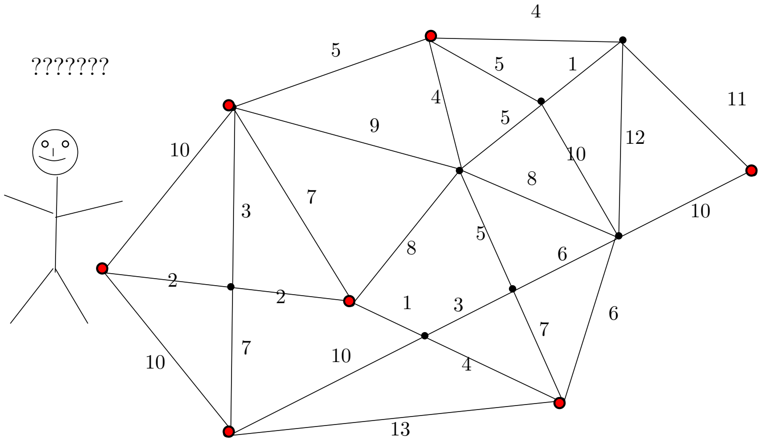
Introduction to Operations Research

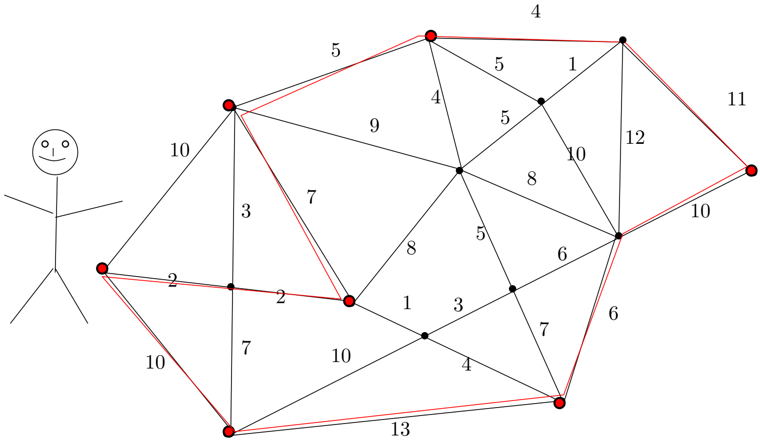
CERMICS,
ENPC

Axel Parmentier

September 26, 2018



Mister Supersales must plan his tour



Mister supersale has planned his tour

Is there an optimal solution?

Is there an optimal solution?

Yes: finite set of solution

Can we enumerate all the solutions?

Is there an optimal solution?

Yes: finite set of solution

Can we enumerate all the solutions?

With 25 cities, we have $24! = 24 \times 23 \times 22 \times \cdots \times 2 \times 1$ possibilities, that is, around 6.204×10^{25} possibilities.

Using paper and pencil, testing 1 possibility per second, requires around 1.976×10^{16} years.

Testing 1 million possibilities per second with a computer, requires 19 billion years.

Part

1. What is Operations Research
2. Syllabus
3. Graphs
4. Complexity

The traveling salesman problem is one of the most famous Operations Research problem.

Operations Research (OR):
mathematical discipline that deals with the optimal allocation of resources (typically in firms).

The traveling salesman problem is one of the most famous Operations Research problem.

Operations Research (OR):
mathematical discipline that deals with the optimal allocation of resources (typically in firms).

Why this name? What was it invented for?

Data mining is descriptive analytics:

- My data contains A and B

Machine Learning is predictive analytics:

- if A happens, then B will happen

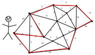
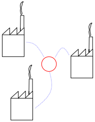
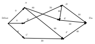
Operations Research is prescriptive analytics:

- if I want B to happen, then I must do A

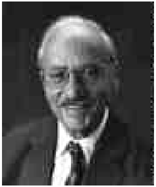
As such, OR is a key tool of artificial intelligence

More prosaically, Big Data have multiplied the fields of applications of OR, and there are too few experts today

Examples of Operations Research problems

- Find the best tour 
- Plan the best timetable 
- Find the most resilient network 
- Fill a container optimally 
- Locate facilities/warehouses optimally 
- Schedule jobs on machines 

Monge, Blackett, Dantzig.



Fast growth since the fifties

- academia (maths, computer science),
- industry (supply chain, transport, telecommunications , etc.)

Two keywords : Modeling et Optimization.

Performances on the traveling salesman problem

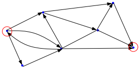
1954	Dantzig, Fulkerson, Johnson	49 cities
1971	Held, Karp	64 cities
1975	Camerini, Fratta, Maffioli	67 cities
1977	Grötschel	120 cities
1980	Crowder, Padberg	318 cities
1987	Padberg, Rinaldi	532 cities
1987	Grötschel, Holland	666 cities
1987	Padberg, Rinaldi	$2'392$ cities
1994	Applegate, Bixby, Chvátal, Cook	$7'397$ cities
1998	Applegate, Bixby, Chvátal, Cook	$13'509$ cities
2001	Applegate, Bixby, Chvátal, Cook	$15'112$ cities
2004	Applegate, Bixby, Chvátal, Cook, Helsgaun	$24'978$ cities

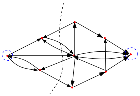
And it is not a question of computer performances. Initial algorithms on today's computers would not deal with 100 cities.

Part

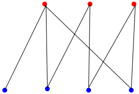
L1, sep. 26 Graphs and complexity

L2, oct. 3 Shortest paths and dynamic programming

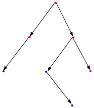




L3, oct. 10 Network flows



L4, oct. 17 Bipartite graphs and Linear Programming



L5, oct. 24 Mixed Integer Linear Programming

L6, nov. 14 Heuristics $e^{-\frac{z}{k_B T}}$

L7, nov. 28 Exercises + Introduction to Implementation (`Julia`)

Groups

3 groups.

L8 dec 5 Project (self-learning)

L9 dec 12 Facility location and bin-packing + Industrial presentation (Frédéric Gardi – Localsolver) 1h

L10 dec 19 Network design + Industrial presentation (Thibault Corneloup – Air France)

L11 jan 9 Routing + Intervention 1h (Mathieu Sanchez)

L12 jan 16 Scheduling (self-learning)

L13 jan 23 Exam

Evaluation

- Multiple choice tests (mandatory) on Educnet
- Project
- Final Exam

$$\text{Note} = \frac{2}{5} \cdot \text{Project} + \frac{3}{5} \cdot \text{Exam}$$

No re-sit exam if more than 1 unjustified absence or more than 1 undone multiple choice test

Frédéric Meunier's monograph

New monograph (progressively updated on Educnet):

- Please indicate me the typos: `axel.parmentier@enpc.fr`
- up to two bonus points for those who identify

Three previous exams

Hackathon

Project

The problem of the miniproject will be the same as the one of the Hackathon

Agenda :

L7, nov. 28 Introduction to Implementation (`Julia`)

nov. 29 Hackathon (voluntary) + subject available on Educnet

L8, dec. 5 “Séance en autonomie” on the project

jan. 16 End of the project

In the textbook

Complexity: Chapter 2

Graphs: Chapter 3

Model =

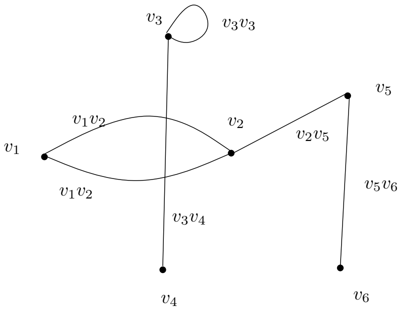
A mathematical transcription of reality that enables to apply mathematical theory and tools, and translate their results into prediction and decisions in the real world.

Two kinds of models

- Epistemology: model to understand a complicated phenomenon
- Praxeology: model to decide

Operations Research: models to find a good/the best decision in a huge set of potential ones

Graph : $G = (V, E)$
V : set of sommets
E : set of edges = unordered pairs of vertices

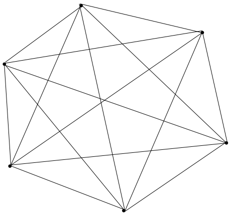


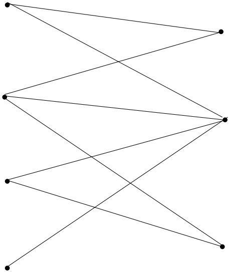
degree $\deg(v)$ of a vertex v: number of incident edges.

Simple graph: at most one edge between two vertices

Complete graph: simple graph where every pair of vertices is an edge

Graphe biparti: vertices partitioned into two subsets such that there is no edge between two vertices of the same subset





K_n: complete graph with n vertices

$K_{m,n}$: bipartite complete graph with m and n vertices

How many edges in K_n? And in $K_{m,n}$?

Path : sequence of the form

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

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

Simple path: crosses at most once an edge.

Elementary path: crosses at most once a vertex

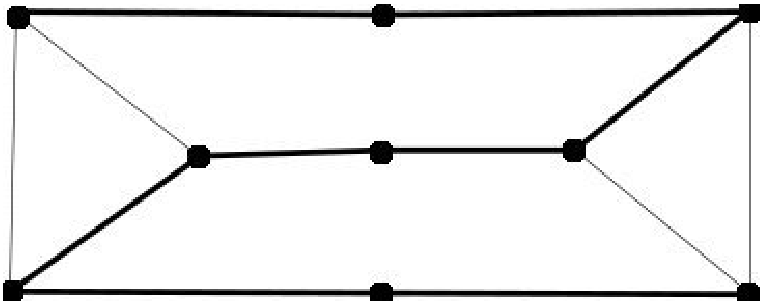
Connected graph: a path between any pair of vertices

Cycle : path such that $v_0 = v_k$ and all the other vertices are contained at most once

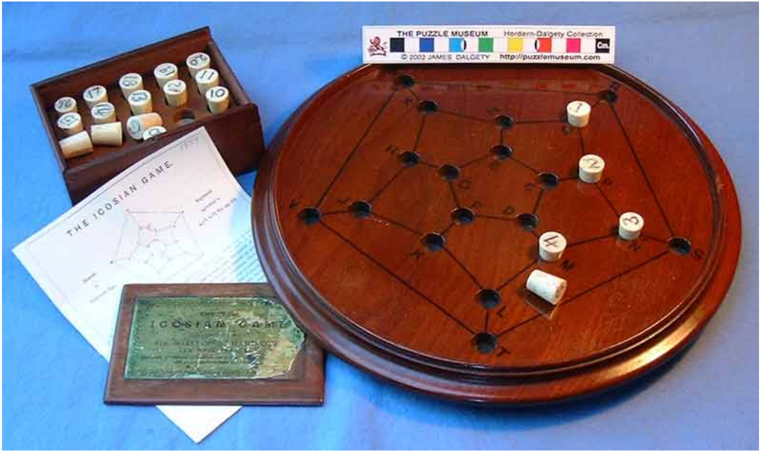
Eulerian path/cycle: simple path/cycle containing all the edges

Hamiltonian path/cycle: elementary path/cycle containing all the vertices

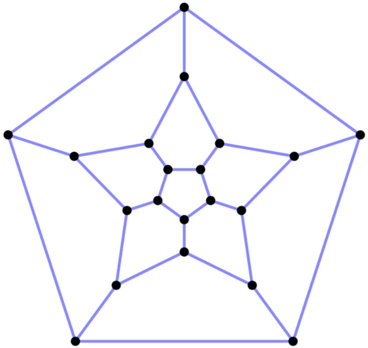
Hamiltonian cycle



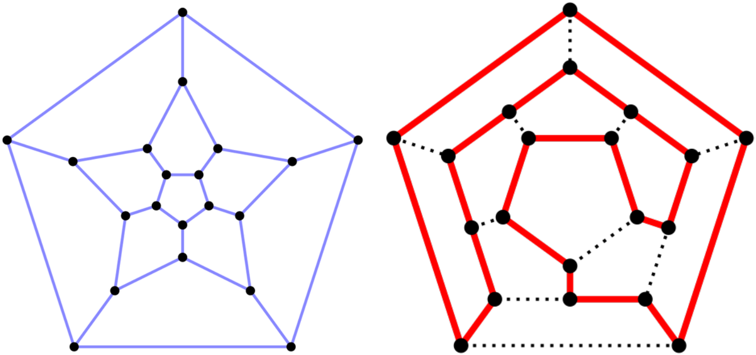
Hamilton (1805–1865) invented the “Icosian Game”, which was not a commercial success.



Find a Hamiltonian cycle



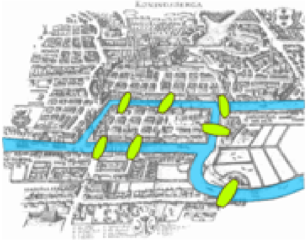
Find a Hamiltonian cycle



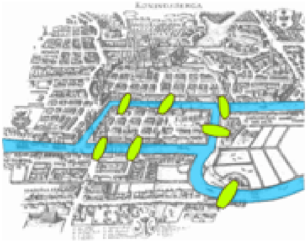
Give examples of real-life problems whose solutions are Hamiltonian / Eulerian cycles.

Give examples of real-life problems whose solutions are Hamiltonian / Eulerian cycles.

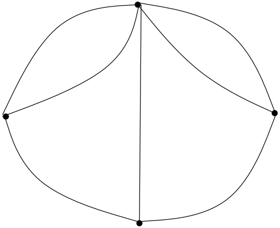
- Traveling salesman
- Post office



Is it possible to go through all the bridge of Königsberg without crossing twice the same bridge?

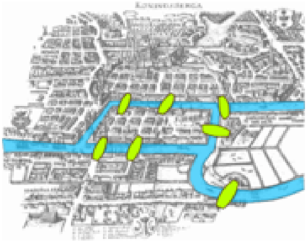


Is it possible to go through all the bridge of Königsberg without crossing twice the same bridge?

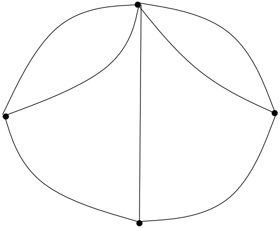


A graph is Eulerian $\Leftrightarrow$ has at most two vertices of odd degree.

What is the minimum number of bridges crossed?



Is it possible to go through all the bridge of Königsberg without crossing twice the same bridge?



A graph is Eulerian $\Leftrightarrow$ has at most two vertices of odd degree.

What is the minimum number of bridges crossed?

8

Coloring : $c : V \to \mathbb{N}$ ($\mathbb{N}$= colors).

Proper coloring : for any neighbor u, v, we have $c(u) \neq c(v)$.

Chromatic number $\chi(G)$: minimum numbers of colors in a proper coloring

A set F of formations must be given to employees of a firm. Each employee i must follow a subset F_i of formations. The firm wants to find the minimum number of formation slots it must schedule so that each employee can attend to its formations. Model this problem as a coloring problem.

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

f: criteria / objective.
" s.t. " = "subject to"
X : set of feasible solutions
" $x \in X$ " : constraints of the Optimization program.

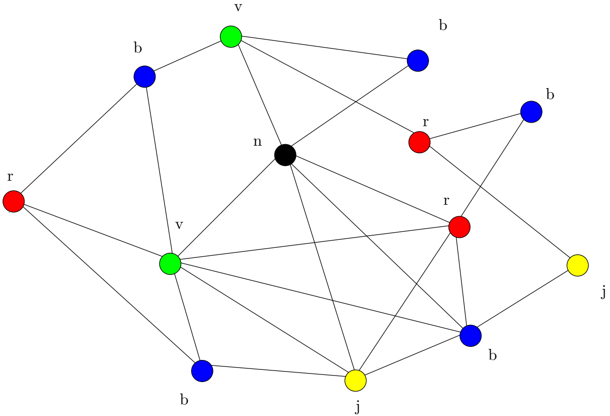
Among the feasible solutions, we seek an *optimal solution* x^*, i.e., a feasible solution that minimizes the criteria.

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

Always ask if there is a simple and good quality lower bound to OPT.

Enables to evaluate the quality of a solution $\rightarrow$ identify when to stop searching, and possibly to the optimality of the solution

Coloration



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

Can we color this graph with fewer than five colors?

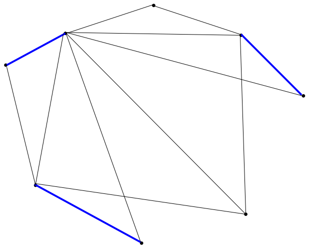
cardinal of a complete subgraph $\leq$ number of colors in a proper coloring.

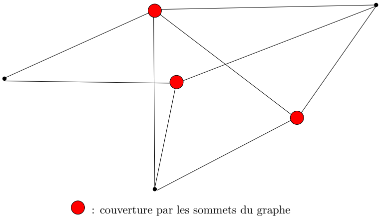
We denote by $\omega(G)$ the maximum cardinality of a complete subgraph of G.

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

Set of edges two by two disjoint: matching

Set of vertices S such that each edge contains a vertex in S: cover





Give example of problems modeled by matching / covers.

$\tau(G)$: minimum cardinality of a cover
$\nu(G)$: maximum cardinality of a matching

Prove that

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

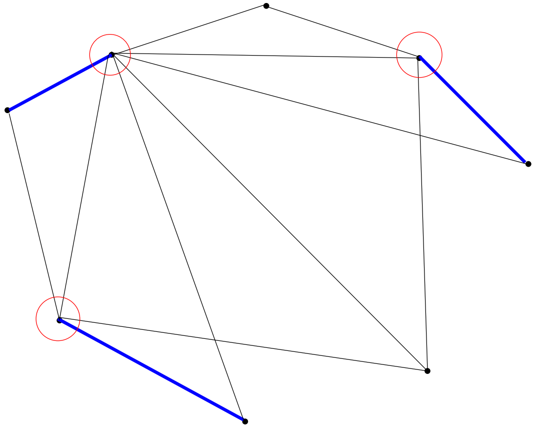
$\tau(G)$: minimum cardinality of a cover
$\nu(G)$: maximum cardinality of a matching

Prove that

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

Let M be a matching C a vertex cover. Then

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



Part

Problem:

- Input
- Question / task

EULERIAN PATH PROBLEM
Input. A graph G
Question. Is there an Eulerian path in G?

MAXIMUM WEIGHT MATCHING
Input. A graph $G = (V, E)$, a weight function $w : E \to \mathbb{Q}_+$.
Output. A maximum weight matching in G.

Decision problem: answer by yes or no to a question.

Optimization problem: find the optimum of a function (under some constraints)

Algorithm: a sequence of *elementary operations* that can be implemented on a computer.

Given a problem $\mathcal{P}$ and an algorithm $\mathcal{A}$ solving it, we can ask how efficient it is.

- Complexity theory

Complexity function

Time complexity $f(n)$ of an algorithm: number of elementary operation that must be realized if the input is of *size* n.

Example:

1. Sorting n integers?
2. Testing if an Eulerian cycles exists?

Time complexity $f(n)$ of an algorithm: number of elementary operation that must be realized if the input is of *size* n.

Example:

1. Sorting n integers?
2. Testing if an Eulerian cycles exists?

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

Complexity function

Time complexity $f(n)$ of an algorithm: number of elementary operation that must be realized if the input is of *size* n.

Example:

1. Sorting n integers?
2. Testing if an Eulerian cycles exists?

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

"Clean" definition of algorithm, size of the input, and time complexity.

- requires to formalize what is an algorithm on a computer
- See textbook for more details
- Informal understanding sufficient for this lecture

Polynomial algorithm: time complexity in $= O(n^a)$ with a fixed.

Otherwise, exponential algorithm.

Time complexity	Size 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: Comparison of different time complexity functions on a computer executing 1 billion operations per second.

Let $\mathcal{A}$ be an algorithm solving a problem $\mathcal{P}$ in 2^n operations. We have a computer that solved $\mathcal{P}$ with $\mathcal{A}$ in 1 hour for instances of size up to $n = 438$.

With a computer 1000 times faster, instances of up to which size wan we solve in 1 hour?

Let $\mathcal{A}$ be an algorithm solving a problem $\mathcal{P}$ in 2^n operations. We have a computer that solved $\mathcal{P}$ with $\mathcal{A}$ in 1 hour for instances of size up to $n = 438$.

With a computer 1000 times faster, instances of up to which size wan we solve in 1 hour?
448

Size of the largest instance that we can solved in 1 hour

Complexity function	Present day computer	Computer 100 $\times$ faster	Computer 1000 $\times$ faster
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: Comparison of different complexity functions

A *decision problem* is polynomial or in $\mathscr{P}$ if there exists a polynomial algorithm that solves it.

A *decision problem* is non-deterministically polynomial or in $\mathscr{NP}$ if, if the answer is yes, there exists a certificate and a polynomial algorithm that enables to check that the solution is yes.

HAMILTONIAN CYCLE
Input. A graph $G = (V, E)$
Question. Does G have a Hamiltonian cycle

Show that the HAMILTONIAN CYCLE problem is in $\mathscr{NP}$ (Give a certificate)

For $F \subseteq E$, we can test in polynomial if F is a Hamiltonian cycle. F is a certificate.
The Hamiltonian cycle problem is in $\mathscr{NP}$.

Class $\mathscr{NP}$

Proposition

$$\mathscr{P} \subseteq \mathscr{NP}.$$

Proof.

If the answer is yes, the input is a certificate. □

$\mathscr{NP}$-complete problem

A problem $\mathcal{P}$ is $\mathscr{NP}$-complete if

- $\mathcal{P}$ is in $\mathscr{NP}$
- $\mathcal{P}$ is at least as difficult as any problem in $\mathscr{NP}$.

If there exists a polynomial algorithm solving an $\mathscr{NP}$-complete problem, then there is a polynomial algorithm solving any problem in $\mathscr{NP}$.

Theorem (Cook, 1970)

There exists $\mathscr{NP}$-complete problems.

What does "at least as difficult" mean

Example of $\mathscr{NP}$-complete problem

HAMILTONIAN CYCLE
Input. A graph $G = (V, E)$
Question. Does G have a Hamiltonian cycle

A *polynomial reduction* of a decision problem $\mathcal{P}'$ to a problem $\mathcal{P}$ is a function f that transforms an instances x' of $\mathcal{P}'$ into an instance x of $\mathcal{P}$ such that

- $\text{size}(x') = O\big(\text{size}\big(P(f(x'))\big)\big)$ where P is a polynomial
- the answer of $\mathcal{P}'$ for x' is yes if and only if the answer of $\mathcal{P}$ for $f(x')$ is yes.

How to prove that a problem $\mathcal{P}$ is $\mathscr{NP}$-complete?

- Prove that $\mathcal{P}$ is in $\mathscr{NP}$
- Prove that an $\mathscr{NP}$-complete problem $\mathcal{P}'$ reduces to (a polynomial number of instances of) $\mathcal{P}$

$\mathscr{NP}$-hard problem

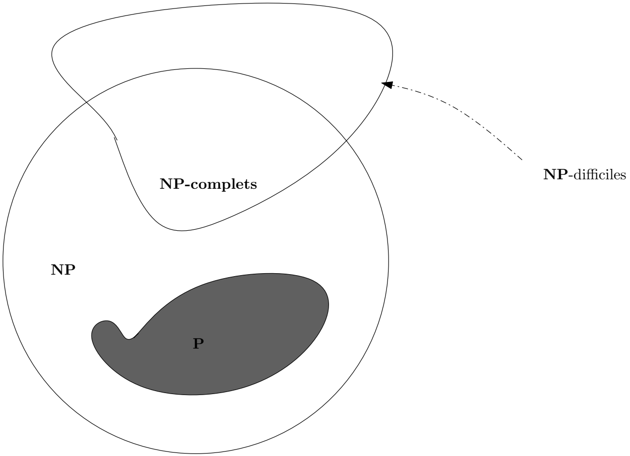
A problem is $\mathscr{NP}$-hard if

- $\mathcal{P}$ is at least as difficult as any problem in $\mathscr{NP}$.

Only decision problems can be $\mathscr{NP}$-complete. Optimization and decisions problems can be $\mathscr{NP}$-hard.

How to prove that a problem $\mathcal{P}$ is $\mathscr{NP}$-hard?

- Prove that an $\mathscr{NP}$-complete problem $\mathcal{P}'$ reduces to $\mathcal{P}$


NP-complets
NP
P
NP-difficiles

$$\mathscr{P} \stackrel{?}{=} \mathscr{N}\mathscr{P}$$

funded by the Clay institute

1. Prove that the problem of existence of a Hamiltonian path is $\mathscr{NP}$-complete, knowing that the problem of existence of a Hamiltonian cycle is $\mathscr{NP}$-complete
2. Prove that the problem of existence of a Hamiltonian cycle is $\mathscr{NP}$-complete, knowing that the problem of existence of a Hamiltonian path is $\mathscr{NP}$-complete