TDSSP - L3

Josua Philippot

26 mars 2020

Exercice 1

Par définition, l'algorithme qui répond au problème d'optimisation est un algorithme qui donne toujours le meilleur résultat. Or si on est capable de d'exhiber une réponse optimisée, on sait s'il y a une réponse ou non (autrement c'est absurde). On pourrait donc imaginer l'algorithme suivant :

```
/* l'algorithme ci-dessous prend en entree F une chaine de caracteres
et retourne un entier K, la taille de la plus petite sous-chaine */

AgloOpti(string F) {
int K = 1
TantQue(AlgoDecision(F,K)){
/*On teste si la chaine F
trouve une solution avec l'algo decisionnel*/

    K++ // incrementation de K, la longueur de la sous sequence opti.
    }
    retourner K;
}
```

On a donc un algorithme qui se base sur le résultat du problème réduit à un problème décisionnel. On peut donc en conclure que le problème d'optimisation est au moins aussi difficile que le problème décisionnel (puisqu'il est nécessaire de connaître la réponse au problème décisionnel pour avoir celle du problème opti).

Exercice 2

Pour répondre au problème, on peut imaginer l'algorithme suivant :

```
/* l'algorithme ci-dessous prend en entree une chaine de carateres
F et un entier K, et retourne un booleen res qui informe de la
presence d'une sous-chaine*/
```

```
AlgoDec(string F, int K){
bool res = false;
//on fait une premiere boucle qui parcours F
Pour i de 0  a tailleDe(F){
// et une seconde qui verifie si on a une sous-chaine
    TantQue(!res){

        Pour j de 0 a k{
            res = {...test si superchaine... }

            }
        }
    }
    return res;
}
```

Exercice 3

Une couverture minimale du graphe est {A,G,F,D}.

Exercice 4

Si on sait faire une transformation de vertex cover en SSP on peut alors appliquer l'algorithme SSP sur cette transformation. Ainsi si l'on connait un algorithme polynomial pour traiter les SSP, il suffit d'appliquer la transformation au vertex covers que l'on veut traiter pour les traiter de façon polynomiale.

Exercice 5

1. ε = {A,B,C,D,E,F,G}
2. {(ACAC,CACA),(ADAD,DADA),(AEAE,EAEA),(CFCF,FCFC),(EGEG,GEGE),(BGBG,GBGB), (DBDB,BDBD),(DCDC,CDCD),(EDED,DEDE),(AGAG,GAGA),(FAFA,AFAF),(FGFG,GFGF),(BFBF,FBFB)}