La complexité

Complexité d'un algorithme

- Trouver un moyen d'estimer le temps d'exécution d'un algorithme donné
- Ce temps peut s'exprimer formellement en fonction du nombre d'instructions exécutées par une machine de Turing déterministe utilisant au moins un alphabet à 3 symboles (B,0,1)
- Ne pas confondre la complexité d'un algorithme et la complexité d'un problème

Un problème important

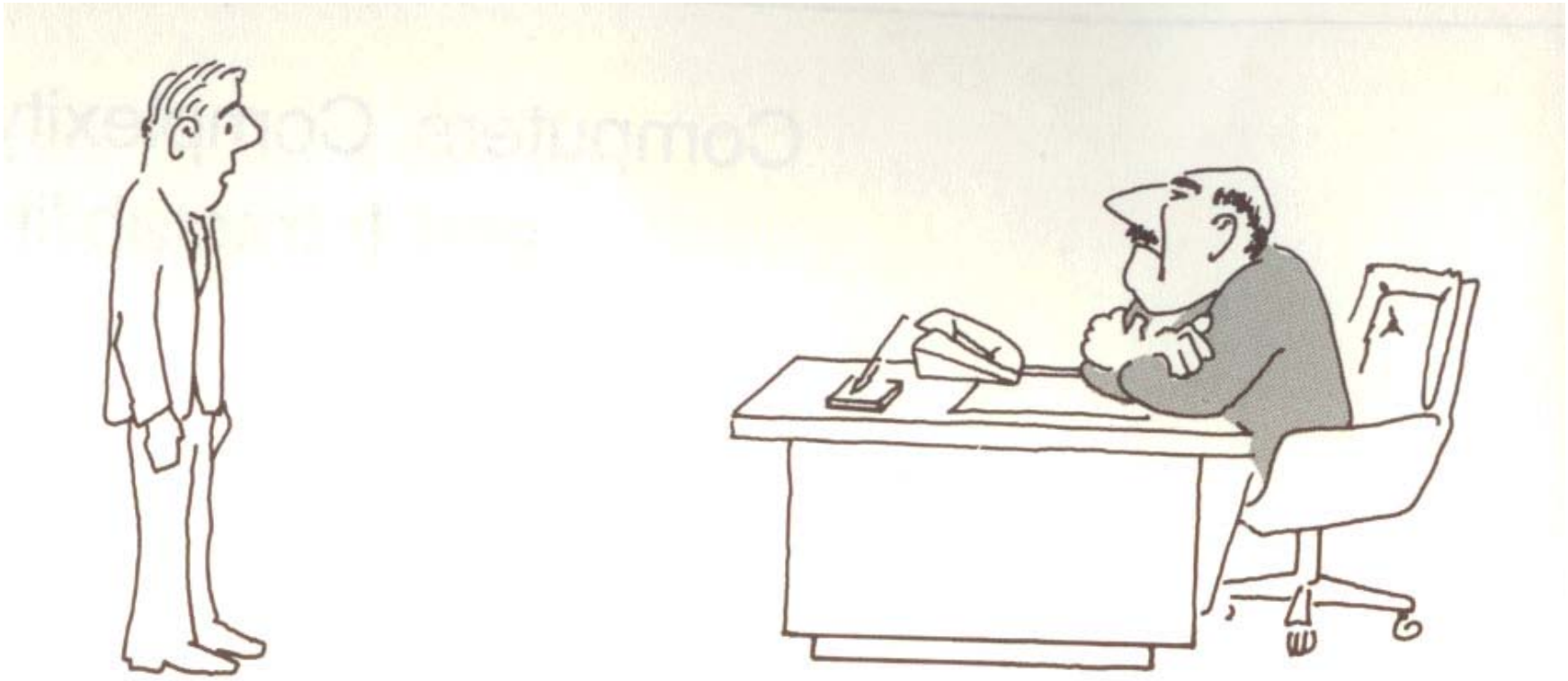
	10	20	30	40	50	60
n	0.00001s	0.00002s	0.00003s	0.00004s	0.00005s	0.00006s
N log(n)	0.00002s	0.00006s	0.00010s	0.00015s	0.00020s	0.00025s
N^2	0.0001s	0.0004s	0.0009s	0.016s	0.025s	0.036s
N^3	0.001s	0.008s	0.027s	0.064s	0.125s	0.216s
N^5	0.1s	3.2s	24.3s	1.7m	5.2m	13.0m
2^N	0.001s	1.0s	17.9m	12.7j	35.7a	366c
3^N	0.059s	58m	6.5a	3855c	2^E8c	1.3^E13c

Algorithmes polynomiaux

- Algorithmes exécutables en temps polynomial sur une machine de Turing déterministe
- Opérations élémentaires (addition, multiplication, etc)
- Algorithmes de tri de tableaux
  - Tri a bulles
  - Shell-Metzner
- Si l'on a trouvé un algorithme polynomial, on sait que le problème est polynomial
- Quid si l'on ne peut trouver d'algorithme polynomial?

Première cas

Je ne peux pas trouver de solution



Deuxième cas

- Il n’existe pas d’algorithme polynomial

Troisième cas

- Je ne trouve pas de solution, mais je ne suis pas le seul

Problème NP

- Problème non déterministe polynomial.
- Problème soluble en un temps polynomial sur une machine de Turing non déterministe

Réduction polynomiale

- Il s'agit d'une opération permettant de transformer en temps polynomial les données d'un problèmes P1 en données pour un problème P2.
- On dit alors que P1 est réductible en P2.

Problème NP-complet

- p est NP-complet si:
  - p est un problème NP
  - Tout problème de NP peut se réduire polynomialement en p
- La solution d'un problème NP complet peut-être vérifié en temps polynomial
- S'il existe p ∈ NPC tel que p ∈ P alors tout problème NP est soluble en temps polynomial.

Théorème de Cook

- Première démonstration de l'existence d'un problème NPC : Cook en 1971
- Problème SAT: soit une formule booléenne F sous forme conjonctive normale, par exemple:
  - F=(a□□b□□c)□ (c□d□□e)
- Existe-t-il une affectation de valeurs pour (a,b,c,d,e) telle que F soit vraie?

P=NP?

- Problème ouvert
- 1M$ de prix pour trouver une preuve
- On pense aujourd'hui que P est probablement différent de NP

Le problème du sac à dos

- Soit un ensemble E d'objets de poids (p_i) et une masse m fixé.
- Existe-t-il un sous-ensemble S de E telle que la somme des poids des objets de S soit égale à m
- Problème important en théorie de la cryptographie

Le problème du voyageur de commerce (TSP)

- Un voyageur de commerce doit visiter n villes. On connait la distance d_{ij} entre chaque ville.
- Il faut construire le circuit de longueur minimale passant par toutes les villes.
- Problème fondamental en logistique (réseau électrique, téléphonique, transport de marchandises...)

Planification de tâches indépendantes

- On dispose de m machines identiques devant exécuter n tâches s'exécutant en un temps t_i
- Trouver l'affectation optimale des tâches sur l'ensemble des machines.

Schéma d'approximation

- Dans l'état actuel des connaissances, il n'existe pas d'algorithmes permettant de résoudre un problème NP en un temps « raisonnable »
- Il faut chercher des méthodes approchées résolvant les problèmes NP-complet

-approximation

- Soit un problème p résolu de façon optimale par l'algorithme OPT. On dit que l'algorithme A est une -approximation de OPT si on a:
  - x |(A(x)-OPT(x))/OPT(x)|<
- On dit que l'on possède un schéma d'approximation pour p si pour tout on peut trouve un A résolvant p

Le problème du voyageur de commerce

- Il n'existe pas d' α-approximation du TSP si P $\neq$ NP.
- Si on considère la restriction au α-voyageur de commerce alors il existe une α-approximation
  - Algorithme de Prim: α=1, complexité n^2
  - Cycle eulérien: α=0.5, complexité n^3

Problèmes pseudo-polynomiaux

- Problèmes NP qui deviennent polynomiaux si on borne la taille des données
- Le problème de partitionnement est pseudo-polynomial.
- Le TSP n'est pas pseudo-polynomial

L'algorithme de Diffie-Hellman

- Premier exemple d'algorithme de cryptage à clef publique (1975)
- Sa robustesse vient du fait que le problème du sac à dos est NP-complet
- Cassé par Rivest-Shamir Addelman qui montreront que la restriction de knapsack utilisé par Diffie-Hellman n'est pas NP.

Algorithme de Diffie-Hellman

- Prendre une suite super-croissante (a_i) et un nombre $N > \sum a_i$
- Prendre $A < N$ tel que $pgcd(A,N) = 1$
- Calculer $b_i \equiv A \times a_i \ [N]$
- Les b_i sont la clef publique, (a_i,N) est la clef privée
- On transforme un message en suite de bits s_i et on envoie $\sum (s_i \times b_i)$

Exemple:

- {a1, a2, · · · , a8} = {3, 7, 15, 31, 63, 151, 317, 673}
- N=1511, A=643 et 1/A≡47[1511]
- {b1, b2, · · · , b8} = {418, 1479, 579, 290, 1223, 389, 1357, 593}
- 10011010 -> c = 418 + 290 + 1223 + 1357 = 3288
- 3288 x 47 [1511]≡414≡317+63+31+3

Algorithme RSA

- Choisir p et q premiers. Calculer $n = pq$ et $\varphi = (p-1)(q-1)$
- Trouver e tel que $1<e<\varphi$ et $pgcd(e,\varphi)=1$
- Calculer $d \equiv e^{-1} [\varphi]$
- La clef publique est (n,e), la clef privée d
- Cryptage: $c \equiv m^e [n]$
- Décryptage: $m \equiv c^d [n]$
- Repose sur l'impossibilité pratique de factoriser n pour trouver φ (problème NP)

Algorithme RSA: exemple

- On prend p=13, q=17. Alors n=pq=221, et φ=(p-1)(q-1)=192
- On choisit e = 7
- $d \equiv 7^{-1} [192] \equiv 55 [192]$
- Cryptage: m=6, $c \equiv 6^{7} [221] \equiv 150 [221]$
- Decryptage: $150^{55} [221] \equiv 6$

Autres classes de complexité: co-NP

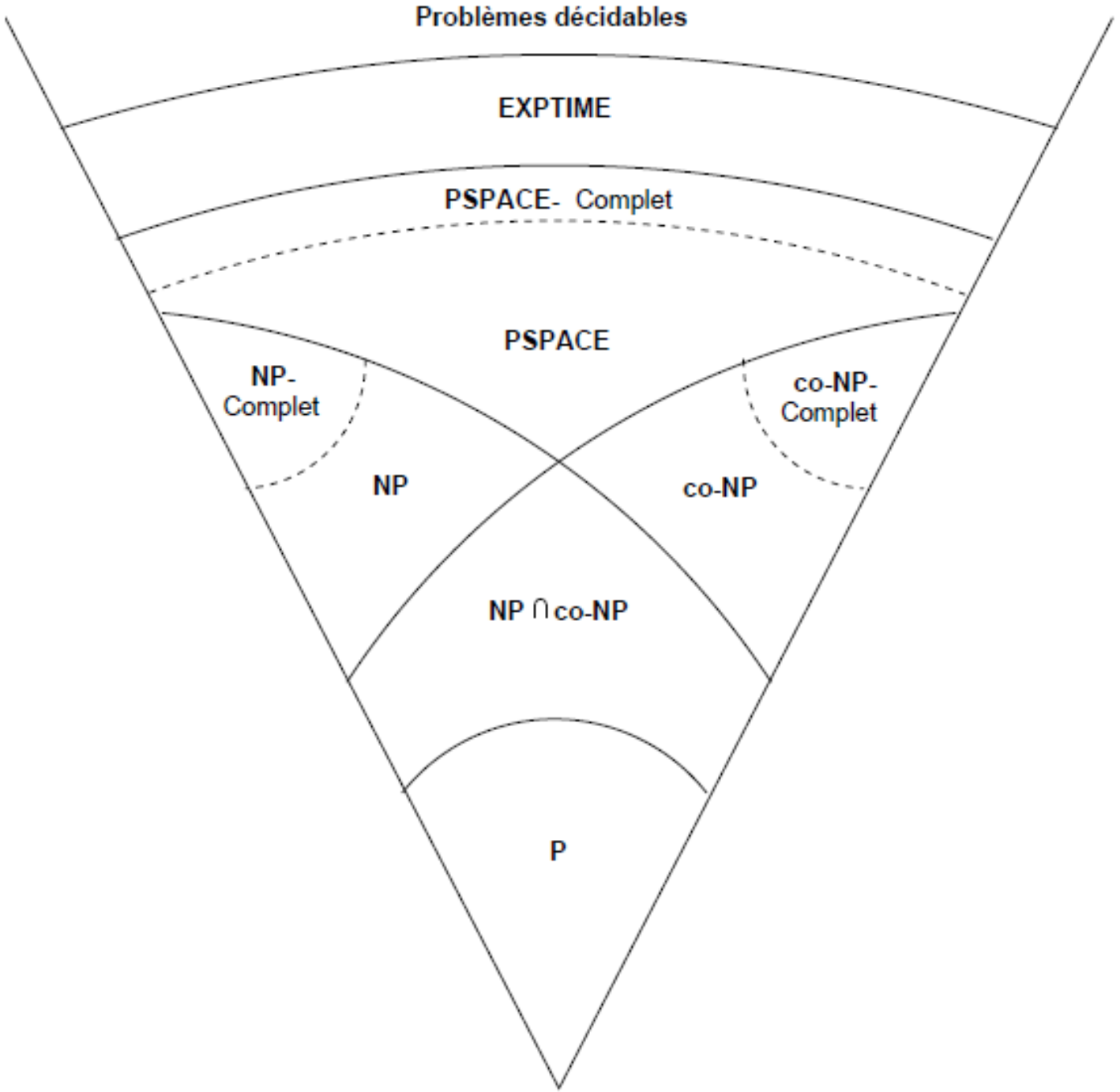
- Co-NP: ensemble des problèmes p tel que le complément de p est dans NP.
- Le complément d'un problème p est le même problème dans lequel on transforme toutes les réponses « oui » en réponses « non »
- Soit p le problème « n est il premier? » Son complément est « n est-il composite? »

PSPACE et NPSPACE

- PSPACE: ensemble des problèmes résolubles en espace polynomial par une machine de Turing déterministe.
- NPSPACE: ensemble de problèmes résolubles en espace polynomial par une machine de Turing non-déterministe.
- Propriété: PSPACE=NPSPACE.

EXPTIME

- Ensemble des problèmes résolubles par une MT déterministe en un temps borné par $2^{p(n)}$ avec p polynome.
- Et bien d'autres :
  - NL (Non deterministic logarithmic space)
  - L (Logarithmic Space)
  - ....


Problèmes décidables
EXPTIME
PSPACE- Complet
PSPACE
NP-
Complet
co-NP-
Complet
NP
co-NP
NP ∩ co-NP
P