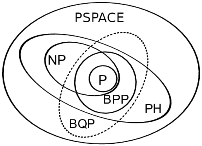
Basics of Computational Complexity

Arnaud Casteigts



Part of Graph Algorithms (14X061)
Masters of Computer Science

University of Geneva

Types of problems

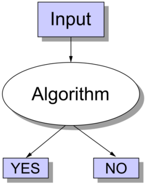
- **Decision:** $\{0,1\}^* \to \{0,1\}$
  Is the picture a cat?
  Is there a path from A to B?
- **Search:** $\{0,1\}^* \to \{0,1\}^*$
  Find the cat on the picture.
  Give me a path from A to B.
- **Counting:** $\{0,1\}^* \to \mathbb{N}$
  How many cats are there on the picture?
  How many paths are there from A to B?
- **Optimisation:** $\{0,1\}^* \to \{0,1\}^*$
  Find the cutest cat in the picture.
  Find a shortest path from A to B

Decision problem (focus)

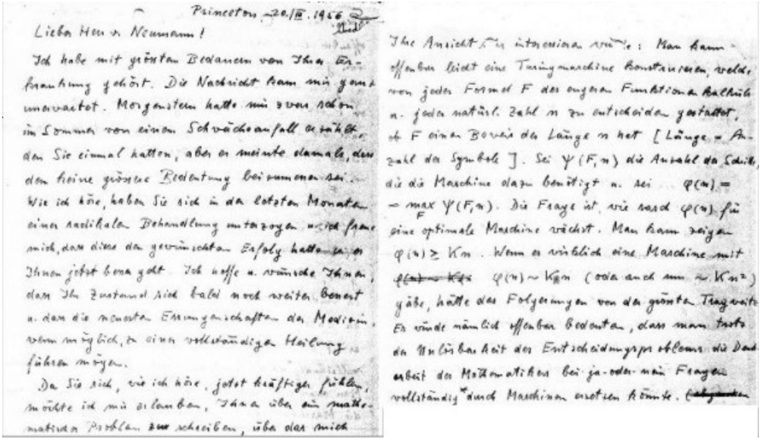
Functions F of type $\{0,1\}^* \to \{0,1\}$ (answer YES or NO)

Set of **positive instances** $\{x \in \{0,1\}^* \mid F(x) = 1\}$ defines a **formal language** L

Solving a decision problem $\equiv$ deciding the corresponding language



Gödel's letter to Von Neumann (1956)



Princeton, 20./III. 1956

Lieber Herr v. Neumann!

Ich habe mit größtem Bedauern von Ihrer Erkrankung gehört. Die Nachricht kam mir ganz unerwartet. Morgenstern hatte mir zwar schon im Sommer von einem Schwächeanfall erzählt, den Sie einmal hatten, aber er meinte damals, dass dem keine größere Bedeutung beizumessen sei. Wie ich höre, haben Sie sich in den letzten Monaten einer radikalen Behandlung unterzogen u. ich freue mich, dass diese den gewünschten Erfolg hatte u. es Ihnen jetzt besser geht. Ich hoffe u. wünsche Ihnen, dass Ihr Zustand sich bald noch weiter bessert u. dass die neuesten Errungenschaften der Medizin, wenn möglich, zu einer vollständigen Heilung führen mögen.

Da Sie sich, wie ich höre, jetzt kräftiger fühlen, möchte ich mir erlauben, Ihnen über ein mathematisches Problem zu schreiben, über das mich Ihre Ansicht sehr interessieren würde: Man kann offenbar leicht eine Turingmaschine konstruieren, welche von jeder Formel F des engeren Funktionenkalküls u. jeder natürl. Zahl n zu entscheiden gestattet, ob F einen Beweis der Länge n hat [Länge = Anzahl der Symbole]. Sei $\psi(F,n)$ die Anzahl der Schritte, die die Maschine dazu benötigt u. sei $\varphi(n) = \max_F \psi(F,n)$. Die Frage ist, wie rasch $\varphi(n)$ für eine optimale Maschine wächst. Man kann zeigen $\varphi(n) \geq K n$. Wenn es wirklich eine Maschine mit ~~$\varphi(n) \sim K n$~~ $\varphi(n) \sim K n$ (oder auch nur $\sim K n^2$) gäbe, hätte das Folgerungen von der größten Tragweite. Es würde nämlich offenbar bedeuten, dass man trotz der Unlösbarkeit des Entscheidungsproblems die Denkarbeit des Mathematikers bei ja-oder-nein Fragen vollständig* durch Maschinen ersetzen könnte.

Gödel's letter to Von Neumann (1956)

I would like to allow myself to write you about a mathematical problem, of which your opinion would very much interest me. One can obviously construct a Turing machine, which for every formula F in first order predicate logic and every natural number n, allows one to decide if there is a proof of F of length n (length = number of symbols). Let $\Psi(F, n)$ be the number of steps the machine requires for this and let $\varphi(n) = \max_F \Psi(F, n)$.

The question is how fast $\varphi(n)$ grows for an optimal machine. (...) If there really were a machine with $\varphi(n) \sim n$ (or even $\sim n^2$), this would have consequences of the greatest importance. Namely, it would mean that (...) the mental work of a mathematician concerning Yes-or-No questions could be completely replaced by a machine.

(...) It would be interesting to know, for instance, the situation concerning the determination of primality of a number and how strongly in general the number of steps in finite combinatorial problems can be reduced with respect to simple exhaustive search.

... and here is computational complexity!

Computational complexity?

Amount of required resources for solving a problem.

What type of resources?

- Time (number of operations)
- Space (amount of memory)
- Non-determinism?
- Randomness?
- ...

Asymptotic point of view

- **Evolution** of these quantities as a function of the input **size** n, when $n \to \infty$
- Notations $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot), \omega(\cdot)$ (ignores **constant factors** and **dominated terms**)

  Intuition $\leq$ $\geq$ $=$ $<$ $>$ Ex: $3n^2 + 5n + 4 = \Theta(n^2)$
- Some adjectives:

Constant	$\Theta(1)$	Quadratic	$\Theta(n^2)$
Logarithmic	$\Theta(\log n)$	Exponential	$\Theta(2^n)$ or $\Theta(2^{n^{O(1)}})$
Linear	$\Theta(n)$	Factorial	$\Theta(n!)$
Quasi-linear	$\Theta(n \log n)$	Polynomial	$O(n^c) = n^{O(1)}$

In general, we are interested in the **worst case** (maximum over all possible instances of a problem).

Time and space

Generic classes

- $\mathrm{TIME}(f(n))$: Decision problems solvable in time $O(f(n))$ (regardless of space).
- $\mathrm{SPACE}(f(n))$: Decision problems solvable in space $O(f(n))$ (regardless of time).

Well-known particular cases

Name	Solvable in...	Definition
LOGSPACE	logarithmic space	$\mathrm{SPACE}(\log n)$
P	polynomial time	$\mathrm{TIME}(n^{O(1)})$
PSPACE	polynomial space	$\mathrm{SPACE}(n^{O(1)})$
EXP	exponential time	$\mathrm{TIME}(2^n)$

$$\overbrace{\mathrm{P} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}}^{\subsetneq}$$

$$\underbrace{\mathrm{LOGSPACE} \subseteq \mathrm{P} \subseteq \mathrm{PSPACE}}_{\subsetneq}$$

$$\mathrm{LOGSPACE} \subseteq \mathrm{P} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXP}$$

The most important is P

Problems solvable "efficiently" (in time $n^{O(1)}$). (robust / composable / realistic)

Class NP

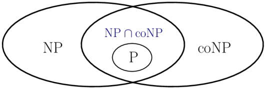
Several definitions, the simplest is:

NP: $\exists$ **short proof** that the answer is YES (if it is YES) – *a.k.a.* positive certificate

coNP: $\exists$ **short proof** that the answer is NO (if it is NO) – *a.k.a.* negative certificate

Short proof = **verifiable** in polynomial time

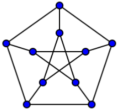
Observation: $\mathrm{P} \subseteq \mathrm{NP}$ and $\mathrm{P} \subseteq \mathrm{coNP}$ (the algorithm itself can be used as a verifier)

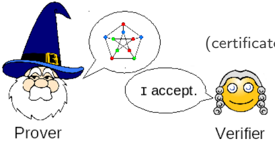


Some problems in NP (presumably not in P): 3-COLORING, CLIQUE, TSP, FACTORISATION, SAT, ...

Exemple: 3-COLORING

Can this graph be colored with 3 colors?



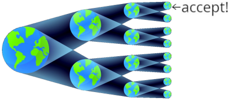


(certificate = the coloring itself)

Historical definition

NP = Non-deterministic Polynomial time

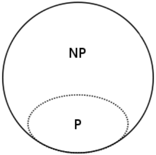
Intuition: ability to “**guess**” the certificate (that it suffices to verify afterwards).



P versus NP

Does "easy to verify" imply "easy to solve"?

(Does $\mathrm{P} = \mathrm{NP}$?)



Relevance of the question

- Most practical problems are in NP.
  If $\mathrm{P} = \mathrm{NP}$, all of them can be solved efficiently.
- Would it be a good news? Yes and no (cryptography).
- One of the 7 "problems of the millenium" (Clay fundation, \$1 M / pb), along with Riemann's conjecture.

Philosophical implications?

- Math: can all humanly verifiable statement be settled by an algorithm?
- More generally: can we mechanize intuition?
- I can recognize a beautiful symphony, does it mean I could have composed it myself?
- *etc.* debate: formalization + how about $O(n^{100})$?

As of today, we don't know the answer.

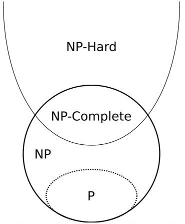
But most of the specialists believe $\mathrm{P} \neq \mathrm{NP}$.

NP-complete problems

Hardness and completeness

- NP-hard: problems **at least as hard** as any problem in NP

  Can be shown through **reductions** among problems.
- NP-complet: both in NP and NP-hard



How to show that a problem is NP-hard?

$\rightarrow$ find a problem that is already NP-hard and **reduce** it to your problem (in polynomial time).

Examples of NP-complete problems

- SAT: $(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_4}) \vee \ldots$ (Cook, Levin'71)
- 3-Coloring, Clique, Set Cover, Hamiltonian Cycle, Tsp,
- Thousands of problems...

If **any** of these problems turns out to be in P, then they all are and $\mathrm{P} = \mathrm{NP}$.

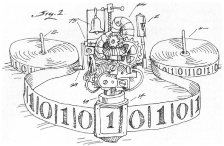
If **any** of these problems turns out not to be in P, then none of them are and $\mathrm{P} \neq \mathrm{NP}$.

Important reminder

This theory focus on **worst case** complexity. Many instances from the real world are solvable in practice.

The real world is often nicer than an adversary.

What about AI?



v.s.

The computational complexity framework applies to AI, without distinction.

Many NP-complete problems are **easy on average**, so nothing precludes that AI can learn to solve them in most of the cases, but it won't do it in the **worst case**.

Some problems are **hard on average** and will remain out of reach for AI. Presumably, FACTORING is one example.

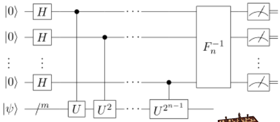
How about quantum computers?

BPP: *Bounded-error probabilistic polynomial time*

- Problems solvable in polynomial time by a **randomized algorithm** (can flip coins) with a probability of error lower than 1/2.

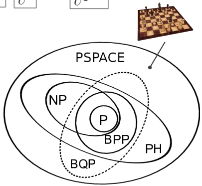
BQP: *Bounded error quantum polynomial time*

- Problems solvable in polynomial time by a **quantum computer** with a probability of error lower than 1/2.



What do we know?

- P ⊆ BPP (0 < 1/2)
- BPP ⊆ BQP (non-reversibility simulable in polynomial time)
- BQP ⊆ PSPACE (Bernstein et Vazirani'97)
- FACTORING ∈ BQP (Shor'94)
- How about BQP *versus* NP ? (expected incomparable)



(expected structure)

Is it expected that a quantum computer can solve NP-complete problems?

→ Unlikely (would contradict many plausible conjectures).

Some graph problems (decision version)

- SHORTEST PATH (G, u, v, k): Does G admit a path of length at most k from u to v? $\in$ P
- LONGEST PATH (G, u, v, k): Does G admit a path of length at least k from u to v? NP-complete
- MATCHING (G, k): Are there k edges in G that share no vertex in common $\in$ P
- CLIQUE (G, k): Does G admit a clique of size k? NP-complete
- INDEPENDENT SET (G, k): Are there k vertices in G, none of them being neighbors? NP-complete
- DOMINATING SET (G, k): Is there a set of k vertices in G s.t. all nodes are either in the set or have a neighbor in the set? NP-complete
- VERTEX COVER (G, k): Are there k vertices that collectively touch every edge? NP-complete
- COLORING (G, k): Can G be properly colored with k colors? $\in$ P (if $k < 3$) NP-complete (if $k \geq 3$)
- HAMILTONIAN CYCLE (G): Does G admit a simple cycle that visits every vertex? NP-complete
- TSP (G, k): Does G admit a simple cycle of cost $\leq k$ that visits every vertex? NP-complete
- GRAPH ISOMORPHISM (G_1, G_2): Are G_1 and G_2 isomorphic? (i.e. structurally identical) NP-intermediate?

You'll play with some of these problems in exercises and we'll use them in subsequent classes.