# Leçon 4.
# Réductions polynomiales

Denis Trystram

October 8, 2012

Motivation : La réduction polynomiale est la technique de base pour prouver que des problèmes sont NP-complet.

## 1 Réductions

Le principe de l'étude de la complexité est de classifier les problèmes par rapport au critère de temps d'exécution sur une machine de Turing. Les deux classes $\mathcal{P}$ et $\mathcal{NP}$ que nous avons définies ne semblent pas assez fines pour discriminer la difficulté des problèmes. Nous aimerions introduire une relation d'ordre sur les problèmes (langages), signifiant qu'un problème est plus facile qu'un autre (que le langage correspondant est plus facile à décider que l'autre). Cette relation d'ordre est définie par la *réduction polynômiale*.

**Définition 1** *Soient $L_1$ et $L_2$ deux langages sur un alphabet $\Sigma$. Une fonction $\tau$ de $\Sigma^*$ vers $\Sigma^*$ est une réduction de $L_1$ vers $L_2$ ssi*

$$\forall x \in \Sigma^*, \quad x \in L_1 \;\Leftrightarrow\; \tau(x) \in L_2$$

Si la transformation $\tau$ est polynômiale, on dit que la réduction est polynômiale. Cette réduction est également appelée *réduction de Karp*.
Il existe d'autres réductions sur $\mathcal{NP}$, par exemple la réduction de Turing plus générale que celle de Karp au sens où elle permet des appels à des opérations plus sophistiquées. On peut également définir d'autres réductions sur d'autres classes de problèmes : par exemple sur NL, qui est l'ensemble des problèmes pour lesquels il existe une exécution utilisant un nombre poly-logarithmique de cases mémoire sur une machine de Turing non-déterministe.

Du point de vue des problèmes, $\tau$ est une réduction polynomiale du problème $\Pi$ au problème $\Pi'$ si elle est une réduction polynomiale entre les langages correspondants. La réduction $\tau$ transforme toutes les instances positives $\Pi$ en instances positives de $\Pi'$, et toutes instances négatives de $\Pi$ en instances négatives de $\Pi'$. L'existence d'une réduction polynômiale de $\Pi$ vers $\Pi'$ montre que $\Pi'$ est au moins aussi difficile que $\Pi$. En effet, si $\Pi$ peut être résolu en temps polynomial, alors $\Pi'$ peut l'être aussi; si par contre $\Pi$ requiert un temps exponentiel, alors $\Pi'$ ne peut être résolu par un algorithme polynomial. Notons bien que le sens premier de la réduction de $\Pi$ vers $\Pi'$ est encore plus fort : une réduction prouve qu'à une transformation polynomiale près des instances, c'est-à-dire à un codage naturel près de $\Pi$, le problème $\Pi$ est simplement un sous-problème de $\Pi'$.
D'un point de vue algorithmique, une réduction est un *preprocessing* des instances du problème $\Pi$, qui permet d'utiliser tout algorithme polynomial de résolution pour $\Pi'$ pour résoudre $\Pi$ en temps polynomial.

On notera $\Pi \propto \Pi'$ si il existe une réduction polynomiale de $\Pi$ vers $\Pi'$. Nous dirons que $\Pi$ se réduit à $\Pi'$.

**Propriété 1** *La réduction polynômiale est un pré-ordre sur les langages : c'est une relation réflexive et transitive.*

Cette propriété n'est pas difficile à montrer. Elle est importante car elle permettra d'établir des réductions à partir de problèmes de référence. On notera par $\equiv$ l'équivalence associée : $\Pi \equiv \Pi'$ si et seulement si $\Pi \propto \Pi'$ et $\Pi' \propto \Pi$.

# 2 Exemples

## 2.1 Deux réductions simples

Détaillons un exemple de réduction polynomiale du problème de décision qui cherche à déterminer si un graphe possède une chaîne hamiltonienne à partir du problème du cycle hamiltonien.

HamiltonianPath (HP)
**Instance.** un graphe $G = (V, E)$ et une paire de sommets disjoints $x$ et $y$
**Question.** Est-ce que le graphe possède une chaîne hamiltonienne dans $G$ entre $x$ et $y$ (chaîne qui passe une fois et une seule par tous les sommets du graphe)?

Pour montrer que nous avons la réduction HP $\propto$ HC, considérons une instance de HP, c'est-à-dire un graphe $G$. À partir de $G$ nous construisons une instance particulière $\tau(G)$ de $HC$ en ajoutant à $G$ un nouveau sommet $x$ relié à tous les autres : $\tau(G) = (V \cup \{x\}, E \cup (x, y) \forall y \in V)$.

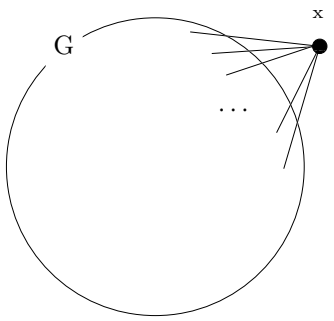


Figure 1: Réduction $HP \propto HC$.

La transformation $\tau$ est évidemment polynomiale. Montrons que c'est une réduction, c'est-à-dire que $G$ admet une chaîne hamiltonienne si et seulement si $\tau(G)$ possède un cycle hamiltonien.
Si $G$ possède une chaîne hamiltonienne $\varphi$, alors le cycle $x\varphi x$ est hamiltonien dans $\tau(G)$. Réciproquement, si $\tau(G)$ admet un cycle hamiltonien, son sous-graphe privé de $x$, $G$, possède une chaîne hamiltonienne.

On peut également établir que nous avons la réduction du cycle hamiltonien au circuit hamiltonien, $HC \propto HP$. Ce résultat n'est pas aussi immédiat, même s'il semble facile de déduire une chaîne hamiltonienne à partir d'un cycle hamiltonien, cela ne suffit pas à définir une réduction. Considérons la transformation $\tau$ suivante d'une instance $G$ de HC vers une instance $\tau(G)$ de HP :
On duplique un sommet quelconque $x$ du graphe $G$ en $x'$, puis on relie $x$ et $x'$ respectivement à deux nouveaux sommets $y$ et $y'$ comme c'est indiqué dans la figure 2. Cette transformation est polynomiale. C'est une réduction : $G$ admet un cycle hamiltonien si et seulement si $\tau(G)$ possède une chaîne hamiltonienne.
En effet, si $G$ possède un cycle hamiltonnien, la chaîne formée de l'arrête $y$ à $x$, de la partie du cycle jusqu'à un voisin de $x$, puis des arrêtes de ce voisin à $x'$ et celle de $x'$ à $y'$ est une chaîne hamiltonienne de $\tau(G)$. Réciproquement, si il existe une chaîne hamiltonienne $\varphi$ dans $\tau(G)$, elle est nécessairement de la forme $yx\psi x'y'$. Alors $x\psi x$ est un cycle hamiltonien sur $G$.

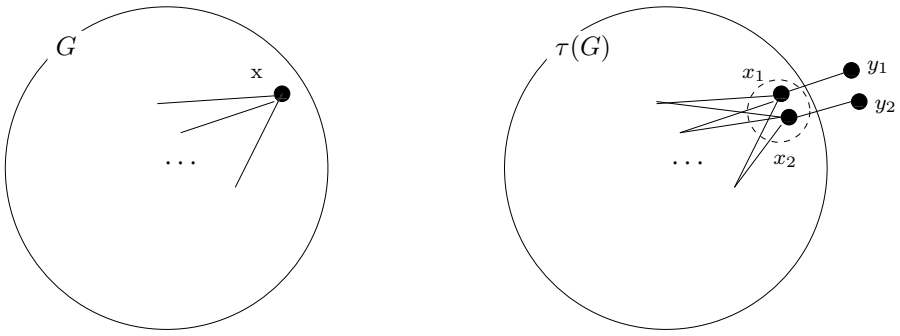


Figure 2: Réduction $HC \propto HP$.

Nous venons de montrer que les problèmes $HC$ et $HP$ sont équivalents : nous pouvons transformer tout algorithme polynomial pour $HC$ en un algorithme polynomial pour $HP$, et inversement.

## 2.2 Voyageur de commerce

Considérons deux problèmes voisins, les versions de décision des problèmes du voyageur de commerce (D-TSP) et de l'existence d'un circuit hamiltonien dans un graphe.

D-TSP
**Instance.** un ensemble $V$ de villes, la matrice des distances inter-villes $(d_{i,j})$ et une constante $k$.
**Question.** Déterminer s'il existe un parcours fermé, passant par toutes les villes, de longueur inférieure à $k$.

Il est facile de démontrer que ces deux problèmes admettent un algorithme de résolution exponentiel en considérant toutes les permutations possibles (en comparant la somme de la longueur de tous les tours pour D-TSP). A ce jour, il n'existe pas d'algorithmes polynomiaux connus pour résoudre ces problèmes, et nombreux sont les informaticiens qui pensent qu'il n'en existe sans doute pas... C'est là une des célèbres questions ouvertes de l'Informatique. On peut réduire D-TSP à partir de HC (HC $\propto$ D-TSP).

Décrivons un algorithme polynomial qui transforme une instance quelconque de HC en une instance positive de D-TSP si et seulement si l'instance de

HC est positive.
La réduction est la suivante : l'ensemble des villes correspond aux sommets du graphe $G$, les distances sont données par $d_{i,j} = 1$ si $i$ et $j$ sont reliés, 2 sinon. La constante $k$ est égale au nombre de villes (cardinal de $V$).
Cette transformation est polynomiale de manière évidente, de plus elle transforme les instances positives de D-TSP en instances positives de HC. En effet, la solution de D-TSP possède par définition des arêtes de coût unitaire puisque la longueur totale est $k$, c'est aussi une solution de HC.

### 2.3 Réduction ou sous-problémes ?

La réduction est la manière canonique pour prouver qu'un problème est NP-complet. On n'utilise pas toujours les réductions mais il suffit souvent d'extraire un sous-problème qui est lui-même NP-complet (en limitant les instances à une classe plus ciblée).

## 3 Ce qu'il faut retenir

Définition des réductions (relation d'ordre entre problèmes).
Construire une réduction.
Attention aux pièges : encodage polynomial (exemple pour le problème PRIME) et transformation doit être polynomiale.