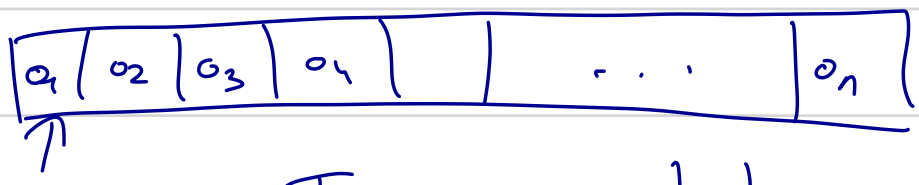
Problèmes NP-complets

Tri : $\Theta(n \log n)$ pour le tri fusion
Multiplication de matrices : $\Theta(n^{\log_2 7})$ par Strassen
Sac à dos 0-1 : $\Theta(W \times n)$ par prog. dynamique

On dit qu'un problème est résolvable <u>en temps polynomial</u> (déterministe) s'il existe un algorithme pour résoudre ce pb et la complexité asymptotique en temps est en $O(P(|I|))$ où P est un polynôme et $|I|$ est la taille de l'entrée du pb (le nombre de bits nécessaire pour écrire l'entrée).

Tri



T bits $\quad T = \max_{1 \leq i \leq n} |a_i|$

Complexité : $\Theta(n \log n \times T) = O((nT)^2)$ Taille d'entrée : $\Theta(n \times T)$

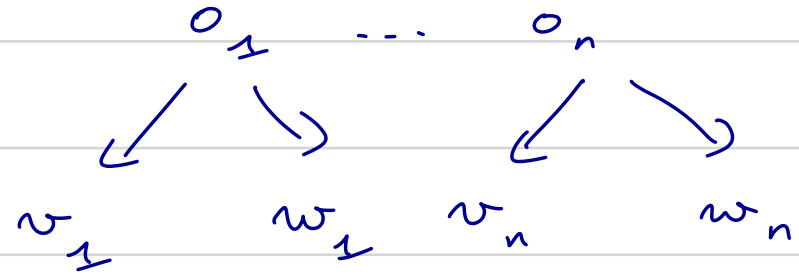
Ouf, le problème du tri est résolvable en temps polynomial.

Mult. de matrices

Complexité $\Theta(n^{\log_2 7} \times T) = O((n^2 \times T)^2)$ Taille d'entrée : $O(n^2 \times T)$

Problème en temps polynomial

Sac à dos



W

$v_i \leq V$ $\quad$ $w_i \leq W$

pour v_i pour w_i pour W

Taille d'entrée $|I| = \textcircled{H}\left(n(\log V + \log W) + \log W\right)$

$= \textcircled{H}\left(n(\log V + \log W)\right)$

Complexité : $\textcircled{H}\left(n \times W \times (\log V + \log W)\right)$

La prog. dynamique ne donne pas une résolution en temps polynomial du pb du sac à dos. On dit parfois que l'algo est pseudo-polynomial (polynomial en les nombres donnés en entrée).

	o_1	o_2	o_3
v_i	25	50	75
w_i	100	10000	1000000

$W = 1000000$

P et NP

Problèmes de décision : problème dont la sortie est soit Vrai soit Faux

Variantes de décision des pb plus haut :
- Est-ce que le tableau en entrée est trié ? $O(n \times T)$
- Vérifier si une matrice C est le produit de deux matrices A et B. $O(n^{\log_2 7} \times T)$
- Existe-t-il $S \subseteq \{1, \dots, n\}$ t.q. $\sum_{i \in S} w_i \leq W$ et $\sum_{v_i \in S} v_i \geq V$ pour un V donné en entrée.

Le pb de décision a une cplxité tjrs au moins aussi faible que le pb initial (de calcul)

La classe de complexité P (ou PTIME) est la classe de l'ensemble des problèmes de décision résolvables en temps polynomial (déterministe).

On introduit un nouvel opérateur (non déterministe)

Deviner dans les algorithmes qui devine une structure de taille polynomiale en la taille de l'entrée.

Guess

Sac à dos non déterministe

Entrée : (v_i), (w_i), W, V

Sortie : Vrai s'il existe $S \subseteq \{1 \dots n\}$ t.q.

$$\sum_{i \in S} v_i \geq V \text{ et } \sum_{i \in S} w_i \leq W$$

Deviner un sous-ensemble $S \subseteq \{1, .., n\}$

Calculer $\sum_{i \in S} w_i$ et $\sum_{i \in S} v_i$

Si $\sum_{i \in S} w_i \leq W$ et $\sum_{c \in S} v_i \geq V$ alors

Renvoyer Vrai

Sinon

Renvoyer Faux

On dit qu'un algorithme non-déterministe (algo utilisant Deviner) résout un problème si :

- quand il existe un guess qui conduit l'algorithme à renvoyer Vrai alors le pb a pour solution Vrai
- sinon, le pb a pour solution Faux.

La classe de complexité NP (nondeterministic polynomial-time) est la classe de l'ensemble des problèmes de décision résolvables en temps polynomial par un algo non-déterministe.

SacÀDos $\in$ NP

$\boxed{P \subseteq NP}$

NP $\neq$ non polynomial

$\boxed{NP \subseteq EXPTIME}$: ts les pb dans NP sont résolvables par un algo déterministe (force brute) en temps exponentiel

P vs NP

$P \overset{?}{=} NP$

$P \overset{?}{\neq} NP$

(7 pb du millénaire
1 M$)

Déf. Une réduction (de Karp, en temps polynomial) entre un problème de décision P et un problème de décision P' est un algorithme A déterministe en temps polynomial qui prend en entrée une entrée du pb P et produit en sortie une entrée de P' t.q. $P'(A(I)) = P(I)$ pour tout I. On écrit $P \leq P'$.

Déf. Un problème Q est dit NP-difficile si pour tout problème $Q' \in NP$, $Q' \leq Q$.
Un problème Q est NP-complet si Q est NP-difficile et $Q \in NP$.

Thm (admis) : SacADos est NP-complet.