UE 34: Actualités des Mathématiques

David BOSSEZ **Mai 2013**

Travail de diffusion, de vulgarisation d'un problème de recherche mathématique à des élèves de Lycée.

Thème choisi (Proposition de Lionel Vaux) : « P versus NP ? »

Public visé : élèves de Terminale scientifique

Sommaire du rapport

Introduction

I. Choix du sujet

II. Méthodologie d' élaboration du contenu

III. Contenu à proposer en classe de Terminale scientifique

Bibliographie

Introduction

Le citoyen français non amateur des mathématiques n'a aucune raison de connaître les problèmes qui font l'actualité de cette discipline. Pourtant, il arrive que la résolution de certains problèmes mathématiques fasse la une des journaux. Le contenu de ces articles met généralement en avant les mathématiciens et non les problèmes en eux-mêmes, leur énoncé est d'ailleurs en général passé sous silence. Pour exemple, citons la résolution du théorème de Fermat par le mathématicien britannique Andrew John Wiles en1994 dont les articles des quotidiens non spécialisés ont essentiellement souligné la ténacité et la performance ou encore la résolution de la conjecture de Poincaré en 2002 par le mathématicien russe Grigori Perelman dans lequel l'intérêt se porte sur la personnalité atypique du mathématicien et son refus de toute récompense honorifique ou financière.

Les élèves, quant à eux, posent souvent deux questions, « à quoi ça sert ? » et « que font les mathématiciens ? », à des moments souvent inadaptés pour qu'une réponse satisfaisante puisse être donnée.

Ainsi à la fascination du tout un chacun pour les exploits de certains mathématiciens, les élèves ajoutent une curiosité pour le travail des mathématiciens et l'utilité de leur travail pour la société.

Un travail, avec les élèves, centré sur un problème d'actualité pour les mathématiques est un moyen de répondre à cette curiosité.Ce court mémoire est destiné à présenter un contenu possible d'un tel travail. Il sera l'occasion, d'une part, de présenter un problème qui préoccupe les mathématiciens d'aujourd'hui ainsi que les enjeux de ce problème pour la société et, d'autre part, de réinvestir certaines connaissances apprises par les élèves dans leur cursus scientifique du secondaire.

I. Choix du sujet

A l'occasion de la présentation des sujets possibles pour la validation de L 'UE « Actualités des mathématiques », Lionel Vaux a évoqué le problème P = NP comme thème de travail possible. Un premier travail documentaire autour de ce problème m'a permis de conclure à la possibilité d'en faire un travail de diffusion auprès des élèves de Terminale scientifique. Les programmes des classes scientifiques de lycée comportent, en effet, une étude de l'« algorithmique », des fonctions, des suites numériques, objets de connaissance mathématiques qui me semblent indispensables à un premier abord du problème P=NP.

Mon expérience d'enseignant m'a appris d'une part que les élèves étaient peu avides de conférences et d'autre part que les apprentissages se faisaient davantage par la pratique que par l'écoute. C'est pourquoi j'ai souhaité que la compréhension du problème se fasse par la mise en activité des élèves autour de situations faisant appel à des connaissances apprises dans leur cursus. Mon travail a essentiellement consisté dans le choix de ces situations.

II. Méthodologie d' élaboration du contenu

Mon travail documentaire autour du problème « P = NP ? » a permis de mettre en évidence plusieurs niveaux d'énoncés de ce problème. Nous allons voir que selon les sources consultées la formulation du problème est très variable tant dans son contenu quand sa forme. Quel énoncé retenir ?

Le problème selon Le Figaro (11/08/2010)

> « Tentons d'expliquer les choses de manière simple. Dans la théorie mathématique de la complexité, on distingue deux grandes familles de problèmes. Les problèmes 'faciles à résoudre' (P) et les problèmes dont les solutions sont 'faciles à vérifier' (NP). On comprend intuitivement qu'un problème 'facile à résoudre' a des solutions 'faciles à vérifier' (2 + x = 4 : la solution est facile à trouver, x= 4 - 2= 2, et facile à vérifier 2 + 2 est bien égal à 4). Cela veut dire que l'ensemble P est inclus dans NP. Par contre, personne n'a jamais réussi à répondre à la question inverse : un problème dont la solution est 'facile à vérifier' est-il oui ou non 'facile à résoudre' ? ».

Il y aurait donc des problèmes plus ou moins « faciles » à résoudre et à vérifier, le critère de « facilité » n'étant pas précisé. On voit difficilement en quoi il s'agit d'un problème d'actualité pour les mathématiques.

Le problème selon Wikipédia

> « Il s'agit plus formellement de savoir si la classe de complexité des problèmes de décision admettant un algorithme de résolution s'exécutant en temps polynomial sur une machine de Turing est équivalente à la classe de complexité des problèmes de décision dont la v*érification* du résultat, une fois celui-ci connu, demande un temps polynomial. ».

Dans cette présentation, moins sommaire, apparaissent des notions nouvelles, notamment d'algorithmes, de machine de Turing et de temps polynomial qui permettent de situer le problème « P=NP » dans le domaine de l'informatique théorique.

La présentation officielle du problème par le Clay Institute

Deux articles introductifs

Sur le site internet de l'Institut Clay, on trouve un article introductif aux problèmes du millénaire et notamment du problème P=NP.

Cette introduction propose un exemple de problème : une liste de 400 étudiants est donnée, dont certaines paires sont définies comme « incompatibles ». Il faut générer une liste de 100 étudiants, parmi ces 400, ne contenant aucun couple d'étudiants incompatibles. A partir d'une liste donnée des 100 étudiants choisis, il est facile de vérifier qu'elle convient. Mais générer une telle liste est actuellement irréalisable en pratique et le demeurera car le nombre de combinaisons possibles est

extrêmement grand : « the total number of ways of choosing one hundred students from the four hundred applicants is greater than the number of atoms in the known universe! » à moins de trouver un algorithme beaucoup plus efficace.

En généralisant cet exemple le texte introductif présente ainsi l'essence du problème P versus NP :

> « In fact, one of the outstanding problems in computer science is determining whether questions exist whose answer can be quickly checked, but which require an impossibly long time to solve by any direct procedure. Problems like the one listed above certainly seem to be of this kind, but so far no one has managed to prove that any of them really are so hard as they appear, i.e., that there really is no feasible way to generate an answer with the help of a computer. »
> Notre Traduction : "En fait, un des problèmes en suspens dans l'informatique est de déterminer s'ils existe des questions dont la réponse peut être rapidement vérifiée, mais qui sont telles que n'importe quelle procédure directe de résolution exige un temps trop important. Les problèmes comme celui exposé ci-dessus semblent certainement être de cette sorte, mais jusqu'ici personne n'a réussi à prouver que n'importe lequel d'entre eux n'est vraiment aussi difficile à résoudre qu'ils le semblent, c'est-à-dire, qu'il n'y a vraiment aucune façon faisable de produire une réponse avec l'aide d'un ordinateur. ».

Le texte précise que la formulation «P versus NP » est à attribuer à Stephen Cook et Leonid Levin et date de 1971. On trouve une autre brève référence au problème « P versus NP » sur la page annonçant officiellement l'attribution du Prix du millénaire à Grégory Perelman pour la résolution de la conjecture de Poincaré :

> « If it is easy to check that a solution to a problem is correct, is it also easy to solve the problem? This is the essence of the P vs NP question. Typical of the NP problems is that of the Hamiltonian Path Problem: given N cities to visit (by car), how can one do this without visiting a city twice? If you give me a solution, I can easily check that it is correct. But I cannot so easily (given the methods I know) find a solution. ».
> Notre traduction : « s'il est facile de vérifier que la la solution d'un problème est correcte, est-il aussi facile de résoudre ce problème ? C'est l'essence du problème P versus NP. Un exemple type de problème NP est celui de détermination d'un chemin Hamiltonien : Étant donné N villes à visiter (en voiture), quel chemin emprunter pour se rendre dans toutes les villes sans passer 2 fois par la même ville ? Si vous me donnez une solution je peux facilement vérifier qu'elle est correcte. Mais je ne peux pas aisément (avec les méthodes que je connais actuellement) trouver une solution. »

La présentation officielle par Stephen Cook

http://www.claymath.org/millennium/P_vs_NP/pvsnp.pdf

Le ton est tout autre que celui des textes introductifs.C'est une présentation formelle de la question qui situe le problème dans le cadre de l'informatique théorique : un ordinateur (computer) est modélisé par une machine de Turing. Les classes de problèmes P et NP sont définies et formalisées à l'aide de ce modèle. S'y ajoute la classe des problèmes NP-complets. Stephen Cook situe historiquement la question et évoque les orientations de recherche sur ce sujet.

Cette présentation, destinée aux mathématiciens, fait appel à des formalismes hors de portée des élèves de terminale scientifique.

Mon travail a pour cible les élèves de Terminale S. Il s'agit de situer l'intervention entre une

présentation introductive sommaire telle que celle proposée par l'article du Figaro et une présentation, telle que celle de Stephen Cook, rigoureuse mais bien trop formelle pour être accessible et pour intéresser le public visé. L'énoncé proposé par wikipédia m'a semblé la plus à même d'atteindre cet objectif : c'est cet énoncé que le travail que je propose a pour but de faire comprendre.
La notion de machine de Turing est simplement remplacée par celle de calculateur (computer), sans précision supplémentaire. La notion centrale, nouvelle pour les élèves de Terminale S, est celle de de complexité d'un algorithme.

Une notion centrale : la complexité algorithmique

Après lecture des programmes de Terminale S, il nous a semblé qu'il était tout à fait possible d'aborder la notion de fonction de complexité algorithmique : la notion d'algorithme est connue des élèves, et l'étude du comportement des fonctions constitue une partie essentielle de leur formation.
Les situations de travail que nous avons choisies ont donc pour principal objectif d'aborder cette notion centrale.

Un travail ambitieux mais réalisable en classe

Comme nous l'avons annoncé plus haut, nous avons choisi d'élaborer notre présentation comme un cours interactif, ponctué d'exercices destinés à faire comprendre les notions abordées par leur mise en situation, nous éloignant ainsi de la structure classique d'un exposé souvent conçu comme « un donné à voir et à entendre » : nous avons souhaité que ce travail soit « un donné à faire pour comprendre ».
Lors de la présentation orale de ce travail les enseignants présents, professeurs de l'UE et étudiants, ont attiré mon attention sur le fait que je sous-estimais le temps nécessité par un tel travail.
Il faut donc considérer qu'il pourrait prendre la forme d'un travail conduit sur plusieurs séances ou comme fil conducteur à suivre au fil d'un trimestre.

Le contenu de ce travail est décrit dans la partie suivante, telle qu'elle pourrait l'être face à des élèves de Terminale scientifique. Il suit le plan suivant :

1. **Présentation du contexte du problème**
2. **Énoncé du problème**
3. **Complexité d'un algorithme : de quoi parle-t-on ?**
   Deux types de tâches :
   – « Déterminer un algorithme de résolution d'un problème ».
   – « Déterminer une fonction de complexité de cet algorithme ».
4. **La classe P**
5. **La classe NP**
6. **Le problème « P versus NP »**

III. Contenu à proposer en classe de Terminale scientifique

Le contexte

Un défi parmi 7

- En 1900, David Hilbert avait présenté les problèmes ouverts à résoudre durant le 20eme siècle.
- **En mai 2000, le Clay Mathematics Institute** : **7 problèmes ouverts pour le nouveau millénaire**. A la clé, un prix de un million de dollars pour celui qui résoudra un des 7 problèmes.

Hypothèse de Riemann (constitue aussi l'un des problèmes de Hilbert)
Conjecture de Poincaré (résolue en 2003)
Problème P = NP (est un des principaux problèmes ouverts de l'informatique théorique.)
Conjecture de Hodge
Conjecture de Birch et Swinnerton-Dyer
Équations de Navier-Stokes
Équations de Yang-Mills

Un premier niveau accessible pour l'amateur

- Pour en comprendre l'essence : bagage mathématique du secondaire
- Il s'agit d'un problème qui concerne les algorithmes, plus exactement, leur efficacité à résoudre certaines tâches.

Une résolution qui résiste aux assauts des mathématiciens

- A ce jour, six des sept problèmes demeurent non résolus.
- Le problème P = NP est l'un d'eux. Sa résolution rapportera 1 million de dollars.

Le problème

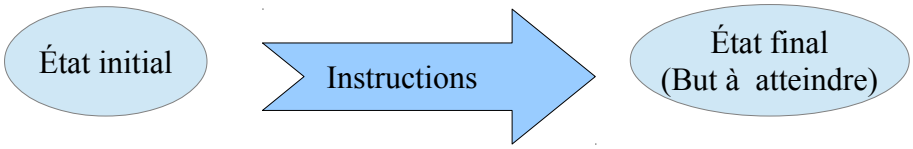
Un énoncé (tel que lu sur Wikipédia) :

« Il s'agit plus formellement de savoir si la classe de complexité **P** des problèmes de décision admettant un algorithme de résolution s'exécutant en temps polynomial sur une machine de Turing est équivalente à la classe de complexité **NP** des problèmes de décision dont la v*érification* du résultat, une fois celui-ci connu, demande un temps polynomial. ».

La complexité d'un algorithme

Algorithme

Un algorithme une suite bien définie d'opérations qui permettent d'arriver à un résultat et plus précisément nous dirons qu' u*n algorithme est* ***une liste d'instructions élémentaires*** *qui à partir d'****un état initial*** *bien connu permettent d'aboutir à* ***un état final*** *bien spécifié.*



Complexité d'un algorithme

- Parler de la complexité d'un algorithme **c'est s'intéresser à son efficacité.**
- Critère pris en compte : les ressources nécessitées par l'algorithme pour la machine qui le réalise :
  - en temps de calcul
  - en espace mémoire
- Causes de l'augmentation du coût : le nombre de données à traiter ou à stocker.
- C'est l'évolution du coût qui nous intéresse, avec l'augmentation des données.
  → Pas un coût précis (fonction de la machine utilisée …) : une estimation du coût

Pour le problème « P = NP »

- complexité en temps de calcul.

Récapitulatif :

Un problème est donné.

Pour le résoudre on utilise un certain algorithme.

On se demande quel sera l'évolution du coût de détermination de la réponse, en temps de calcul, avec l'augmentation du nombre de données.

Déterminer la complexité d'un algorithme (5 problèmes)

Objectif : se familiariser avec la notion de complexité d'un problème, par la pratique.

P1 : Compter le nombre de 1 dans une suite de 0 et de 1 de longueur n.

Algorithme choisi :
– une machine accomplit ce travail : dotée d'un tête de lecture et d'un compteur de lecture.
– la tête de lecture découvre les chiffres de la liste un à un ce qui incrémente un compteur à chaque fois qu'un chiffre 1 est lu
Ces 2 opérations, lecture et incrémentation sont nos opérations élémentaires.

On considère comme unité de temps le temps d'exécution d'une opération élémentaire : quel est le coût maximum de cet algorithme ?

Réponse envisagée

1) Nombre nécessaire d' opérations élémentaires :
Pour accomplir la tâche, il faut lire toute la suite de chiffres et compter chaque 1 trouvé : au maximum la machine va accomplir n lectures et incrémenter n fois le compteur. Elle va donc achever la tâche après 2n opérations élémentaires.

2) Complexité en temps de cet algorithme
On considère comme unité de temps le temps d'exécution d'une opération élémentaire.
La complexité en temps est ici de 2n.
C(n) = 2n : fonction linéaire du nombre de chiffres de la suite, c'est à dire du nombre de données de départ.

→ **On parle d'un algorithme de *complexité d'ordre linéaire* .**

P2 : Recherche d'une valeur dans un tableau trié (ordre croissant) de taille n.

Imaginons un tableau formé des numéros de cartes de cantine de tous les élèves demi-pensionnaires d'un lycée. On souhaite vérifier (une borne d'entrée du self) qu'un numéro donné est bien dans le tableau.

Premier algorithme proposé : la borne d'entrée vérifie s'il s'agit du premier nombre du tableau. Si c'est le cas elle affiche vraie, sinon elle recommence avec le nombres suivant du tableau, jusqu'à obtenir le bon nombre (ouverture de la borne), sinon (fermeture). **Quel est le coût de cet algorithme ? (minimum et maximum)**

Second algorithme proposé : Recherche dichotomique de la valeur. Nous proposons que la borne d'entrée utilise l'algorithme suivant : consulter le numéro situé au centre du tableau. Si c'est le bon numéro la borne affiche *vrai (et le lycéen peut entrer)*. Sinon si le numéro trouvé est plus grand que celui entré alors recommencer avec la moitié gauche du tableau, sinon recommencer avec la moitié droite du tableau. **Quel est le coût de cet algorithme ? (minimum et maximum)**

Réponse envisagée

1) Premier algorithme : Cas favorable : 1 coup ! Cas défavorable : n coups !

2) Second algorithme :

Cas favorable : 1 coup !

Cas défavorable : le nombre maximum de fois (k) où le tableau peut être partagé en 2, soit k tel que $n = 2^k$ c'est à dite $k = \log_2 (n)$ coups.

→ **On parle d'un algorithme *de complexité logarithmique*.**

P3 : Trier dans l'ordre croissant un tableau de taille n.

Algorithme proposé : tri de base 2 par 2.

Réponse envisagée

On trie les éléments deux à deux ce qui nécessite de comparer chacun des N éléments avec chacun des N-1 autres éléments, donc le nombre de comparaisons sera de l'ordre de N^2 (polynôme de plus haut degré).

→ Un tel algorithme est dit de **complexité quadratique.**

De façon générale un algorithme de complexité de la forme Cn^k (où k >1 et C une constante) est dit de complexité polynomiale.

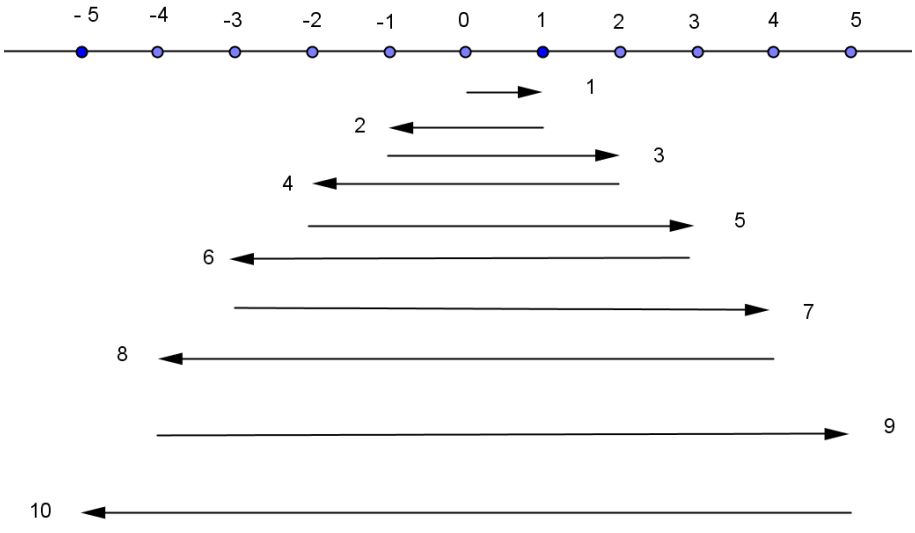
P4 : Trouver un ami habitant la rue Z

Vous êtes au numéro zéro de la rue Z, une rue infinie où les numéros des immeubles sont des entiers relatifs. Dans une direction, vous avez les immeubles numérotés 1, 2, 3, 4, . . . et dans l'autre direction les immeubles numérotés −1, −2, −3, −4, ...
Vous vous rendez chez un ami qui habite rue Z sans savoir à quel numéro il habite. Son nom étant sur sa porte, il vous suffit de passer devant son immeuble pour le trouver (on suppose qu'il n'y a des immeubles que d'un coté et, par exemple, la mer de l'autre). On notera n la valeur absolue du numéro de l'immeuble que vous cherchez (bien entendu n est inconnu).
Trouver un algorithme pour votre déplacement dans la rue Z qui permette de trouver votre ami à coup sûr. Estimer sa complexité.

Réponse envisagée

- Complexité minimale :
  Au minimum, il faut bien se rendre chez notre ami, donc parcourir une distance de n immeubles, ce qui donne bien une complexité minimale de n (donc de type linéaire).
  Ne connaissant pas le numéro convenable, il faut imaginer des méthodes pour y parvenir, et si possible avec le moins d'effort possible.
- Étude des algorithmes proposés.
- Un algorithme possible : Pour le premier, on commence par aller vers le n°1, puis vers le n° – 1, puis on reprend à 2, puis à – 2, etc …., k, - k, etc.

Un schéma heuristique possible :



Les distances parcourues à chaque étape (on rappelle qu'on part de 0) sont 1, 2, 3, 4, 5, 6, . . . , 2k − 1, 2k, etc.

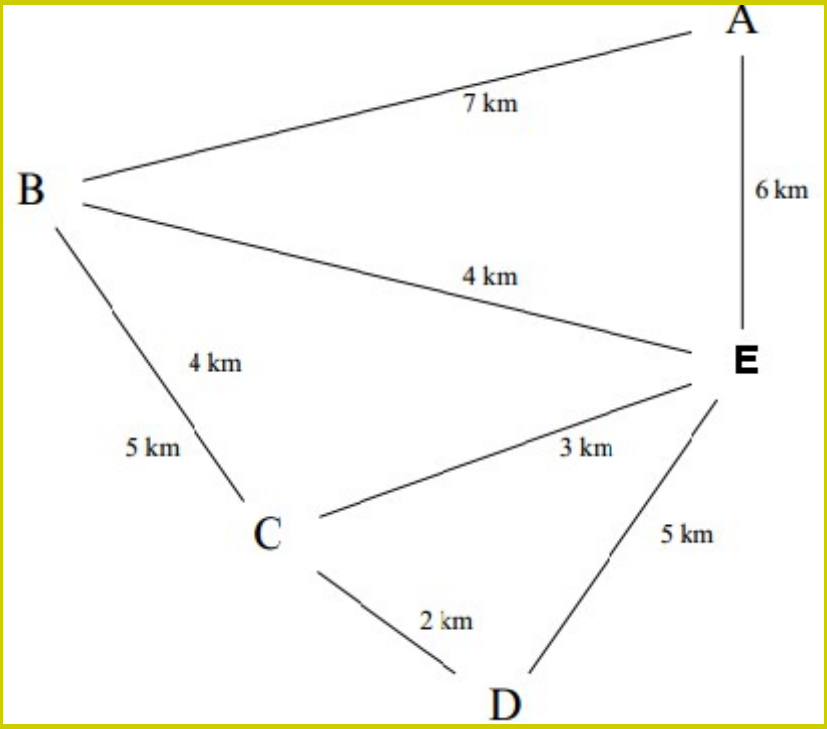
Ainsi, si notre ami habite au numéro n, respectivement au numéro −n, on va parcourir une distance de :

$$\sum_{i=1}^{2n-1} i = \frac{(2n-1)(2n)}{2} \quad \text{respectivement} \quad \sum_{i=1}^{2n} i = \frac{(2n+1)(2n)}{2}$$

La distance parcourue s'exprime donc par $2n^2 - n$ dans le cas où n est positif, par $2n^2 + n$ sinon.
En nombre d'immeubles. Cette distance est quadratique en n.

P5 : Un voyageur de commerce doit se rendre dans un nombre donné de villes et il souhaite le faire en parcourant un trajet minimal.

1)Quel est le trajet optimal qui permette au voyageur de se rendre, en partant de A, dans toutes les autres villes (un seul passage par ville)?



2) Que passe-t-il s'il doit se rendre dans 10 villes ? Dans N villes ?

1) On peut de calculer tous les trajets possibles et choisir le plus petit. ...et il y en quelques-uns ! (Max : 4 x 3 x 2 x 1 = 24)

2) Avec 10 villes, il y en a au maximum 10 ! (10 factorielle) c'est à dire 3 628 800 . Et avec 25, plus de 10^{26} !
C'est énorme !
A ce jour, personne ne connaît de méthode pratique pour ce problème. Le nombre d'opérations augmente très vite avec le nombre de villes. Le nombre d'itinéraires augmente de manière exponentielle.

→ Il s'agit d'un problème à complexité exponentielle.

En conclusion

- On s'intéresse à l'évolution du temps nécessaire à la réalisation d'un algorithme avec l'augmentation du nombre de données à traiter.
- Dès que la taille du problème devient suffisamment grande, seul l'ordre de grandeur de la fonction complexité est important. On parle alors d'étude asymptotique de la fonction de complexité.
- Les fonctions de complexité algorithmique communément rencontrée :

Constante ; « En n »: linéaire ; « En k^n »: (k réel>1) exponentielle

« En nlogn » : quasi-linéaire ; « En log(n) »: logarithmique

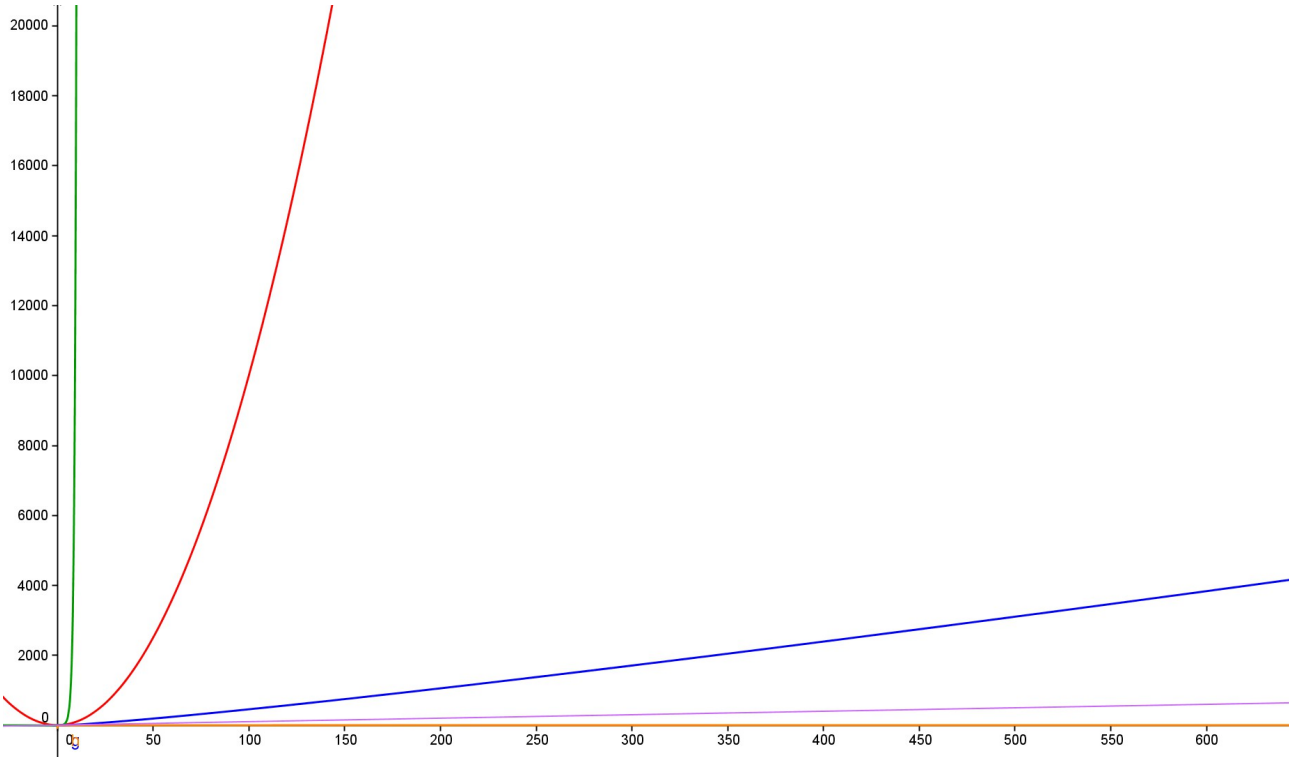
« En n^k » : (k réel>1) polynomiale : quadratique(en n^2), cubique (en n^3).

Une hiérarchie des fonctions de complexité

Exercice : Comment comparer les types de fonctions que nos venons de décrire : lesquelles traduisent une plus grande complexité du problème .

Pour répondre à cette question :
– on peut construire un tableau de valeurs comparées de ces fonctions.
– on peut tracer une représentation graphique des différentes fonctions de complexité : Calculatrice graphique.

n	2	10	20	50	100	200	500	1000
n^2	4,00E+000	1,00E+002	4,00E+002	2,50E+003	1,00E+004	4,00E+004	2,50E+005	1,00E+006
n^3	8,00E+000	1,00E+003	8,00E+003	1,25E+005	1,00E+006	8,00E+006	1,25E+008	1,00E+009
n^{10}	1,02E+003	1,00E+010	1,02E+013	9,77E+016	1,00E+020	1,02E+023	9,77E+026	1,00E+030
log(n)	1,00E+000	3,32E+000	4,32E+000	5,64E+000	6,64E+000	7,64E+000	8,97E+000	9,97E+000
n log n	2,00E+000	3,32E+001	8,64E+001	2,82E+002	6,64E+002	1,53E+003	4,48E+003	9,97E+003
2^n	4,00E+000	1,02E+003	1,05E+006	1,13E+015	1,27E+030	1,61E+060	3,27E+150	1,07E+301



Légende : Complexité exponentielle (en e^n) Complexité quadratique (en n^2)
Complexité en n log(n) Complexité en n Complexité en log(n)

La classe P

Problème de décision

- Un problème de décision est un **problème dont la réponse est soit OUI soit NON.**

- Exemple : Le problème du voyageur de commerce
  Tel que nous l'avons formulé (trouver le trajet optimal) n'est pas un problème de décision. Il le devient si on se demande s'il existe un circuit possible inférieur à une distance déterminée à l'avance.

Classes de problème de décision

Il s'agit simplement de créer une frontière entre les problèmes à croissance très rapide et ceux à croissance extrêmement rapides.On va distinguer les problèmes de décision selon :

– Qui se résolvent à l'aide d'un algorithme de complexité polynomial : constituent **la classe P.**
– Qui se résolvent à l'aide d'un algorithme de complexité exponentielle : constituent la classe E.

La classe NP

- **Ce n'est pas la classe « non P » !**
- On appelle **NP est l'ensemble de tous les problèmes de décision, de réponse OUI, <u>vérifiables</u> en temps polynomial par rapport à la taille de l'entrée** :

Récapitulatif :

P : ensemble de tous les problèmes de décision (j'arrive à répondre par «oui» ou par «non») **pouvant être résolu en temps polynomial. (par rapport à la taille de l'entrée).**

E : ensemble des problèmes de décision solubles en temps exponentiel par rapport à la taille de l'entrée.

NP : ensemble des problèmes pour lesquels on peut donner un algorithme qui vérifie que la réponse est «oui» en temps polynomial en fonction du nombre d'entrées.

Le problème P = NP

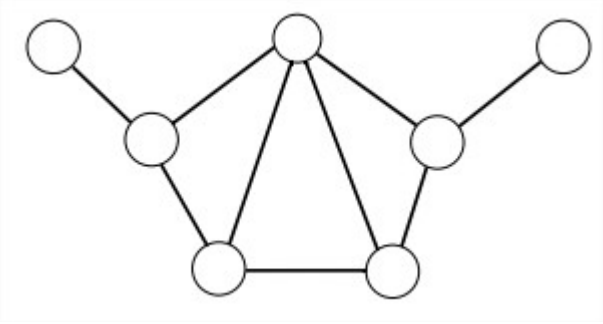
Un problème de décision particulier : colorabilité d'un graphe

Un graphe *G est une paire (V; E) où V est un ensemble de sommets et E un ensemble d'arêtes reliant les sommets. Deux sommets sont voisins si ils sont reliés par une arête.*

K-colorabilité *: Étant donnés un graphe G et un entier k, peut-on colorier les sommets de G avec au plus k couleurs de façon à ce que deux sommets adjacents ne portent jamais la même couleur.*

Un problème de décision : « pour n'importe quel graphe donné, est-ce que ce graphe est colorable avec 2 couleurs ? ».

1) Sur un exemple : Avec ce graphe ?



2) De manière générale ?

<u>Je peux proposer un algorithme</u> : parcourir tous les sommets dans un certain ordre et de colorer au fur et à mesure. C'est à dire que je choisis un sommet que je colorie d'une certaine couleur (en bleu) , les sommets voisins (reliés au sommet choisi) sont immédiatement colorés d'une couleur différente (en rouge). On passe à un sommet voisin dont on considère tour à tour les voisins : Il y a alors 3 possibilités :

- soit le voisin n'a pas de couleur et il prend automatiquement la couleur différente du sommet sur le quel je me trouve. L'algorithme continue en testant un autre voisin.

- soit le voisin a déjà une couleur qui est la même que le sommet où je me trouve : l'algorithme s'arrête et retourne la réponse « Faux ».

- soit le voisin a déjà une couleur qui est différente de celle du sommet où je me trouve : l'algorithme continue en testant un autre voisin.

L'algorithme retourne la valeur « vrai » une fois que tous les sommets ont été testé (et donc colorés) sans que la valeur « faux » apparaisse.

Estimation de la Complexité de mon algorithme :

On ne visite les sommets qu'une fois ; au pire l'algorithme doit vérifier tous les autres sommets pour répondre « vrai » ou « faux ». Considérons le test d'un sommet comme une opération élémentaire. Combien d'opérations élémentaires sont nécessaire dans le pire des cas ?

Soit n le nombre de sommets. Pour chaque sommet au plus il y a n – 1 à tester .

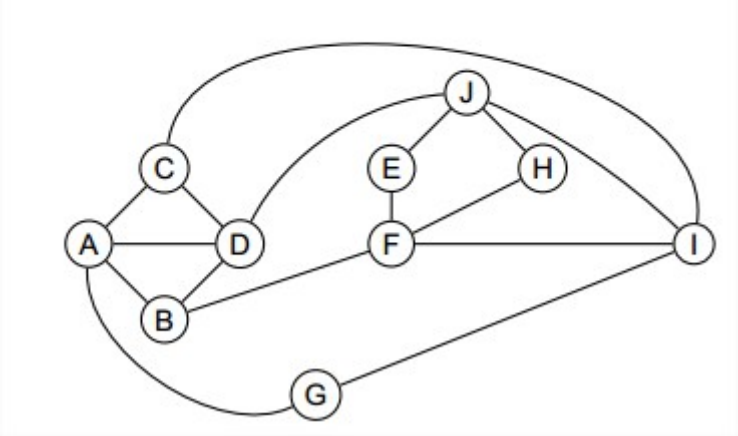
Donc pour l'ensemble des sommets il y a au plus il y a voisins et donc n*(n – 1) tests à faire. Donc n*(n-1) opérations élémentaires. Mon algorithme est donc polynomial (en Kn^2 donc quadratique) en fonction du nombre de sommets du graphe.

Pour ce problème, qui est un problème de décision, je dispose d'un algorithme qui permet de le résoudre en temps polynomial en fonction du nombre d'entrées (le nombre de sommets).

→ Ce problème est donc un problème de la classe P (quadratique en fonction du nombre de sommets du graphe).

Un second problème de décision : « pour n'importe quel graphe donné, est-ce que ce graphe est colorable avec 3 couleurs ? ».

1) Sur un exemple : Avec ce graphe ?



2) De manière générale ?

Algorithme de décision proposé : En testant toutes les possibilités

Pour chaque nouveau sommet il y a 2 couleurs possibles et pour chacune il va falloir tester ce qu'il se passe pour le sommet suivant qui pourra lui aussi prendre 2 couleurs possibles : on voit que le procédé se complique et que le nombre de tests augmente rapidement. En fait , effectuer le test pour chaque sommet cela représente $\mathbf{3^n}$ où n est le nombre de sommets.

→ Ce problème n'est pas actuellement dans P.

Pas d'algorithme de décision connu de complexité polynomiale.

Vérification d'une solution donnée

- Je dispose d'un algorithme simple: il suffit de regarder tous les arcs l'un après l'autre et de vérifier que les deux sommets de l'arc sont de couleurs différentes.

On numérote les arcs.

On va au numéro 1 et on compare les 2 couleurs : identiques = stop ; différentes = Ok , on va sur l'arc suivant. Si le dernier arc donne OK, c'est vérifié comme valide.

Le nombre de tests à faire est donc égal au nombre d'arcs c'est à dire au max à n(n – 1)/2 où n est le nombre de sommets.

→ Donc la **complexité** de cet **algorithme de vérification est de type polynomiale.**

Conclusion

Le problème de 3-colorabilité d'un graphe est donc :

– un problème qui est NP.

– un problème qui à priori n'est pas dans P.

Mais **On n'en est pas certain (qu'il ne soit pas dans P)!** Car ce n'est pas parce qu'on a pas trouvé un algorithme de complexité polynomiale pour ce problème qu'il n'en existe pas un !

En existe-il un ? Est-il possible que ce problème soit dans P ?

Le problème « P versus NP »

La même question se pose pour tous les problèmes dont on n'a pas trouvé d'algorithme de décision efficace en temps polynomial, mais pour lesquels on dispose d'un algorithme de vérification en temps polynomial : est-il ou non dans P ?

Le problème « P est-il égal à NP » posé par les mathématiciens est un problème d'équivalence, donc il se décline en deux questions :

– Un problème qui est décidable en temps polynomial est-il vérifiable en temps polynomial ?
– Un problème qui est vérifiable en temps polynomial est-il décidable en temps polynomial ?

- **Une évidence : les problèmes qui sont dans P sont aussi dans NP :** Il me suffit d'exécuter l'algorithme polynomial qui me répond «oui» ou «non», et de vérifier qu'il répond «oui» !
- **La vraie question : les problèmes qui sont dans NP sont-ils aussi dans P ?**

Problèmes NP-complets: le pavé dans la mare

- Dans les années 60, les chercheurs pensaient que P et NP étaient des classes différentes. Mais sans preuve.
- Stephen Cook (1971) : il existe **un problème NP** doté d'une propriété très étrange : **prouver qu'il est vérifiable en temps polynomial reviendrait à prouver que TOUS les problèmes NP sont solubles en temps polynomial** : un tel problème est un problème NP-complet.
- Depuis, de nombreux problèmes NP-complets ont été découverts. Notamment, les **21 problèmes NP-complets de Karp** (1972).
- Le problème de coloration en 3 couleurs est un problème NP-complet.

Un seul suffirait

- Prouver qu'<u>un</u> problème de NP n'est pas soluble en temps polynomial → P ≠ NP.
- Prouver qu'un problème <u>NP-complet</u> est soluble en temps polynomial → P = NP.

Qu'en pensent-les mathématiciens ?

Une petite enquête

Le mathématicien William Gasarch a mené une enquête en 2002 auprès de 100 mathématiciens et informaticiens théoriciens.

Résolution possible ou non ?	**Pour quelle réponse ?**
• Résolution avant 2050 : 5 % • Résolution après 2050 : 27 % • Ne sera jamais résolue : 5 % • Ne se prononcent pas : 63 %	• « P ≠ NP » : 61 % • « P = NP » : 9 % • Ne se prononcent pas ou question indécidable : 30 %

Quelques avis

<u>Deux avis qui penchent pour P ≠ NP</u> :

Pour John Conway c'est une question de temps et de travail de développement d'outils adaptés.
Juris Hartmanis qui soutient que P ≠ NP pense que la démonstration viendra assez rapidement et précise qu'il ne serait pas surpris qu'elle soit assez courte.
Yuri Gurevich, qui pense que P = NP, mais qu'un un algorithme polynomial pour un problème NP-complet, sera probablement de degré élevé et donc de peu d'intérêt pour la résolution concrète.

<u>Un avis qui penche pour P = NP</u> :

Donald Knuth envisage l'idée que P = NP, mais que la preuve trouvée ne permettra pas d'écrire un algorithme polynomial de résolution.

<u>Un pari risqué et un avis prudent</u> :

Michael Sipser avait parié avec Len Adleman une once d'or que le problème serait résolu avant la fin du XXe siècle : pari perdu.
Avi Wigderson de l'*Institute for Advanced Study* déclare : « La seule chose que je puisse dire de manière certaine est qu'il s'agit d'une des plus importantes et intéressantes questions que l'humanité s'est jamais posée, et que plus de gens et plus de moyens devraient y être consacrés. »

Une piste ?

Une piste considérée comme sérieuse est la proposition récente (6 août 2010) d'une démonstration de **P ≠ NP** effectuée par Vinay Deolalikar, qui s'est avérée contenir des erreurs qui font l'objet d'une attention relativement importante de nombreux mathématiciens et informaticiens de renom.

Un enjeu économique majeur

Prouver que « P = NP » aurait des conséquences importantes : cela signifierait qu'il existe un procédé de résolution réalisable en un temps « acceptable » pour des problèmes considérés comme actuellement très difficiles à résoudre.

Un enjeu essentiel est **la sécurisation des communications :**

– Le cryptage des messages électroniques (communications militaires, bancaires notamment) dépend du fait que le décryptage du code ne soit pas un problème P.
– Si on démontrait que s'en était un, la sécurité du système serait compromise.
– Reste à trouver l'algorithme (prouver qu'il existe et en écrire un n'est pas la même chose)
– Reste à ce que l'algorithme trouvé soit efficace.

Voici abordé un des 7 défis du millénaire :j'invite chacun à s'intéresser aux 6 autres ...

Bibliographie

– Des cours en ligne et notamment un article de :

- Jean-Paul Delahaye (mathématicien lillois) : Un article(92 pages) : Calculabilité et décidabilité, Université des Sciences et Technologies de Lille. Laboratoire d'Informatique Fondamentale de Lille

– Des ouvrages de Mathématiques

- Pierre Wolper : *Introduction à la calculabilité*, 3eme édition, 2006, éditions Dunod
- Ridoux olivier et Lesventes Gilles : *Calculateurs, Calculs, Calculabilité*, 2008,édition Dunod
- Rey Jean- François : *Calculabilité, complexité et approximation*, édition Vuibert

– Des sites internet :

Wikipédia :

Article P = NP http://fr.wikipedia.org/wiki/Probl%C3%A8me_P_%3D_NP (Page française)

Article P = NP http://en.wikipedia.org/wiki/P_versus_NP_problem

Article Théorie de la complexité (pages française et anglaise):

http://fr.wikipedia.org/wiki/T %C3%A9orie_de_la_complexi%C3%A9_des_algorithmes :

http://en.wikipedia.org/wiki/Computational_complexity_theory

The Clay Institute :

http://www.claymath.org/millennium/ ; http://www.claymath.org/millennium/P_vs_NP/

http://www.claymath.org/millennium/P_vs_NP/pvsnp.pdf (Présentation officielle par Stephen Cook)

http://www.lefigaro.fr/sciences-technologies/2010/08/11/01030-20100811ARTFIG00533-un-probleme-mathematique-a-un-million-de-dollars-resolu.php

Autres :

Le blog de balise : http://www.pasithee.fr/blog/?p=783

Un article dans la revue Intestice de Jean-Paul Delahaye (03/04/2007):

http://interstices.info/jcms/c_21832/p-np-un-probleme-a-un-million-de-dollars

– Un article du quotidien Le Figaro (11/08/2010) : http://www.lefigaro.fr/sciences-technologies/2010/08/11/01030-20100811ARTFIG00533-un-probleme-mathematique-a-un-million-de-dollars-resolu.php

– Un ouvrage de vulgarisation:

Keith Devlin, *Les Enigmes mathématiques du 3e millénaire* (Poche, Le Pommier 2007)