes à tout le sujet

L'ensemble des entiers naturels est noté $\mathbb{N}$ et celui des entiers naturels strictement positifs est noté $\mathbb{N}^*$. Pour deux entiers naturels a et b tels que $a \leq b$ on notera $[\![a, b]\!]$ l'ensemble $\{x \in \mathbb{N} \mid a \leq x \leq b\}$. On appelle *intervalles* les sous-ensembles d'entiers de la forme $[\![a, b]\!]$, avec $a, b \in \mathbb{N}$ et $a \leq b$. Pour $n \in \mathbb{N}$, l'ensemble $[\![1, n]\!]$ sera appelé l'ensemble des entiers de 1 à n.

Une *séquence* $S = (s_i)_{1 \leq i \leq k}$, avec $k \in \mathbb{N}$, est une suite finie d'éléments. L'entier k est appelé la longueur de la séquence S et par convention la séquence est vide lorsque $k = 0$. Dans la suite, les séquences S seront notées entre crochets : $S = [s_1, s_2, \ldots, s_k]$. De manière intuitive, une *sous-séquence* S' de S est une séquence formée d'éléments de S qui apparaissent dans S' avec le même ordre relatif que dans S. Formellement, une sous-séquence S' de longueur $k' \leq k$ d'une séquence $S = [s_1, s_2, \ldots, s_k]$ est une séquence de k' éléments telle qu'il existe une application strictement croissante σ de $[\![1, k']\!]$ dans $[\![1, k]\!]$ telle que $S' = [s_{\sigma(1)}, s_{\sigma(2)}, \ldots, s_{\sigma(k')}]$. Une séquence est dite *sans répétitions* lorsque tous ses éléments sont deux à deux distincts. On dit qu'une séquence d'entiers sans répétitions S de longueur k *inverse deux de ses éléments* s_i et s_j, avec $i, j \in [\![1, k]\!]$, si et seulement si $i < j$ et $s_i > s_j$. Une *permutation* S de $[\![1, n]\!]$ est une séquence sans répétitions de longueur n dont les éléments sont les entiers de 1 à n.

Le *cardinal* d'un ensemble fini est son nombre d'éléments. Pour un ensemble X et un entier naturel n, on utilisera l'abus de notation qui consiste à écrire $x_1, x_2, \ldots, x_n \in X$ à la place de $(x_1, x_2, \ldots, x_n) \in X^n$. On dit qu'un ensemble S est minimal (resp. maximal) pour l'inclusion parmi une famille d'ensembles $\mathcal{F}$ si et seulement si aucun élément de $\mathcal{F}$ n'est strictement contenu dans S (resp. ne contient strictement S).

1 Trier une séquence d'entiers avec k files en parallèle

Le but de cet exercice est de trier une séquence d'entiers strictement positifs sans répétitions en faisant circuler les entiers qui la composent dans un réseau de files en parallèle. Nous considérons l'ensemble de toutes les séquences, *sans restriction sur leur longueur*. On se propose de répondre aux questions suivantes : quelles séquences peut-on trier avec k files en parallèle ? ou réciproquement, combien de files faut-il pour trier une séquence donnée ?

Dans toute la suite, même si on omet de le préciser, les séquences d'entiers considérées sont des *séquences d'entiers strictement positifs sans répétitions*.

Une *file* est un espace de stockage linéaire à deux extrémités dans lequel les éléments entrent d'un côté, appelé la *queue* de la file, et ressortent de l'autre côté, appelé la *tête* de la file, sans modification de leur ordre relatif. C'est-à-dire que l'ordre de sortie des éléments par la tête de file est le même que leur ordre d'entrée par la queue de file. Les notions de *croissance,*

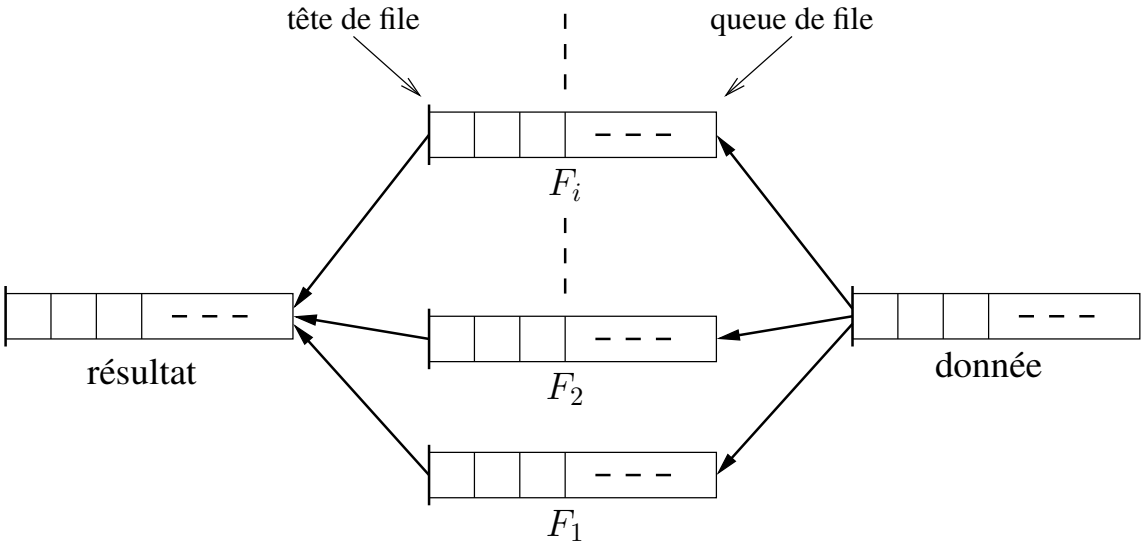

FIG. 1 – Un réseau de files en parallèle. La tête des files est matérialisée par un trait plus long et plus épais.

décroissance et monotonie d'une file se rapportent à la notion en question pour la séquence des entiers qu'elle contient pris depuis la tête de file vers la queue de file. Les seules opérations possibles sur une file sont : tester si la file est vide, lire la tête et la queue de la file, défiler la file (c'est-à-dire extraire son élément de tête), enfiler un élément dans la file (c'est-à-dire insérer l'élément en queue de file).

Un réseau de k files en parallèle est composé au total de $k + 2$ files :

- 1 *file donnée*
- k *files intermédiaires* en parallèle
- 1 *file résultat*

Pour trier une séquence S par un réseau de k files en parallèle (voir figure 1) on fait circuler les entiers composant S dans le réseau en suivant les arcs joignant les files, exclusivement dans le sens de leur orientation, depuis la file donnée jusque dans la file résultat.

Initialement, la séquence $S = [s_1, s_2, \ldots, s_n]$ à trier, où $n \in \mathbb{N}^*$ est la longueur de S, est placée dans la file donnée : s_1 en tête de file et s_n en queue de file. Ensuite, on effectue les déplacements des entiers dans le réseau : un déplacement après l'autre et un entier à la fois. On ne peut opérer que deux types très contraints de déplacement :

- les *déplacements d'entrée* consistent à retirer l'élément en tête de la file donnée pour le placer en queue d'une file intermédiaire quelconque F_i, avec $i \in [\![1, k]\!]$. Un tel déplacement est noté $In(i)$.
- les *déplacements de sortie* consistent à retirer l'élément en tête d'une file intermédiaire quelconque non vide F_i, avec $i \in [\![1, k]\!]$, pour le placer en queue de la file résultat. Un tel déplacement est noté $Out(i)$.

Le tri se termine lorsque tous les entiers se retrouvent dans la file résultat triés dans l'ordre croissant depuis la tête de file vers la queue de file. Rappelez vous que l'algorithme de tri prend en entrée une séquence quelconque sans restriction sur sa longueur. Aussi, dans tout le problème, on considère que les files intermédiaires ainsi que la file donnée et la file résultat sont de capacité infinie. On appelle scénario de tri une séquence de déplacements qui

amène tous les éléments dans la file résultat rangés dans l'ordre croissant. Exemple : le scénario $[In(1), In(2), Out(2), Out(1), In(1), Out(1)]$ trie la séquence $[9, 3, 20]$ avec 2 files intermédiaires.

Le but de cette partie est de prouver le théorème suivant et de donner un algorithme de tri.

Théorème 1 *Pour trier une séquence S d'entiers strictement positifs sans répétitions dont la plus grande sous-séquence décroissante est de longueur k, k files en parallèle sont nécessaires et suffisantes. De plus, étant donné un réseau d'une infinité dénombrable de files en parallèle, il existe un algorithme qui trie toute séquence d'entiers strictement positifs sans répétition S à l'aide de ce réseau en n'en utilisant que les k premières files intermédiaires.*

Question 1 *De combien de déplacements se compose un scénario de tri ? Justifiez.*

Question 2 *Montrez que pour n'importe quel scénario de tri $\mathcal{T}$, on peut construire un scénario de tri $\mathcal{T}'$ qui utilise exactement les mêmes déplacements mais qui soit tel que tous les déplacements d'entrée soient effectués avant tous les déplacements de sortie.*

Un tel scénario $\mathcal{T}'$ est dit normal. Dans toute la suite, on omettra de le préciser mais *on ne considère que des scénarios normaux.*

Question 3 *Montrez qu'à chaque étape d'un scénario de tri, chacune des files intermédiaires qui sont non vides est triée dans l'ordre croissant.*

Question 4 *Déduisez-en que, dans un scénario de tri, deux éléments qui sont inversés par S ne peuvent pas aller dans la même file intermédiaire. Et montrez que si S contient une sous-séquence décroissante de longueur k, avec $k \in \mathbb{N}^*$, il faut au moins k files en parallèle pour trier S.*

Question 5 *Montrez que pour trier une séquence d'entiers S, il suffit de ranger tous les éléments de S dans les files intermédiaires de sorte que chacune de ces files soit triée dans l'ordre croissant.*
Donnez un algorithme (décrit en pseudo-code ou en langage courant) pour terminer le tri à partir de cette configuration. Veillez à n'utiliser que les opérations autorisées sur les files.

Question 6 *Quel est le nombre minimum de files nécessaire pour trier la séquence suivante ?*

$$S = [6, 3, 1, 7, 5, 9, 8, 2, 4]$$

Dessiner l'état des files intermédiaires, dans un scénario de tri utilisant ce nombre minimum de files, juste après que tous les éléments soient sortis de la file donnée, et avant qu'aucun d'entre eux n'entre dans la file résultat.

Soit n un entier naturel, soit S une permutation de $[\![1, n]\!]$ et soit $a, b \in [\![1, n]\!]$. L'intervalle $[\![a, b]\!]$ est dit *conservé* par la permutation S si et seulement si $\{i \in [\![1, n]\!] \mid a \leq s_i \leq b\}$ est un intervalle. L'intervalle $[\![1, n]\!]$ et les intervalles singletons $[\![a, a]\!]$, pour $a \in [\![1, n]\!]$, sont toujours conservés par toutes les permutations de $[\![1, n]\!]$, on les appelle les *intervalles triviaux*. Exemple : la permutation $S = [1, 3, 5, 4, 2]$ conserve l'intervalle $[\![3, 5]\!]$ car $\{i \in [\![1, 5]\!] \mid 3 \leq s_i \leq 5\} = \{2, 3, 4\} = [\![2, 4]\!]$; par contre S ne conserve pas l'intervalle $[\![2, 4]\!]$ car $\{i \in [\![1, 5]\!] \mid 2 \leq i \leq 4\} = \{2, 4, 5\}$ n'est pas un intervalle.

Question 7 *Donnez une permutation S de $[\![1, 9]\!]$ qu'il soit possible de trier avec 2 files en parallèle et telle que S ne conserve aucun intervalle non trivial.*

On laisse de côté le cas particulier des permutations envisagé dans la question précédente pour revenir au cas général des séquences d'entiers strictement positifs sans répétitions, jusqu'à la fin de cette partie.

Question 8 *Dans cette question, on considère le cas où on dispose d'une infinité dénombrable de files en parallèle $(F_i)_{i\in\mathbb{N}^*}$. Pour $i \in \mathbb{N}^*$, on note $Q(F_i)$ l'élément en queue de la file F_i lorsque celle-ci est non vide, et $Q(F_i) = 0$ sinon. Donnez un algorithme qui effectue tous les déplacements d'entrée et qui garantit les invariants suivants. A chaque étape de l'algorithme,*

- *pour tout $i \in \mathbb{N}^*$, si F_i est non-vide alors F_i est triée en ordre croissant ; et*
- *la suite $(Q(F_i))_{i\in\mathbb{N}^*}$ est décroissante.*

Veillez à n'utiliser que les opérations autorisées sur les files.

Question 9 *On note $S = [s_1, \ldots, s_n]$, avec $n \in \mathbb{N}^*$, la séquence fournie en entrée de l'algorithme. Montrez qu'un algorithme $\mathcal{A}$ vérifiant l'invariant de la question 8 vérifie également la propriété suivante : après le p-ième déplacement d'entrée, avec $1 \leq p \leq n$, pour tout $i \in \mathbb{N}^*$ tel que F_i est non vide, il existe une sous-séquence décroissante de la séquence $S_p = [s_1, \ldots, s_p]$ qui est de longueur i et dont le dernier élément est $Q(F_i)$.*
Déduisez en que $\mathcal{A}$ utilise au plus k files pour trier S, où k est le maximum des longueurs d'une sous-séquence décroissante de S.

On s'intéresse maintenant à l'implémentation de l'algorithme que vous avez décrit à la question 8, à la fois celle de la partie de l'algorithme effectuant les déplacements d'entrée et celle de la partie effectuant les déplacements de sortie (voir question 5). Vous pouvez utiliser uniquement des listes et des tableaux. Vous compterez la complexité de votre implémentation en comptant un coût constant pour les opérations de lecture et d'écriture d'un entier ou d'une adresse mémoire. Vous pouvez réserver un espace consécutif en mémoire pour stocker des entiers et des adresses mémoires. Cette opération sera effectuée pour un coût égal à la taille de l'espace réservé, en considérant que la taille d'un entier et d'une adresse mémoire est constante. L'accès au contenu stocké à une adresse mémoire ainsi que l'accès à la case numéro i d'un tableau, où i est un entier naturel, seront considérés avoir un coût constant.

Question 10 *Quelle est la complexité dans le pire des cas de l'algorithme que vous avez proposé à la question 8 en fonction de la longueur n de la séquence donnée et de la longueur k de sa plus longue sous-séquence décroissante ? Donnez en une implémentation en $O(k\,n)$ dans le pire des cas. Pensez à décrire l'implémentation de la partie de l'algorithme effectuant les déplacements de sortie.*

Dans la question suivante on s'intéresse uniquement à améliorer la complexité de la partie de l'algorithme effectuant les déplacements d'entrée.

Question 11 *Comment implémenter cette partie de l'algorithme pour obtenir une complexité de $O(n \log k)$ dans le pire des cas, pour l'ensemble des déplacements d'entrée ? On considèrera que la longueur n de la séquence se trouvant initialement dans la file donnée vous est fournie au début de l'algorithme (en pratique, il suffit de parcourir la file donnée pour la déterminer, ce que vous n'avez pas à faire).*

Dans les deux questions suivantes, on s'intéresse à la partie de l'algorithme traitant les déplacements de sortie.

Question 12 *Dans cette question, et dans cette question seulement, vous avez droit à toute structure de donnée que vous jugez utile. Comment implémenter la partie de votre algorithme traitant les déplacements de sortie pour obtenir une complexité de $O(n \log k)$ dans le pire des cas ?*

La question suivante vise à améliorer la complexité de la partie de l'algorithme traitant les déplacements de sortie dans le cas particulier où la séquence S fournie en entrée est une permutation.

Question 13 *Lorsque la séquence donnée S est une permutation, que peut-on faire lors des mouvements d'entrée pour pouvoir implémenter la partie effectuant les mouvements de sortie en temps $O(n)$ dans le pire des cas, pour l'ensemble des mouvements de sortie ? Pour cette question vous n'avez droit qu'aux listes et aux tableaux. Comme précédemment, on considère que la longueur n de la permutation se trouvant initialement dans la file donnée vous est fournie au début de l'algorithme.*

2 Des graphes parfaits

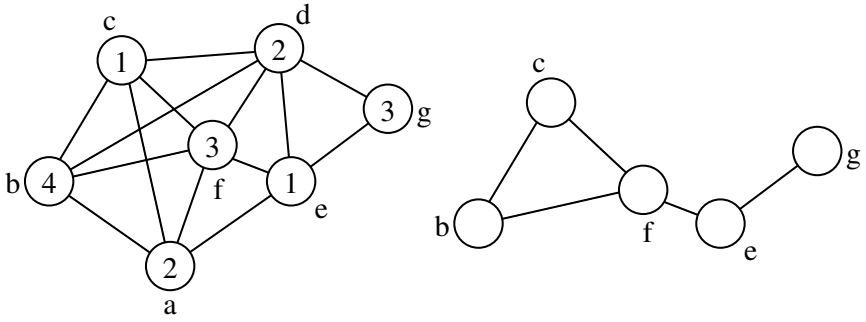

FIG. 2 – A gauche, un graphe exemple G à 7 sommets a, b, c, d, e, f, g avec une coloration propre à 4 couleurs $1, 2, 3, 4$. A droite, le graphe induit de G par les sommets b, c, e, f, g.

Un *graphe* G est la donnée d'un ensemble $V(G)$ dont les éléments sont appelés les *sommets* de G, et d'un sous-ensemble $E(G)$ de l'ensemble des parties à deux éléments de V, dont les éléments sont appelés les *arêtes* de G. On note $G = (V(G), E(G))$. Pour deux sommets distincts x et y de G, tels que $\{x, y\} \in E(G)$, on dit que x et y sont *adjacents* dans G, ou encore qu'il y a une arête entre x et y. L'arête $\{x, y\}$ sera alors abusivement notée xy ou yx, indifféremment. Une non-arête est une paire de sommets non adjacents. Le voisinage d'un sommet x, noté $N(x)$, est l'ensemble des sommets de G adjacents à x. Par exemple, ab et df sont des arêtes du graphe G donné sur la figure 2 et bg et ad en sont des non arêtes. Le voisinage du sommet b de G est $N(b) = \{a, c, d, f\}$.

Une *clique* d'un graphe G est un sous-ensemble K des sommets de G tel que $\forall x, y \in K, xy \in E(G)$. Le maximum des entiers k tel qu'il existe une clique de cardinal k dans G est

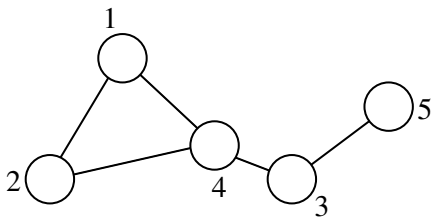

FIG. 3 – Un graphe de permutation à 5 sommets, correspondant à la permutation $[4, 2, 1, 5, 3]$.

noté $\omega(G)$. Exemple : dans le graphe G de la figure 2, $\{d, g\}$, $\{d, e, f\}$ et $\{b, c, d, f\}$ sont des cliques et $\omega(G) = 4$.

A une permutation π des entiers de 1 à n, on associe un graphe $\mathcal{G}[\pi]$ tel que les sommets de $\mathcal{G}[\pi]$ sont les entiers de 1 à n, et il y a une arête entre les sommets i et j, avec $i \neq j$, si et seulement si i et j sont inversés par π. Un *graphe de permutation* G est un graphe tel qu'il existe une permutation π des entiers de 1 à n, où n est le nombre de sommets de G, telle qu'il est possible d'identifier les sommets de G aux entiers de 1 à n de sorte que $G = \mathcal{G}[\pi]$. Un exemple de graphe de permutation, ainsi que la permutation correspondante, sont donnés sur la figure 3.

Une *coloration propre* d'un graphe G est une assignation d'une couleur à chaque sommet de G de sorte que deux sommets adjacents se voient assigner des couleurs différentes. Plus formellement, une coloration propre est une application ϕ de l'ensemble $V(G)$ des sommets de G dans $\mathbb{N}$ telle que $\forall x, y \in V(G), xy \in E(G) \Rightarrow \phi(x) \neq \phi(y)$. Le nombre chromatique d'un graphe G, noté $\chi(G)$, est le nombre minimum de couleurs d'une coloration propre, c'est-à-dire le minimum, sur toutes les colorations propres ϕ, du cardinal de $\phi(V(G))$ (ensemble des images de ϕ). Le graphe G de la figure 2 est donné avec une coloration propre à 4 couleurs $1, 2, 3, 4$. Le nombre chromatique de G est 4 car il n'est pas possible de le colorer avec 3 couleurs seulement.

Question 14 *Montrez que si G contient une clique de cardinal k, alors toute coloration propre de G utilise au moins k couleurs.*
Déduisez-en une inégalité entre le nombre chromatique $\chi(G)$ d'un graphe et le cardinal maximum de ses cliques $\omega(G)$.

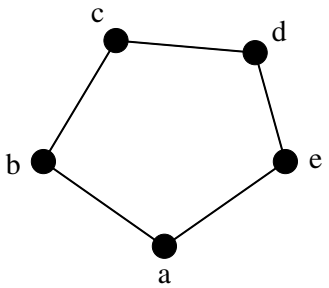

FIG. 4 – Le graphe C_5 utilisé dans la question 15.

Question 15 *Quel est le nombre chromatique du graphe C_5 donné sur la figure 4 ? Que vaut $\omega(C_5)$?*

Question 16 *Donnez un graphe G tel que $\omega(G) = 2$ et $\chi(G) = 4$.*

Le *sous-graphe induit* d'un graphe G par un sous-ensemble de sommets $X \subseteq V(G)$ est le graphe noté $G[X]$ et défini par $V(G[X]) = X$ et $E(G[X]) = \{xy \in E(G) \mid x, y \in X\}$ (un exemple de sous-graphe induit est donné sur la figure 2). Un *graphe parfait* est un graphe G tel que pour tout sous-graphe induit H de G on a $\chi(H) = \omega(H)$.

Question 17 *Soit un scénario de tri $\mathcal{T}$ d'une permutation π utilisant l files en parallèle, avec $l \in \mathbb{N}^*$. Montrez que $\mathcal{T}$ donne naturellement une coloration propre à l couleurs de $\mathcal{G}[\pi]$. Réciproquement, soit une coloration propre de $\mathcal{G}[\pi]$ utilisant l couleurs, montrez qu'elle définit un scénario de tri de π avec l files en parallèle.*

Question 18 *Déduisez-en la relation entre le nombre minimum de files en parallèle pour trier π et le nombre chromatique de $\mathcal{G}[\pi]$.*

Question 19 *Montrez que tout sous-graphe induit d'un graphe de permutation est un graphe de permutation et déduisez-en que les graphes de permutation sont parfaits.*

On s'intéresse maintenant à une autre classe de graphes, dont nous allons montrer qu'ils sont aussi parfaits. Il s'agit des graphes triangulés.

Un *chemin* P d'un graphe G est une séquence $P = [x_1, x_2, \ldots, x_p]$ de sommets de G deux à deux distincts, avec $p \in \mathbb{N} \setminus \{0, 1\}$, telle que $\forall i \in [\![1, p-1]\!], x_i x_{i+1} \in E(G)$. L'entier p est appelé la longueur du chemin P, et on dit que P est un chemin de x_1 à x_p.

Un *cycle* C d'un graphe G est un chemin $[x_1, x_2, \ldots, x_p]$ de G tel que $x_p x_1 \in E(G)$. L'entier p est appelé la longueur du cycle C. Une corde d'un cycle $C = [x_1, x_2, \ldots, x_p]$ est une arête joignant deux sommets x_i et x_j non-consécutifs sur le cycle C, c'est-à-dire tels que $\{i, j\} \neq \{1, p\}$ et $|i - j| \neq 1$. Exemple : dans le graphe de la figure 2, $[a, c, f, e]$ est un chemin de longueur 4 et $[a, b, c, d, e]$ et $[a, b, d, e]$ sont des cycles. Le cycle $[a, b, c, d, e]$ possède deux cordes ac et bd, alors que $[a, b, d, e]$ ne possède aucune corde.

Un *graphe cycle* est un graphe $G = (V, E)$, où $V = \{x_1, x_2, \ldots, x_n\}$, tel que $C = [x_1, x_2, \ldots, x_n]$ est un cycle et G ne contient aucune autre arête que celles du cycle C (par exemple, le graphe C_5 de la figure 4 est un graphe cycle à 5 sommets). Un *graphe triangulé* est un graphe qui ne contient pas de sous-graphe induit d'au moins 4 sommets qui soit un graphe cycle.

Question 20 *Montrez que, de manière équivalente, un graphe triangulé est un graphe dont tous les cycles de longueur au moins 4 possèdent au moins une corde.*

Question 21 *Le graphe F donné sur la figure 5 est-il triangulé ?*

Question 22 *Montrez que tout sous-graphe induit d'un graphe triangulé est un graphe triangulé.*

Un graphe G est dit *connexe* si et seulement si pour tous $x, y \in V(G)$, il existe un chemin de x à y dans G. Une *composante connexe* d'un graphe G est un sous-ensemble X de sommets de G tel que $G[X]$ est connexe et X est maximal pour l'inclusion parmi les sous-ensembles

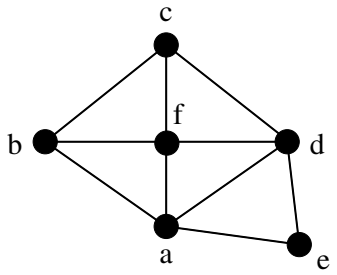

FIG. 5 – Le graphe F à six sommets utilisé dans la question 21.

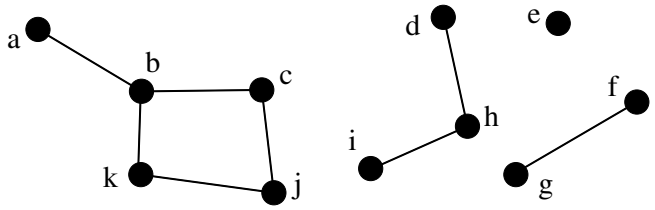

FIG. 6 – Un graphe non connexe. Ses composantes connexes sont $\{a, b, c, j, k\}$, $\{d, h, i\}$, $\{e\}$ et $\{f, g\}$.

ayant cette propriété. Un exemple de graphe et de ses composantes connexes est donné sur la figure 6.

Pour un sous-ensemble S de sommets de G, on note $G - S$ le sous-graphe induit de G par $V(G) \setminus S$. Soit G un graphe connexe et soient a et b deux sommets distincts de G, un *ab-séparateur* est un sous-ensemble de sommets S de G tel que a et b sont dans des composantes connexes différentes de $G - S$. Un *ab-séparateur minimal* est un ab-séparateur S qui est minimal pour l'inclusion parmi tous les ab-séparateurs. Un *séparateur minimal* d'un graphe G connexe est un sous-ensemble de sommets S de G tel qu'il existe deux sommets distincts $a, b \in V(G) \setminus S$ tels que S est un ab-séparateur minimal.

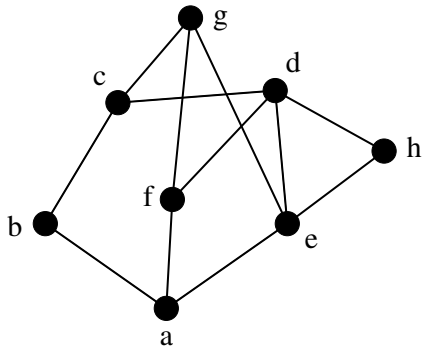

FIG. 7 – Le graphe H à huit sommets utilisé dans la question 23.

Question 23 *Quels sont les ad-séparateurs minimaux dans le graphe H donné sur la figure 7 ? En plus des précédents, H possède quatre autres séparateurs minimaux : quels sont-ils ?*

Question 24 *Donnez un graphe G dans lequel il existe un séparateur minimal strictement inclus dans un autre séparateur minimal.*

Question 25 *Montrez que si deux sommets a et b d'un graphe connexe sont non-adjacents, alors il existe au moins un ab-séparateur.*

Question 26 *Soient a et b deux sommets d'un graphe G connexe et soit S un ab-séparateur de G. On note A et B les composantes connexes de $G - S$ contenant respectivement les sommets a et b. Montrez que s'il existe $x \in S$ tel que x n'est adjacent à aucun élément de A alors $S \setminus \{x\}$ est un ab-séparateur.*
En déduire que si S est un ab-séparateur minimal, alors tous les sommets de S sont adjacents à au moins un sommet de A et à au moins un sommet de B.

Le but des questions 27 et 28 est de montrer le théorème suivant.

Théorème 2 *Un graphe connexe est triangulé si et seulement si tous ses séparateurs minimaux sont des cliques.*

Question 27 *Montrez que si un graphe G connexe possède un cycle sans corde C de longueur au moins 4, alors G possède un séparateur minimal S qui n'est pas une clique.*
Indication : montrez qu'il existe un séparateur minimal S contenant au moins 2 sommets non-consécutifs de C.

Question 28 *Soit S un séparateur minimal d'un graphe G connexe tel que S n'est pas une clique. Montrez que G contient un cycle sans corde de longueur au moins 4.*
Indication : vous pourrez considérer une non-arête xy dans S et montrer l'existence d'un chemin joignant x à y dans G et dont tous les sommets (distincts de x et y) appartiennent à une unique composante connexe de $G - S$.

Un *ordre d'élimination simplicial* d'un graphe G est un ordre linéaire $x_1, x_2, \ldots, x_n$ sur les sommets de G tel que $\forall k \in [\![1, n-1]\!], N(x_k) \cap \{x_{k+1}, \ldots, x_n\}$ est une clique de G. Le but des questions 29 à 31 est de montrer le théorème suivant.

Théorème 3 *Un graphe est triangulé si et seulement s'il admet un ordre d'élimination simplicial.*

Question 29 *Pourquoi suffit-il de montrer le théorème 3 pour les graphes connexes pour en déduire le résultat général ? Justifiez votre réponse.*

Question 30 *En utilisant la caractérisation des graphes triangulés par leurs cycles, montrez qu'aucun graphe non triangulé n'admet d'ordre d'élimination simplicial.*

Question 31 *Montrez qu'à l'inverse tout graphe triangulé connexe admet un ordre d'élimination simplicial.*
Indication : vous pourrez utiliser les séparateurs minimaux pour faire une récurrence sur le nombre de sommets du graphe.

Question 32 *En utilisant leur caractérisation par l'existence d'un ordre d'élimination simplicial, montrez que les graphes triangulés sont parfaits.*