CM5 - Complexité des problèmes

Cours RO202

Zacharie ALES
(zacharie.ales@ensta.fr)

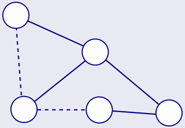
Adapté de cours de Marie-Christine Costa, Alain Faye et Sourour Elloumi

Créé le 21/01/2018
Modifié le 7/12/2021 (v5)

Sujets couverts

Optimisation dans les graphes

Chapitre 1

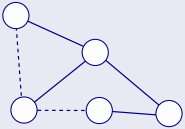


1.1 - Arbre couvrant

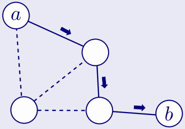
Sujets couverts

Optimisation dans les graphes

Chapitre 1



1.1 - Arbre couvrant

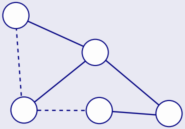


1.2 - Chemin

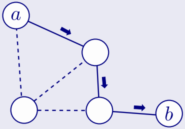
Sujets couverts

Optimisation dans les graphes

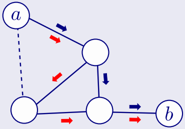
Chapitre 1



1.1 - Arbre couvrant



1.2 - Chemin

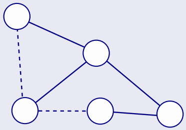


1.3 - Flot

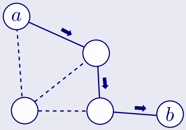
Sujets couverts

Optimisation dans les graphes

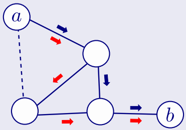
Chapitre 1



1.1 - Arbre couvrant



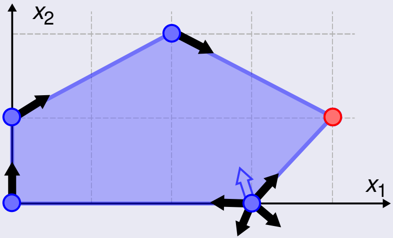
1.2 - Chemin



1.3 - Flot

Programmation linéaire (PL)

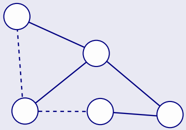
Chapitre 2



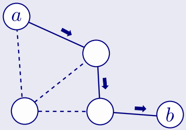
Sujets couverts

Optimisation dans les graphes

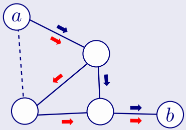
Chapitre 1



1.1 - Arbre couvrant



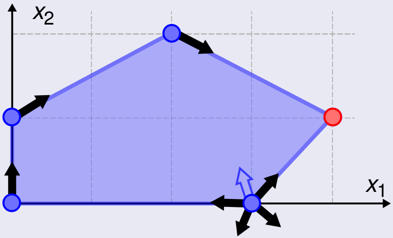
1.2 - Chemin



1.3 - Flot

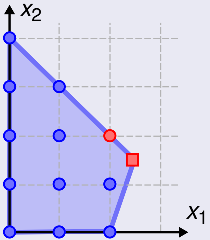
Programmation linéaire (PL)

Chapitre 2



PL en nombres entiers

Chapitre 3



Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée

Supposons 1 étape = $1\mu s$

n	**Nombre d'étapes élémentaires**				
	log(n)	**n**	**n^2**	**2^n**	**n!**
10	10^{-6}				
25	10^{-6}				
50	2.10^{-6}				

En secondes

Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée

Supposons 1 étape = $1\mu s$

n	Nombre d'étapes élémentaires				
	$\log(n)$	n	n^2	2^n	$n!$
10	10^{-6}	10^{-5}			
25	10^{-6}	2.10^{-5}			
50	2.10^{-6}	5.10^{-5}			

En secondes

Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée

Supposons 1 étape = $1\mu s$

n	Nombre d'étapes élémentaires log(n)	n	n^2	2^n	n!
10	10^{-6}	10^{-5}	10^{-4}		
25	10^{-6}	2.10^{-5}	6.10^{-4}		
50	2.10^{-6}	5.10^{-5}	3.10^{-3}		

En secondes

Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée ↓ Supposons 1 étape = $1\mu s$

n	Nombre d'étapes élémentaires				
	$\log(n)$	n	n^2	2^n	$n!$
10	10^{-6}	10^{-5}	10^{-4}	10^{-3}	
25	10^{-6}	2.10^{-5}	6.10^{-4}	30	
50	2.10^{-6}	5.10^{-5}	3.10^{-3}	10^{9}	

En secondes

Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée

Supposons 1 étape = $1 \mu s$

n	Nombre d'étapes élémentaires				
	$\log(n)$	n	n^2	2^n	$n!$
10	10^{-6}	10^{-5}	10^{-4}	10^{-3}	4
25	10^{-6}	2.10^{-5}	6.10^{-4}	30	2.10^{19}
50	2.10^{-6}	5.10^{-5}	3.10^{-3}	10^{9}	3.10^{58}

En secondes

Un tout petit peu de combinatoire

Durée d'exécution d'un algorithme

Taille des données d'entrée

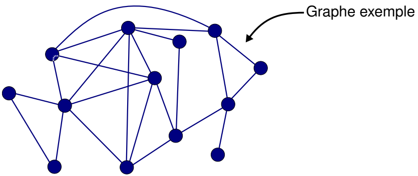
Supposons 1 étape = 1μs

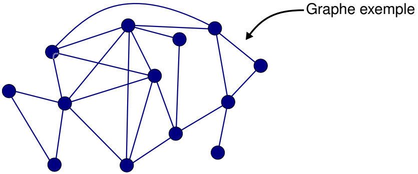
n	Nombre d'étapes élémentaires				
	$\log(n)$	n	n^2	2^n	$n!$
10	10^{-6}	10^{-5}	10^{-4}	10^{-3}	4
25	10^{-6}	2.10^{-5}	6.10^{-4}	30	2.10^{19}
50	2.10^{-6}	5.10^{-5}	3.10^{-3}	10^{9}	3.10^{58}
	.... ("efficace")			.... ("non efficace")	

En secondes

Complexité

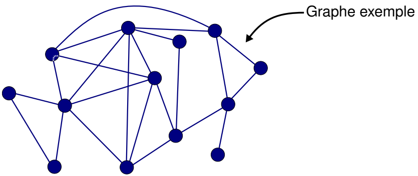
- Problème "facile"
  Peut se résoudre de façon exacte par un algorithme polynomial
- Problème "difficile"
  Seuls algorithmes connus pour les résoudre de façon exacte sont "exponentiels"


Graphe exemple



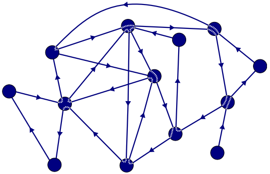
Définition : Chaîne eulérienne

Chaîne passant exactement . . . .
. . . .

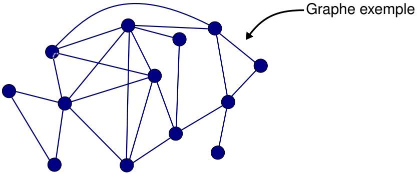


Définition : Chaîne eulérienne

Chaîne passant exactement . . . .
. . . .



Exemple de chaîne eulérienne

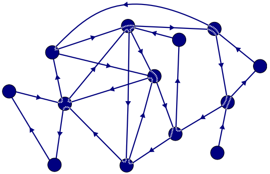


Définition : Chaîne eulérienne

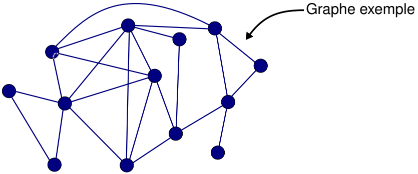
Chaîne passant exactement . . . .
. . . .

Définition : Chaîne hamiltonienne

Chaîne passant exactement . . . .
. . . .



Exemple de chaîne eulérienne

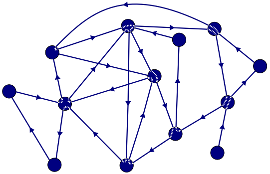


Définition : Chaîne eulérienne

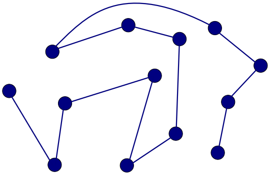
Chaîne passant exactement . . . .
. . . .

Définition : Chaîne hamiltonienne

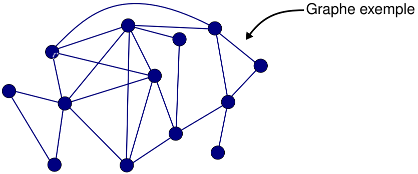
Chaîne passant exactement . . . .
. . . .



Exemple de chaîne eulérienne



Exemple de chaîne hamiltonienne

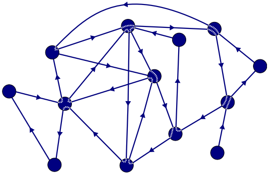


Définition : Chaîne eulérienne

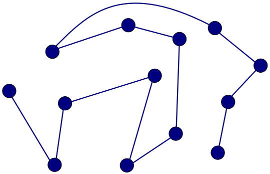
Chaîne passant exactement . . . .
. . . .

Définition : Chaîne hamiltonienne

Chaîne passant exactement . . . .
. . . .



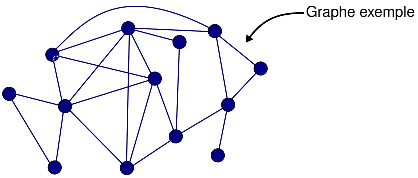
Exemple de chaîne eulérienne



Exemple de chaîne hamiltonienne

Trouver une chaîne eulérienne

Problème **"facile"**

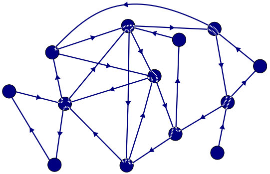


Définition : Chaîne eulérienne

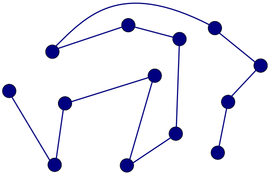
Chaîne passant exactement . . . .
. . . .

Définition : Chaîne hamiltonienne

Chaîne passant exactement . . . .
. . . .



Exemple de chaîne eulérienne



Exemple de chaîne hamiltonienne

Trouver une chaîne eulérienne

Problème **"facile"**

Trouver une chaîne hamiltonienne

Problème **"difficile"**

Comment reconnaître la difficulté d'un problème ?

Théorie de la complexité

Attention : ce cours n'en donne qu'une idée **intuitive**

Définition - **Problème de décision**

. . . .

Définition intuitive - Problème de décision P de **classe NP**

Si vous savez que P a pour réponse oui, il est facile d'en convaincre quelqu'un d'autre
Mais trouver que la réponse est oui peut rester difficile

Exemple

Si on connaît un cycle hamiltonien, il est facile de convaincre quelqu'un qu'il en existe un
Mais trouver un cycle hamiltonien peut être difficile

Définition intuitive - Classe P ou problèmes "faciles" (polynomiaux)

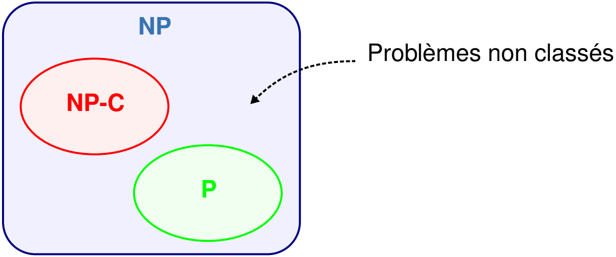
Problèmes de NP qu'on peut résoudre par un algorithme polynomial en fonction de la taille de l'instance

Définition - Problème "difficile"

Problèmes pour lesquels, les seules méthodes de résolution connues exigent un temps de calcul exponentiel en fonction de la taille de l'instance

Définition - Classe NP-Complet

Un problème de NP est NP-complet si "savoir le résoudre efficacement" implique "savoir résoudre efficacement TOUS les problèmes de NP"

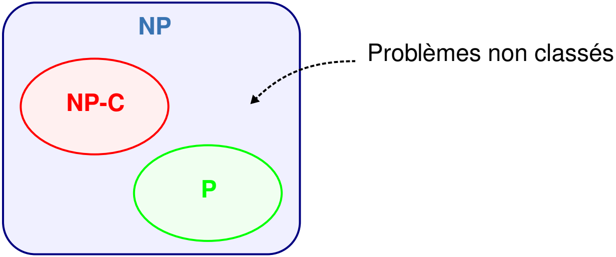


Comment montrer qu'un problème est polynomial ?

1. Trouver un algorithme pour le résoudre
2. Prouver que cet algorithme s'exécute en un temps qui augmente de façon polynomiale en fonction de la taille de l'instance traitée

Comment montrer qu'un problème Π est NP-complet

1. Choisir un problème Π_c déjà connu pour être NP-complet
2. Montrer que Π_c peut se "transformer" en Π
   - Donc, si on savait résoudre (facilement) Π, on saurait résoudre Π_c
   - Or, on ne sait pas résoudre Π_c, il sera donc au moins aussi difficile de résoudre Π

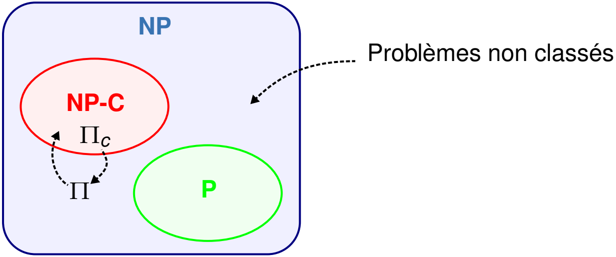


Comment montrer qu'un problème est polynomial ?

1. Trouver un algorithme pour le résoudre
2. Prouver que cet algorithme s'exécute en un temps qui augmente de façon polynomiale en fonction de la taille de l'instance traitée

Comment montrer qu'un problème Π est NP-complet

1. Choisir un problème Π_c déjà connu pour être NP-complet
2. Montrer que Π_c peut se "transformer" en Π
    - Donc, si on savait résoudre (facilement) Π, on saurait résoudre Π_c
    - Or, on ne sait pas résoudre Π_c, il sera donc au moins aussi difficile de résoudre Π



Remarques

- Les problèmes NP-complets sont classés de façon incrémentale
  La classe d'un nouveau problème est déduite de celle d'un ancien problème
- Il existe donc un "premier" problème NP-complet

Le problème SAT ("satisfiabilité" d'une expression)

Problème de décision SAT

Fonction booléenne

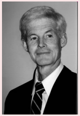
Existe-t-il une affectation des variables telle que f soit vraie ?

Exemple

$$f(x) = (x_1 + \overline{x}_2 + x_3)(\overline{x}_1 + \overline{x}_2 + x_4) + (x_2 + \overline{x}_3 + x_4)(x_1 + x_3 + \overline{x}_4)$$

- une solution : $(x_1, x_2, x_3, x_4) = ($......................$)$

- Stephen Cook a classé le problème SAT comme NP-complet
- SAT est le premier problème NP-complet connu



Réduction polynomiale

Comment transformer un problème NP-complet P_{NP} en un problème $P_?$ de complexité inconnue ?

Définition - **Réduction polynomiale**

P_{NP} se **réduit polynomialement** en $P_?$ s'il existe un algorithme polynomial transformant n'importe quelle instance I_{NP} de P_{NP} en une instance $I_?$ de $P_?$ et tel que I_{NP} et $I_?$ ont la même solution

$$I_{NP} \rightarrow \boxed{\text{Algorithme polynomial}} \rightarrow I_?$$

Intérêt de réduire polynomialement P_{NP} en $P_?$

I_{NP} peut être résolue :

1. en calculant $I_?$ $\leftarrow$ Obtenu par transformation polynomiale
2. en résolvant $I_?$

Résoudre $I_?$ est donc au moins aussi difficile que résoudre I_{NP}

Exemple de réduction

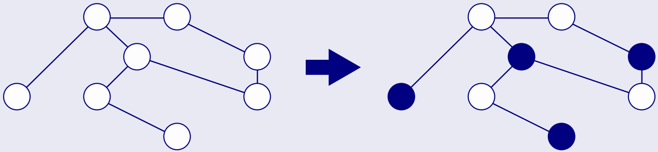
Objectif

Déterminer la complexité du problème stable

Problème stable (notre problème $P_?$)

Soient $G = (V, E)$ et $n \in \mathbb{N}$
$\exists$? un ensemble stable $S \subseteq V$ de taille $\geq n$?

Sommets deux à deux non adjacents



Problème de 3-SAT (notre problème P_{NP})

Problème SAT où la fonction est une forme normale conjonctive de 3 littéraux
i.e., une multiplication de sommes de 3 littéraux

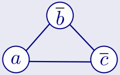
Exemple : $(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$

Exemple de réduction

Réduction d'une instance I_{SAT} du problème 3-SAT vers une instance I_{Stable} du problème stable

- Associer à chaque somme de 3 littéraux, 3 sommets reliés entre eux
  $\rightarrow$ Graphe comportant $3n$ sommets
  $n =$ nombre de sommes de 3 littéraux dans I_{SAT}

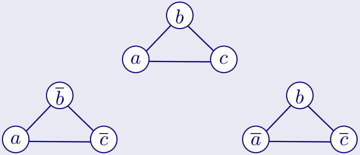
$(a + \overline{b} + \overline{c})$ $\Rightarrow$



- $\forall x$ toutes les occurrences de x sont reliées à toutes les occurrences de $\overline{x}$

$$(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$$

$\Rightarrow$



- 1 littéral vrai dans chaque somme $\Leftrightarrow$ stable de taille n

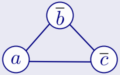
Le problème de stable est donc au moins aussi difficile que celui de 3-SAT !

Exemple de réduction

Réduction d'une instance I_{SAT} du problème 3-SAT vers une instance I_{Stable} du problème stable

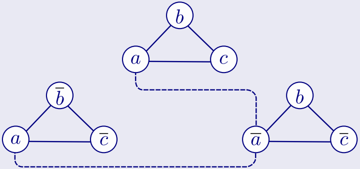
- Associer à chaque somme de 3 littéraux, 3 sommets reliés entre eux
  $\rightarrow$ Graphe comportant $3n$ sommets
  $n =$ nombre de sommes de 3 littéraux dans I_{SAT}

$$(a + \overline{b} + \overline{c})$$



- $\forall x$ toutes les occurrences de x sont reliées à toutes les occurrences de $\overline{x}$

$$(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$$



- 1 littéral vrai dans chaque somme $\Leftrightarrow$ stable de taille n

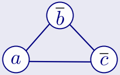
Le problème de stable est donc au moins aussi difficile que celui de 3-SAT !

Exemple de réduction

Réduction d'une instance I_{SAT} du problème 3-SAT vers une instance I_{Stable} du problème stable

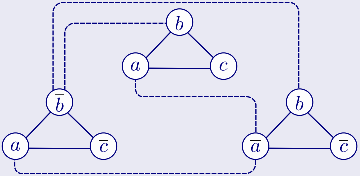
- Associer à chaque somme de 3 littéraux, 3 sommets reliés entre eux
  $\rightarrow$ Graphe comportant $3n$ sommets
  $n =$ nombre de sommes de 3 littéraux dans I_{SAT}

$$(a + \overline{b} + \overline{c})$$



- $\forall x$ toutes les occurrences de x sont reliées à toutes les occurrences de $\overline{x}$

$$(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$$



- 1 littéral vrai dans chaque somme $\Leftrightarrow$ stable de taille n

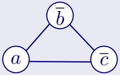
Le problème de stable est donc au moins aussi difficile que celui de 3-SAT !

Exemple de réduction

Réduction d'une instance I_{SAT} du problème 3-SAT vers une instance I_{Stable} du problème stable

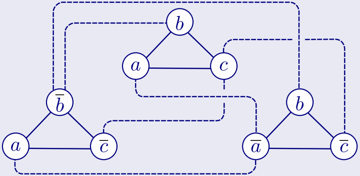
- Associer à chaque somme de 3 littéraux, 3 sommets reliés entre eux
  → Graphe comportant $3n$ sommets
  $n =$ nombre de sommes de 3 littéraux dans I_{SAT}

$(a + \overline{b} + \overline{c})$



- $\forall x$ toutes les occurrences de x sont reliées à toutes les occurrences de $\overline{x}$

$$(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$$



- 1 littéral vrai dans chaque somme ⇔ stable de taille n

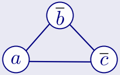
Le problème de stable est donc au moins aussi difficile que celui de 3-SAT !

Exemple de réduction

Réduction d'une instance I_{SAT} du problème 3-SAT vers une instance I_{Stable} du problème stable

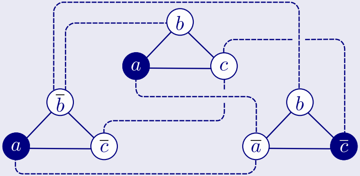
- Associer à chaque somme de 3 littéraux, 3 sommets reliés entre eux
  → Graphe comportant $3n$ sommets
  $n =$ nombre de sommes de 3 littéraux dans I_{SAT}

$$(a + \overline{b} + \overline{c})$$



- $\forall x$ toutes les occurrences de x sont reliées à toutes les occurrences de $\overline{x}$

$$(a + \overline{b} + \overline{c}) \times (a + b + c) \times (\overline{a} + b + \overline{c})$$



- 1 littéral vrai dans chaque somme $\Leftrightarrow$ stable de taille n

Le problème de stable est donc au moins aussi difficile que celui de 3-SAT !

ABOUT PROGRAMS MILLENNIUM PROBLEMS PEOPLE PUBLICATIONS EVENTS EUCLID

Millennium Problems

Yang–Mills and Mass Gap

Experiment and computer simulations suggest the existence of a "mass gap" in the solution to the quantum versions of the Yang-Mills equations. But no proof of this property is known.

Riemann Hypothesis

The prime number theorem determines the average distribution of the primes. The Riemann hypothesis tells us about the deviation from the average. Formulated in Riemann's 1859 paper, it asserts that all the 'non-obvious' zeros of the zeta function are complex numbers with real part 1/2.

P vs NP Problem

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, 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 find a solution.

Navier–Stokes Equation

This is the equation which governs the flow of fluids such as water and air. However, there is no proof for the most basic questions one can ask: do solutions exist, and are they unique? Why ask for a proof? Because a proof gives not only certitude, but also understanding.

Hodge Conjecture

The answer to this conjecture determines how much of the topology of the solution set of a system of algebraic equations can be defined in terms of further algebraic equations. The Hodge conjecture is known in certain special cases, e.g., when the solution set has dimension less than four. But in dimension four it is unknown.

CMI

ABOUT PROGRAMS MILLENNIUM PROBLEMS PEOPLE PUBLICATIONS EVENTS EUCLID

Millennium Problems

Yang–Mills and Mass Gap

Experiment and computer simulations suggest the existence of a "mass gap" in the solution to the quantum versions of the Yang-Mills equations. But no proof of this property is known.

Riemann Hypothesis

The prime number theorem determines the average distribution of the primes. The Riemann hypothesis tells us about the deviation from the average. Formulated in Riemann's 1859 paper, it asserts that all the 'non-obvious' zeros of the zeta function are complex numbers with real part 1/2.

P vs NP Problem

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, 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 find a solution.

Navier–Stokes Equation

This is the equation which governs the flow of fluids such as water and air. However, there is no proof for the most basic questions one can ask: do solutions exist, and are they unique? Why ask for a proof? Because a proof gives not only certitude, but also understanding.

Hodge Conjecture

The answer to this conjecture determines how much of the topology of the solution set of a system of algebraic equations can be defined in terms of further algebraic equations. The Hodge conjecture is known in certain special cases, e.g., when the solution set has dimension less than four. But in dimension four it is unknown.

Qui veut gagner 1 000 000 $?

Il "suffit" de démontrer que

- $P \neq NP$ ou
- $P = NP$

Remarque

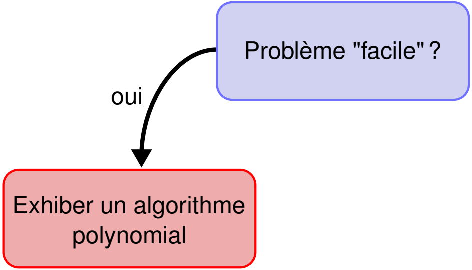
Pour prouver que $P = NP$ il faudrait résoudre un problème NP-complet avec un algorithme polynomial

Cela montrerait que l'ensemble des problèmes NP-complets sont polynomiaux

Approche d’un problème de RO

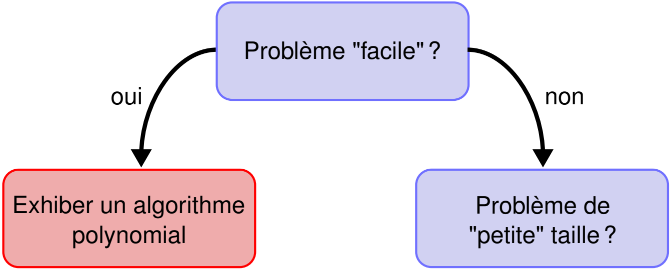
Problème "facile" ?

Approche d'un problème de RO



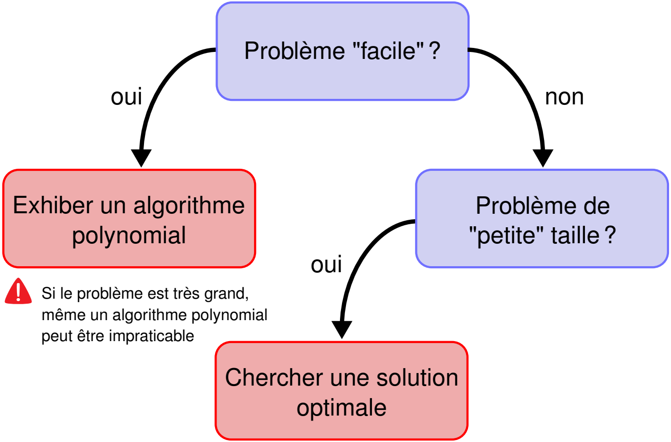
Si le problème est très grand, même un algorithme polynomial peut être impraticable

Approche d’un problème de RO



Si le problème est très grand, même un algorithme polynomial peut être impraticable

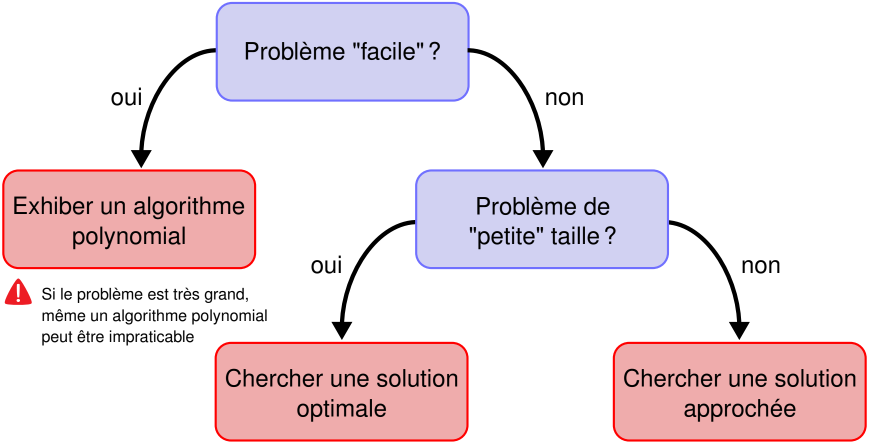
Approche d’un problème de RO



Si le problème est très grand, même un algorithme polynomial peut être impraticable

- *branch-and-bound*
- *branch-and-cut*
- ...

Approche d’un problème de RO



- *branch-and-bound*
- *branch-and-cut*
- ...

Métaheuristiques (entre autres)

Exploration aléatoire astucieuse de l’espace des solutions

- Recuit simulé
- Méthode tabou
- Colonie de fourmis
- Algorithme génétique
- ...