

Correction TD 03 – Transversal

Exercice 2.

Transversal (vertex cover)

4. Donner un algorithme prenant en entrée un graphe G et un entier k , et qui teste si G possède un transversal de taille k ou moins. Évaluer sa complexité.

Attention ce n'est pas suffisant de comparer k au transversal minimal retourné par l'algorithme précédent, car un transversal peut être minimal mais pas minimum !

L'algorithme consiste à tester tous les sous-ensembles de k sommets (moins ce n'est pas nécessaire). Il y a $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ sous-ensembles de taille k . On sait que $\binom{n}{k} < 2^n$. Avec un peu d'astuce comme à l'exercice 2 du TP 1 pour borner inférieurement $n!$ par $(\frac{n}{2})^{\frac{n}{2}}$, on peut montrer que $\sqrt{2}^n < \binom{n}{\frac{n}{2}}$ donc dans le pire cas c'est exponentiel.

Un algorithme simple consiste à générer tous les remplissages d'un tableau de n bits (ou toutes les décompositions binaires des entiers de 0 à $2^n - 1$), et pour chaque remplissage contenant k bits à 1, tester le potentiel transversal correspondant. Complexité $\mathcal{O}(2^n n^2)$ en analysant grossièrement, et plus finement $\Theta(2^n)$ pour la génération de tous les sous-ensembles plus $\Theta(n^2)$ pour chaque test, ce qui donne $\Theta(2^n + \binom{n}{k} n^2)$. On peut atteindre $\mathcal{O}(\binom{n}{k} n^2)$ en énumérant plus efficacement seulement les sous-ensembles de taille k .

En cours on a vu les bornes $(\frac{n}{k})^k \leq \binom{n}{k} \leq (\frac{ne}{k})^k$ qui sont intéressantes pour comparer les coefficients binomiaux aux fonctions exponentielles.

5. Supposer que nous ayons accès à une implémentation `stable` (G, p) résolvant efficacement le problème de décider si G possède un ensemble de sommets indépendants (sans aucun lien entre eux) de taille p ou plus. Utiliser un appel à `stable` pour résoudre le problème précédent de décider si G possède un transversal de taille k ou moins.

Il s'agit d'une réduction (concept bientôt abordé en cours), qui permet de comparer la difficulté de deux problèmes. Ici, résoudre `stable` va nous permettre de résoudre transversal (au prix de très peu de travail supplémentaire), donc on en déduit que résoudre `stable` est au moins aussi difficile que résoudre transversal.

Pour voir le lien entre `stable` et transversal, on peut se convaincre en essayant sur quelques exemples que si $X \subseteq V$ est un stable, alors son complémentaire $V \setminus X$ est un transversal. En effet (on démontre les contraposées), si $V \setminus X$ n'est pas un transversal alors une arête non couverte par $V \setminus X$ devrait se trouver entre deux sommets de X donc X ne serait pas un stable. Réciproquement, si X n'est pas un stable alors une arête entre deux sommets de X ne serait pas couverte par $V \setminus X$ donc $V \setminus X$ ne serait pas un transversal. On en déduit l'algorithme suivant, qui est cohérent avec $\leq k$ (minimisation difficile pour transversal) et $\geq p$ (maximisation difficile pour `stable`) :

```
transversal( G , k ) :
    retourner stable( G , G.n - k )
```

Si l'appel à `stable` est efficace, alors l'algorithme `transversal` est efficace.