Introduction à la Recherche Opérationnelle

CERMICS,
ENPC

Maël Forcier

22 septembre 2021

$$\begin{array}{ll} \text{Min} & f(x) \\ \text{s.c.} & x \in X. \end{array}$$

f: critère / objectif.
" s.c. " = "sous contraintes"
X : ensemble des solutions admissibles
" $x \in X$ " : contraintes du problème.

Parmi les solutions admissibles, on cherche la solution optimale $x^\star$, i.e., une solution admissible qui minimise la fonction objectif

$$\text{OPT} = \begin{array}{ll} \text{Min} & f(x) \\ \text{s.c.} & x \in X. \end{array}$$

Toujours se demander s'il y a une borne inférieure simple (à calculer) et de bonne qualité.

Cela permet d'évaluer la qualité d'une solution
$\rightarrow$ évite de continuer à chercher des solutions et éventuellement, montre l'optimalité d'une solution.

Part

Un PLNE (MILP en anglais) est un problème d'optimisation de la forme

$$\begin{aligned} \min\ & \boldsymbol{c}^t \boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{Q}^{n-p} \end{aligned}$$

avec $\boldsymbol{c} \in \mathbb{Q}^n$, $\boldsymbol{b} \in \mathbb{Q}^m$, et $\boldsymbol{A} \in \mathbb{Q}^{m \times n}$

Un PLNE (MILP en anglais) est un problème d'optimisation de la forme

$$\begin{aligned} \min\ & \boldsymbol{c}^t \boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{Q}^{n-p} \end{aligned}$$

avec $\boldsymbol{c} \in \mathbb{Q}^n$, $\boldsymbol{b} \in \mathbb{Q}^m$, et $\boldsymbol{A} \in \mathbb{Q}^{m \times n}$

- peut modéliser beaucoup de situations
- algorithmes très efficaces en pratiques (excellents solveurs open sources et commerciaux)

Un PLNE (MILP en anglais) est un problème d'optimisation de la forme

$$\begin{aligned} \min\ & \boldsymbol{c}^t \boldsymbol{x} \\ \text{s.t. } & \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{Z}^p \times \mathbb{Q}^{n-p} \end{aligned}$$

avec $\boldsymbol{c} \in \mathbb{Q}^n$, $\boldsymbol{b} \in \mathbb{Q}^m$, et $\boldsymbol{A} \in \mathbb{Q}^{m \times n}$

- peut modéliser beaucoup de situations
- algorithmes très efficaces en pratiques (excellents solveurs open sources et commerciaux)

Modéliser avec des PLNE est l'une des compétences principales à la fin de ce cours (très utile dans l'industrie)

Faire un plan de table pour un dîner de telle sorte que

- les invités susceptibles de se battre ne soient pas à la même table
- les invités aient le plus d'amis possible à leur table

Modeliser ce problème en utilisant un PLNE

Paramètres du problème

Paramètres du problème

- Les indices $i = 1, \ldots, G$ représentent les invités et $t = 1, \ldots T$ les tables
- $e_{ij} = 1$ si i est susceptible de se battre avec j et 0 sinon,
- $f_{ij} = 1$ si i est ami avec j et 0 sinon.

Variables

Plan de table avec PLNE – solution

Paramètres du problème

- Les indices $i = 1, \ldots, G$ représentent les invités et $t = 1, \ldots T$ les tables
- $e_{ij} = 1$ si i est susceptible de se battre avec j et 0 sinon,
- $f_{ij} = 1$ si i est ami avec j et 0 sinon.

Variables

- $x_{it} = 1$ si i est à la table t et 0 sinon,
- $z_{ijt} = 1$ si i et j sont tous les deux à la table t et 0 sinon.

Paramètres du problème

- Les indices $i = 1, \dots, G$ représentent les invités et $t = 1, \dots T$ les tables
- $e_{ij} = 1$ si i est susceptible de se battre avec j et 0 sinon,
- $f_{ij} = 1$ si i est ami avec j et 0 sinon.

Variables

- $x_{it} = 1$ si i est à la table t et 0 sinon,
- $z_{ijt} = 1$ si i et j sont tous les deux à la table t et 0 sinon.

$$
\begin{aligned}
\max \; & \sum_{i=1}^{n} \sum_{j=i+1}^{n} \sum_{t=1}^{T} f_{ij} z_{ijt} & \\
\text{s.c.} \; & \sum_{t=1}^{T} x_{it} = 1 & \forall i \\
& x_{it} + x_{jt} \leq 2 - e_{ij} & \forall i, j \\
& z_{ijt} \leq x_{it}, \; z_{ijt} \leq x_{jt} & \forall i, j, t \\
& x_{i,t} \in \{0, 1\}, \; z_{i,j,t} \in \{0, 1\} & \forall i, j, t
\end{aligned}
$$

Part

1. Modeliser avec la programmation linéaire en nombre entiers (PLNE)

2. Graphes

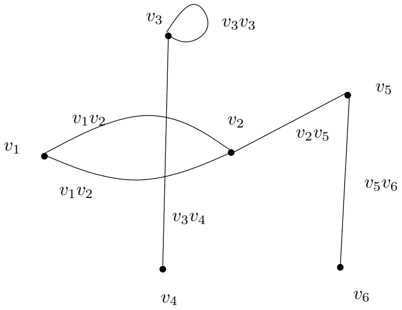
3. Mariages stables

4. Algorithmes et Heuristiques

Graphe : $G = (V, E)$

V : ensemble de sommets (vertices en anglais)

E : ensemble d'arêtes (edges en anglais)= paires de sommets

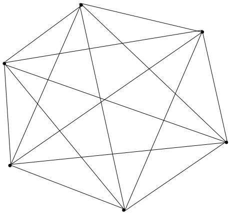


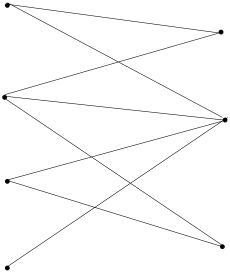
degré $\deg(v)$ d'un sommet v: nombre d'arêtes incidentes.

Graphe simple : au plus une arête entre deux sommets

Graphe complet : graphe simple où chauqe paire de sommets est une arête

Graphe biparti : les sommets sont partitionnés en deux sous-ensembles V^+ et V^-, tels qu'il n'existe pas d'arêtes de V^+ vers V^+ et de V^- vers V^-





K_n: Graphe complet avec n sommets

$K_{m,n}$: Graphe complet biparti avec $|V^+| = m$ et $|V^-| = n$

Combien d'arêtes y a-t-il dans K_n ? et dans $K_{m,n}$?

K_n: Graphe complet avec n sommets

$K_{m,n}$: Graphe complet biparti avec $|V^+| = m$ et $|V^-| = n$

Combien d'arêtes y a-t-il dans K_n ? et dans $K_{m,n}$?

$\frac{n(n-1)}{2}$ dans K_n

Graphes complets, bipartis

K_n: Graphe complet avec n sommets

$K_{m,n}$: Graphe complet biparti avec $|V^+| = m$ et $|V^-| = n$

Combien d'arêtes y a-t-il dans K_n ? et dans $K_{m,n}$?

$\frac{n(n-1)}{2}$ dans K_n
mn dans $K_{m,n}$

Chemin : suite finie de forme

$$v_0, e_1, v_1, \ldots, e_k, v_k$$

$v_i \in V$, $e_j \in E$ avec $e_j = v_{j-1}v_j$.

Chemin simple : passe par chaque arête au plus une fois.

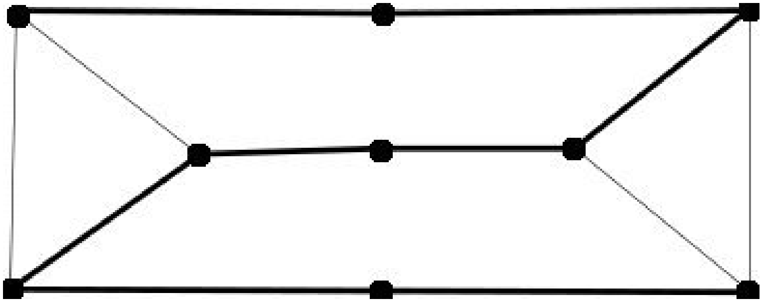
Chemin élementaire : passe par chaque sommet au plus une fois.

Graphe connecté : Pour toute paire de sommets, il existe un chemin entre ces deux sommets.

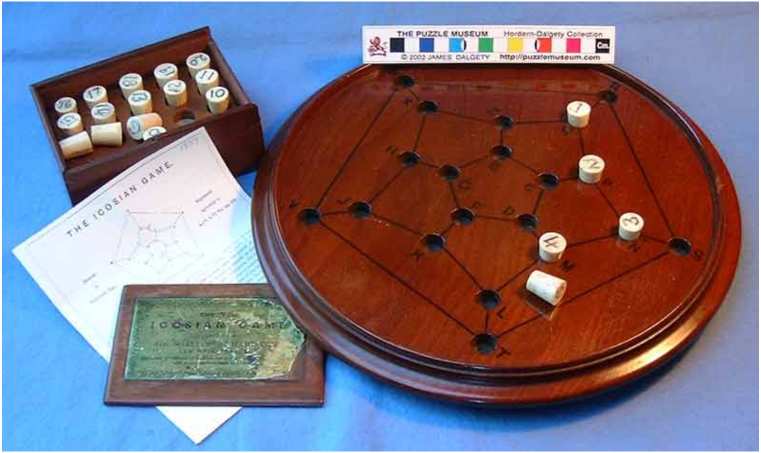
Cycle : chemin tel que $v_0 = v_k$ et les autres sommets apparaissent au plus une fois

Chemin/cycle eulerien : chemin/cycle simple contenant toutes les arêtes

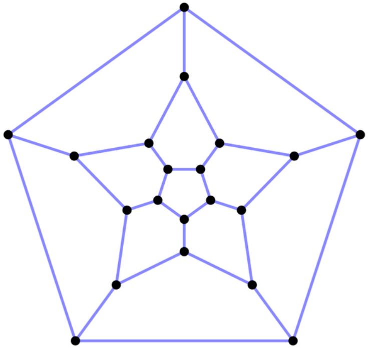
Chemin/cycle hamiltonien : chemin/cycle élementaire contenant tous les sommets



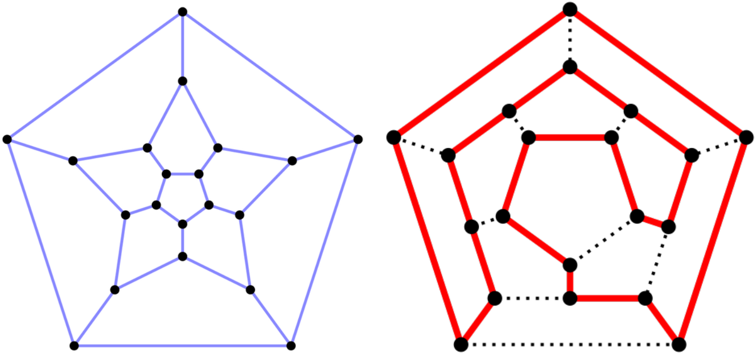
Le jeu d'Hamilton (1805–1865) : "Icosian Game".



Trouver un cycle hamiltonien



Trouver un cycle hamiltonien

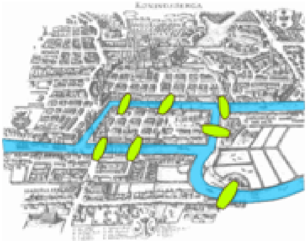


Trouver des exemples de la vie réelle dont les solutions sont des cycles hamiltoniens / euleriens cycles.

Trouver des exemples de la vie réelle dont les solutions sont des cycles hamiltoniens / euleriens cycles.

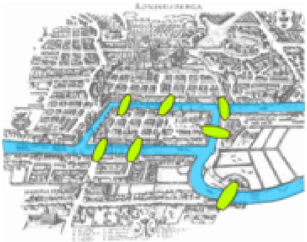
- Le voyageur de commerce
- Le tour du facteur
- Ramassage des ordures

Les ponts de Königsberg (1736) – Euler (1707–1783)

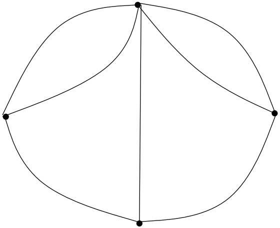


Est-il possible de traverser tous les ponts de Königsberg sans traverser deux fois le même pont ?

Les ponts de Königsberg (1736) – Euler (1707–1783)



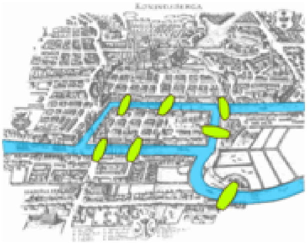
Est-il possible de traverser tous les ponts de Königsberg sans traverser deux fois le même pont ?



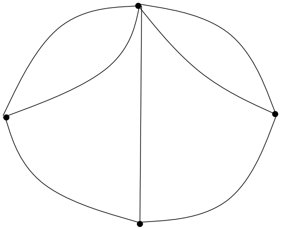
Un graphe est eulerien $\Leftrightarrow$ tous ses sommets sont de degrés pairs.

Quel est le minimum de ponts traversés ?

Les ponts de Königsberg (1736) – Euler (1707–1783)



Est-il possible de traverser tous les ponts de Königsberg sans traverser deux fois le même pont ?



Un graphe est eulerien $\Leftrightarrow$ tous ses sommets sont de degrés pairs.

Quel est le minimum de ponts traversés ?

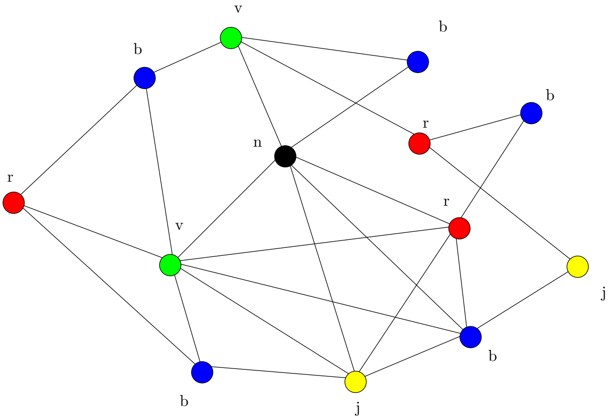
8

Coloration : $c : V \to \mathbb{N}$ ($\mathbb{N}$= couleurs).

Coloration admissible : pour tout couple de voisins u, v, $c(u) \neq c(v)$.

Nombre chromatique $\chi(G)$: nombre minimum de couleurs d'une coloration admissible

On doit donner un ensemble F de formations aux salariés d'une entreprise. Chaque employé i doit suivre un sous-ensemble $F_i \subset F$ de formations. L'entreprise veut trouver le nombre minimal de séances à programmer pour que chaque employé suive ses formations. Modéliser ce problème comme un problème de coloration de graphe.



Graphe coloré avec cinq couleurs: r, b, j, v, n.

Peut-on colorier ce graphe avec moins de cinq couleurs ?

Clique : sous graphe complet

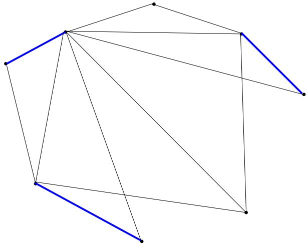
cardinal d'une clique $\leq$ nombre de couleurs d'une coloration admissible.

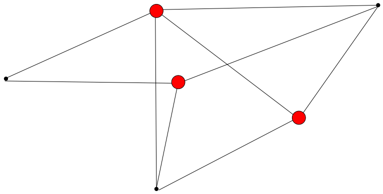
On note $\omega(G)$ le cardinal maximal d'une clique de G.

$$\omega(G) \leq \chi(G).$$

Sous-ensemble d'arêtes $M \subset E$ tel qu'il n'existe pas deux arêtes dans M incidentes au même sommet : couplage

Sous-ensemble de sommets $S \subset V$ tels que toute arête $e \in E$ contient un sommet dans S : couverture de sommets (vertex cover)





: couverture par les sommets du graphe

$\tau(G)$: cardinal minimal d'une couverture (de sommets)
$\nu(G)$: cardinal maximal d'un couplage

Montrer que

$$\nu(G) \leq \tau(G).$$

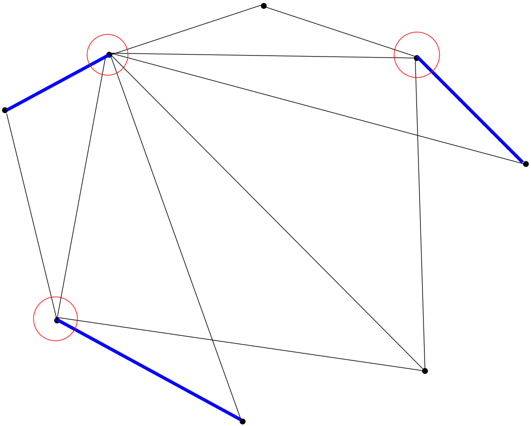
$\tau(G)$: cardinal minimal d'une couverture (de sommets)
$\nu(G)$: cardinal maximal d'un couplage

Montrer que

$$\nu(G) \leq \tau(G).$$

Soit M un couplage C une couverture. Alors

$$|M| \leq |C|.$$



Vous êtes l'agent de sécurité d'une soirée. Vous savez quelle paire d'étudiants vont se battre s'ils boivent trop et vous voulez choisir un nombre minimum d'étudiants à exclure pour éviter les bagarres. Comment les choisir ?

Modéliser ce problème avec les outils présentés précedemment

Fête organisée par le BDE

Vous êtes l'agent de sécurité d'une soirée. Vous savez quelle paire d'étudiants vont se battre s'ils boivent trop et vous voulez choisir un nombre minimum d'étudiants à exclure pour éviter les bagarres. Comment les choisir ?

Modéliser ce problème avec les outils présentés précedemment

Solution : couverture de sommets minimale

Vous êtes l'agent de sécurité d'une soirée. Vous savez quelle paire d'étudiants vont se battre s'ils boivent trop et vous voulez choisir un nombre minimum d'étudiants à exclure pour éviter les bagarres. Comment les choisir ?

Modéliser ce problème avec les outils présentés précedemment

Solution : couverture de sommets minimale

Il y a un grand dîner à la fête, et vous voulez organiser le plan de table de telle sorte que les invités qui ont des chances de se battre ne soient pas à la même table.

Modéliser ce problème avec les outils présentés précedemment

Fête organisée par le BDE

Vous êtes l'agent de sécurité d'une soirée. Vous savez quelle paire d'étudiants vont se battre s'ils boivent trop et vous voulez choisir un nombre minimum d'étudiants à exclure pour éviter les bagarres. Comment les choisir ?

Modéliser ce problème avec les outils présentés précedemment

Solution : couverture de sommets minimale

Il y a un grand dîner à la fête, et vous voulez organiser le plan de table de telle sorte que les invités qui ont des chances de se battre ne soient pas à la même table.

Modéliser ce problème avec les outils présentés précedemment

Solution : coloration de graphe

Part

Les mariages hétérosexuels monogames peuvent être vus comme un problème de couplage dans un graphe biparti .

Est-ce une bonne idée de modéliser ce problème comme un couplage de taille maximale ?

Les mariages hétérosexuels monogames peuvent être vus comme un problème de couplage dans un graphe biparti .

Est-ce une bonne idée de modéliser ce problème comme un couplage de taille maximale ?

On peut aussi rajouter des poids sur les arêtes.

Est-ce une bonne idée de modéliser ce problème comme un couplage de taille maximale ?

Un couplage peut être instable

Un couplage peut être instable

Modéliser la *stabilité* :

- Chaque femme $f \in F$ a un ordre de préférence sur les hommes : $\preceq_f$.
- Chaque homme $g \in G$ a un ordre de préférence sur les femmes : $\preceq_g$.

Un mariage (couplage) $M \subset E$ est *stable* si

$$\{f,g\} \in E \setminus M \quad \Longrightarrow \quad \begin{array}{r} \exists g' \in G, \{f,g'\} \in M \text{ avec } g' \succeq_f g \\ \text{ou } \exists f' \in F, \{f',g\} \in M \text{ avec } f' \succeq_g f. \end{array}$$

École des Ponts ParisTech

Shapley a reçu le prix Nobel d'économie (2012) pour le théorème suivant:

Theorem

Il existe toujours un mariage stable. De plus, un tel mariage peut être trouvé en $O(nm)$.

Application : Admission Post-Bac.

Algorithme

Le premier jour :

- Chaque homme invite la femme qu'il préfère.
- Chaque femme choisit l'homme qu'elle préfère parmi ceux qui l'ont invitée.

Le k-ème jour :

- Chaque homme invite la femme qu'il préfère parmi celles qui ne lui ont jamais refusé une invitation.
- Chaque femme choisit l'homme qu'elle préfère parmi ceux qui l'ont invitée.

Quand aucun homme ne se fait refuser, i.e. dès que les couples de deux jours consécutifs sont identiques, on a des mariages stables.

Preuve

1. *Chaque femme qui a été invité par un homme un jour est certaine qu'elle sera invitée par un homme qu'elle aime au moins autant le lendemain.*
2. *L'algorithme s'arrête.*
3. *L'algorithme fournit un mariage stable.*

Est-ce mieux de proposer ou de choisir ?

Est-ce mieux de proposer ou de choisir ?

- Chaque homme est marié à la femme qu'il préfère parmi celles avec qui il aurait pu être marié dans un mariage stable.
- Ce n'est pas le cas pour les femmes.

Il est plus optimal de proposer.

Est-ce mieux de proposer ou de choisir ?

- Chaque homme est marié à la femme qu'il préfère parmi celles avec qui il aurait pu être marié dans un mariage stable.
- Ce n'est pas le cas pour les femmes.

Il est plus optimal de proposer.

La façon d'implémenter l'algorithme n'est donc pas neutre.
Par exemple, comment implementer APB ? Donner la priorité au universités ou au élèves ?

Part

1. Modeliser avec la programmation linéaire en nombre entiers (PLNE)

2. Graphes

3. Mariages stables

4. Algorithmes et Heuristiques

Algorithme: Une suite d'*opérations élémentaires* pouvant être implémentées sur un ordinateur.

Complexité en temps d'un algorithme : nombre d'opérations élémentaires qui doivent être réalisées si la *taille* de l'entrée est n.

Exemples:

1. Classer n entiers ?
2. Tester l'existence d'un cycle eulérien

Algorithme: Une suite d'*opérations élémentaires* pouvant être implémentées sur un ordinateur.
Complexité en temps d'un algorithme : nombre d'opérations élémentaires qui doivent être réalisées si la *taille* de l'entrée est n.

Exemples:

1. Classer n entiers ?
2. Tester l'existence d'un cycle eulérien

1. $O(n\log(n))$
2. $O(m+n)$

Complexity function

Algorithme: Une suite d'*opérations élémentaires* pouvant être implémentées sur un ordinateur.

Complexité en temps d'un algorithme : nombre d'opérations élémentaires qui doivent être réalisées si la *taille* de l'entrée est n.

Exemples:

1. Classer n entiers ?
2. Tester l'existence d'un cycle eulérien

1. $O(n\log(n))$
2. $O(m+n)$

Les définitions formelles de "problèmes", "algorithmes", "taille de l'entrée" et "complexité en temps" :

- Formalisation assez technique avec machine de Turing (cf poly).
- Compréhension informelle suffisante pour ce cours (cf cours 4).

Algorithme polynomial : complexité en $= O(n^a)$ avec a fixé.

Sinon, algorithme exponentiel.

Complexité	Taille de n			
	10	20	50	60
n	$0,01$ µs	$0,02$ µs	$0,05$ µs	$0,06$ µs
n^2	$0,1$ µs	$0,4$ µs	$2,5$ µs	$3,6$ µs
n^3	1 µs	8 µs	125 µs	216 µs
n^5	$0,1$ ms	$3,2$ ms	$312,5$ ms	$777,6$ ms
2^n	~ 1 µs	~ 1 ms	~ 13 jours	~ 36.5 years

Table: Comparaison des temps de calculs pour différentes complexités et tailles de l'entrée sur un ordinateur effectuant 1 milliard d'opérations élémentaires par seconde.

Soit $\mathcal{A}$ un algorithme qui résoud un problème $\mathcal{P}$ en 2^n operations. Nous avons (aujourd'hui) un ordinateur qui résoud $\mathcal{P}$ avec l'algorithme $\mathcal{A}$ en moins d'une heure pour des instances de taille allant jusqu'à $n = 438$.

Avec un ordinateur 1000 fois plus rapide, nous pouvons résoudre des instances de quelle taille en moins d'une heure ?

Soit $\mathcal{A}$ un algorithme qui résoud un problème $\mathcal{P}$ en 2^n operations. Nous avons (aujourd'hui) un ordinateur qui résoud $\mathcal{P}$ avec l'algorithme $\mathcal{A}$ en moins d'une heure pour des instances de taille allant jusqu'à $n = 438$.

Avec un ordinateur 1000 fois plus rapide, nous pouvons résoudre des instances de quelle taille en moins d'une heure ?
448

Taille de la plus grande instance que l'on peut résoudre en 1 heure

Complexité function	Ordinateur aujourd'hui	Ordinateur 100 $\times$ plus rapide	Ordinateur 1000 $\times$ plus rapide
n	N_1	$100N_1$	$1000N_1$
n^2	N_2	$10N_2$	$31.6N_2$
n^3	N_3	$4.64N_3$	$10N_3$
n^5	N_4	$2.5N_4$	$3.98N_4$
2^n	N_5	$N_5 + 6.64$	$N_5 + 9.97$
3^n	N_6	$N_6 + 4.19$	$N_6 + 6.29$

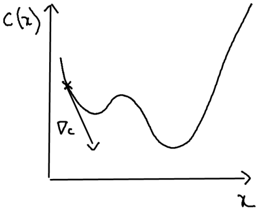
Table: Comparaison de differentes complexités

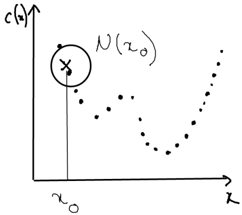
Nous verrons dans le cours 4 que l'on peut montrer que des problèmes sont intrinsèquement difficiles.

Définition: une *heuristique* pour un problème d'optimisation est un algorithme d'optimisation sans garantie de convergence.

Definir une fonction voisinage $\mathcal{N}$ qui à chaque solution x associe un ensemble de voisins $\mathcal{N}(x)$

- Commencer avec une solution admissible x
- S'il existe une $x' \in \mathcal{N}(x)$ qui est meilleure que x, alors $x \leftarrow x'$? Sinon, s'arrêter.





Soit $G = (V, E)$ un graphe et k un entier. Trouver une coloration $c : V \to [k]$ telle que le nombre d'arêtes uv avec $c(u) = c(v)$ est minimal.

- Colorier G arbitrairement avec k couleurs.
- Repéter :
  - Trouver une arête monochrome telle que, si l'on modifie la couleur de l'une des extrémités, le nombre d'arêtes uv avec $c(u) = c(v)$ diminue.
  - Si une telle arête existe, faire cette modification.
  - Sinon, s'arrêter.