IFT800 - Série d'exercices #5

Manuel Lafond

Décomposition en arbre

Exercice 1. Considérez les variantes des graphes simples avec une seule arête modifiée, et calculez la treewidth.

1. Soit G un cycle avec une arête supplémentaire (i.e. G est le cycle $(v_1, v_2, \ldots, v_n, v_1)$, plus une arête $v_i v_j$ quelconque). Est-ce que $tw(G) = 2$?

2. Soit G un arbre avec une arête supplémentaire ajoutée. Est-ce que $tw(G) = 1$?

3. Soit G une clique avec une arête supprimée. Est-ce que $tw(G) = n - 1$?

Exercice 2. Soit G une grille $n \times n$ (i.e. les sommets sont $v_{i,j}$ pour $i, j \in \{1, \ldots, n\}$ et il y a une arête entre $v_{i,j}$ et $v_{i+1,j}$ et entre $v_{i,j}$ et $v_{i,j+1}$ lorsque $i + 1 \leq n$ et $j + 1 \leq n$, respectivement).

Montrez que $tw(G) \leq n$. Il est recommandé de décrire une décomposition.

Si vous voulez un défi, montrez que $tw(G) = n$. Par contre, je ne donnerai pas la solution.

Exercice 3. Soit $G = (V, E)$ un graphe. La contraction d'une arête $ab \in E$ consiste à 1) ajouter un nouveau sommet z; 2) mettre $N(z) = N(a) \cup N(b)$; 3) enlever a et b de G.

Soit G' le graphe obtenu par la contraction d'une arête ab de G. Montrez que $tw(G') \leq tw(G)$.

Exercice 4. Soit $G = (V, E)$ un graphe. Un sous-ensemble de sommets $X \subseteq V$ est appelé un *séparateur* si $G - X$ n'est pas connexe. Soient $G_1, \ldots, G_k$ les graphes formant les composantes connexes de $G - X$. Montrez que $tw(G) \leq |X| + \max_i(tw(G_i))$.

Exercice 5. Soit $G = (V, E)$ un graphe. Soit le problème suivant.

VERTEX-COVER
Entrée : Graphe $G = (V, E)$
Paramètre : $tw(G)$
Sortie : un ensemble couvrant les arêtes $X \subseteq V$ de taille minimum

Supposez qu'on vous donne un décomposition en arbre $T = (V_T, E_T)$ de G. Donnez un algorithme de programmation dynamique en temps $O(f(tw(G))n^c)$ sur T pour résoudre le problème VERTEX-COVER. Il est suffisant de décrire vos récurrences.

Exercice 6. Soit $G = (V, E)$ un graphe. Soit le problème suivant.

MIN-DOMSET
Entrée : Graphe $G = (V, E)$
Paramètre : $tw(G)$
Sortie : un ensemble dominant $X \subseteq V$ de taille minimum

Supposez qu'on vous donne un décomposition en arbre $T = (V_T, E_T)$ de G. Donnez un algorithme de programmation dynamique en temps $O(f(tw(G))n^c)$ sur T pour résoudre le problème MIN-DOMSET. Il est suffisant de décrire vos récurrences.

Ici, il est commode d'utiliser trois couleurs pour vos sommets. Une couleur pour un sommet dans l'ensemble indépendant, une couleur pour un sommet qui n'y est pas, mais qui est présentement dominé, et une couleur pour un sommet qui n'y est pas, et qui n,est pas encore dominé.

Exercice 7. Soit $G = (V, E)$ un graphe *orienté* et sans cycle de taille 2. La *version non-orientée* de G est le graphe G' obtenu en ignorant la direction des arêtes. Une décomposition en arbre d'un graphe orienté G réfère à une décomposition de la version non-orientée de G.

Dans le problème FEEDBACK ARC SET, on veut ordonner les sommets de façon à avoir un nombre minimum d'arêtes qui vont "vers l'arrière".

FEEDBACK ARC SET
Entrée : Graphe orienté $G = (V, E)$
Paramètre : $tw(G')$, où G est la version non-orientée de G
Sortie : une permutation $(v_1, v_2, \ldots, v_n)$ telle que le nombre d'arêtes (v_j, v_i) avec $j > i$ est minimum.

Supposez qu'on vous donne un décomposition en arbre $T = (V_T, E_T)$ de la version non-orientée de G. Donnez un algorithme de programmation dynamique en temps $O(f(t(w))n^c)$ sur T pour résoudre ce problème. Il est suffisant de décrire vos récurrences.

Ici, il ne faut pas considérer les sous-ensembles à chaque sac, mais plutôt les permutations à chaque sac.

Exercice récapitulatifs

Exercice 8. Dans le problème SEQUENCES-INTRUES, on reçoit m séquences de caractères $S = \{S_1, \ldots, S_m\}$ de même longueur. On veut savoir si on peut supprimer k de nos séquences de façon à ce que chaque paire de séquences soit à distance au plus d, où d nous est donné.

C'est-à-dire, est-ce qu'il existe $S' \subseteq S$ tel que $|S'| \leq k$, et tel que pour tout $S_i, S_j \in S \setminus S'$, on a $d(S_i, S_j) \leq d$?

Montrez que cet algorithme est FPT en le paramètre k, le nombre de séquences à supprimer. Notez que d ne fait pas partie des paramètres.

Exercice 9. Soit P un problème de minimisation résoluble en temps polynomial (et ce indépendamment d'un paramètre). Montrez que P admet un noyau de taille $O(1)$ selon le paramètre k, la valeur d'une solution minimum.

Exercice 10. Dans MIN-ONES-2-SAT-FACILE, on reçoit des clauses $C_1, \ldots, C_m$ chacune avec deux variables. On suppose également que chaque clause a ses deux variables positives. Ceci rend la question beaucoup plus facile que si on permettait des variables négatives. Le but est de savoir s'il existe une assignation qui satisfait les clauses où au plus k variables sont $true$, et les autres à $false$.

Montrez qu'on peut obtenir un noyau pour MIN-ONES-2-SAT-FACILE, où le paramètre k est le nombre maximum de variables à $true$.

Exercice 11. Donnez un algorithme de branchement FPT pour le problème MIN-ONES-2-SAT ci-haut, mais sans supposer que chaque clause a toujours deux variables positives.

Exercice 12. Rappelez-vous du problème LIVRAISON-CIRCULAIRE de la section 7.2.2 des notes de cours. On a un cycle $G = (V, E)$ avec n sommets et on a un ensemble de paquets représentés par des couples $(s_i, t_i) \in V \times V$, où s_i est le départ et t_i l'arrivée. Il faut choisir une direction pour chaque paquet de façon à ce que la charge maximum d'une arête soit au plus k. La charge d'une arête est le nombre de chemins qui passent par cette arête.

Montrez que LIVRAISON-CIRCULAIRE est FPT en le paramètre $n + k$.

Note : je ne sais pas si c'est FPT en le paramètre k seulement. Si vous trouvez, dites-le moi!

Exercice 13. Considérez la variante suivante de LIVRAISON-CIRCULAIRE. On a la même entrée qu'au problème précédent, mais on veut choisir un chemin pour chaque paquet en minimisant le nombre d'arêtes de charge 2 ou plus. Montrez que ce problème est résoluble en temps polynomial.

Note : cet exercice n'est pas un problème FPT, mais il entraînera tout de même votre "mode de réflexion algorithmique".

Si vous aimez les défis, une version paramétrée serait la suivante : décider s'il est possible d'avoir au plus k arêtes de charge c ou plus, avec $k + c$ comme paramètre. Je ne sais pas si c'est FPT.