中国科学院大学
University of Chinese Academy of Sciences

CS101

Algorithmic Thinking

P vs. NP

zxu@ict.ac.cn
zhangjialin@ict.ac.cn

Outline

- What is algorithmic thinking
- Divide-and-conquer paradigm
- Other interesting paradigms
- P vs. NP
  - Time complexity
  - P and NP
  - NP examples

Thinking problem

- Hanoi game
  - 2^n – 1 steps

- If there are 4 piles in the Hanoi game, how many steps do we need?

Hard problems versus easy problems

- Complexity of an algorithm: total steps run by the algorithm
  - Usually consider the worst case
  - Example: bubble-sort algorithm, quick-sort algorithm

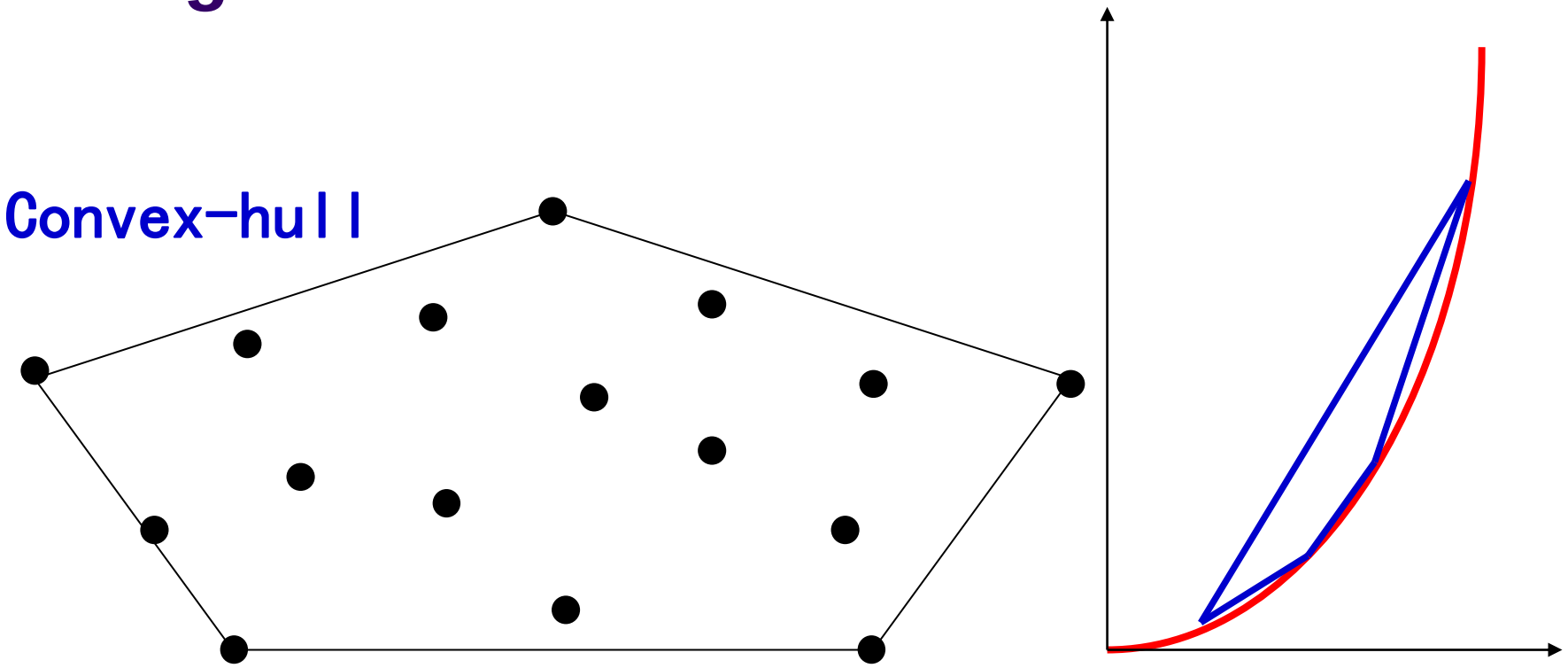
- Complexity of a problem: the complexity of the best algorithm which can solve this problem
  - Sorting problem based on comparison: $O(n \log n)$
  - Integer multiplication: $O(n \log n)$

Reduction

- Suppose A and B are two computing tasks
  - We can have a reduction from problem A to problem B (written as $A \leq_P B$), if
  - Given any algorithm which can solve problem B, we can "use" this algorithm to solve problem A

- Intuition: A is "easier" than B

Sorting versus Convex-hull

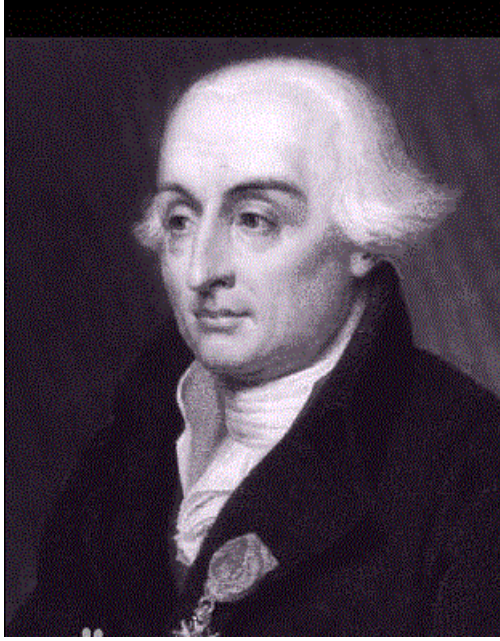


- sorting $\leq_P$ convex-hull
  - The input of sorting problem: $x_1, x_2, \ldots, x_n$ $(x_i > 0)$
  - Construct input for convex-hull problem
    - $P_1(x_1, x_1^2)$, $P_2(x_2, x_2^2), \ldots, P_n(x_n, x_n^2)$

Example of reduction

- Problem A:
  - Determine whether a polynomial equation with integral coefficients has integer solutions.
- Problem B:
  - Determine whether a polynomial equation with integral coefficients has non-negative integer solutions.
- Examples: $x^3 + y^3 = z^3,\ x^3 + y^3 = z^3 + u^3$

- Prove: $A \leq_P B,\ B \leq_P A$



Lagrange
1736~1813

- **A $\leq_P$ B**:
  - $f(x, y, z) \rightarrow F(p, q, s, t, u, v) = f(p\text{-}q, s\text{-}t, u\text{-}v)$

- **B $\leq_P$ A**:
  - $F(x, y) \rightarrow f(a, b, c, d, p, q, s, t) = F(a^2+b^2+c^2+d^2, p^2+q^2+s^2+t^2)$
  - $23 = 3^2 + 3^2 + 2^2 + 1^2$

Lagrange four-integer-squares theorem

P vs NP: P

- P class (Polynomial time): we say a problem $\in P$ if there exists an algorithm A to solve such a problem with time complexity $O(n^c)$, where c is a constant
  - n: input size
  - Polynomial time: $O(n), O(n^2), O(n^3), O(n^{10000}), O(n^{2^{100}})$
  - P class $\approx$ the problems which can be efficiently solved by a computer

- Equivalent definition: the tasks which can be solved by a Turing machine within polynomial steps

P vs NP: NP

- P: the decision problems which can be solved by a deterministic Turing machine in polynomial steps
- NP: the decision problems which can be solved by a non-deterministic Turing machine in polynomial steps
- An equivalent definition of NP:
  - A decision problem belongs to NP, if its solution can be verified by a deterministic Turing machine in polynomial steps

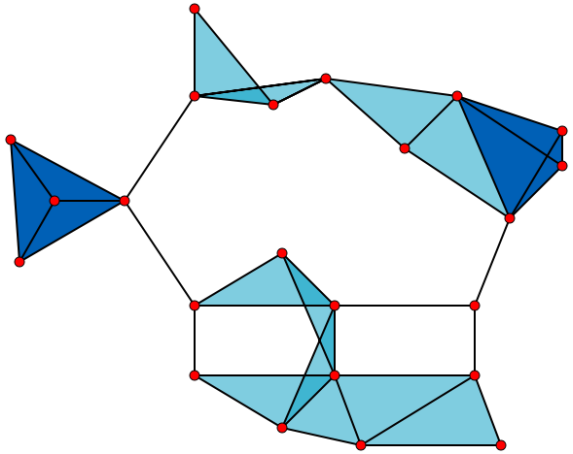
P vs NP: NP

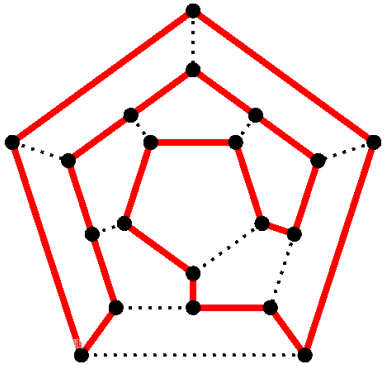
- The **class NP** consists of those problems whose answer can be verifiable in polynomial time
  - There exists a polynomial time verification algorithm A, for any input:
    - If the correct answer is YES, then there exists a witness with which the algorithm A can accept
    - If the correct answer is NO, then no witness can make the algorithm A accept
- $P \subseteq NP$
  - No witness is needed for problems in P

Examples of NP

- Given a map, can we color each area with red or blue or green so that no adjacent areas have the same color? (3-coloring problem)
  - Witness: a color assignment
  - Verification algorithm: verify that each area is colored with red or blue or green; verify that all adjacent areas have different colors.
- Given Boolean formula ϕ, decide whether there exists an assignment so that the value of such formula under this assignment is TRUE? (SAT problem)
  - Witness: an assignment

Examples of NP





- Given a graph, decide whether there exists a Hamiltonian cycle.
  - Hamiltonian cycle: a cycle which passes each vertex exactly once
  - Witness: a cycle
- Given a graph and a parameter k, decide whether there exists a k-clique.
  - Clique: a subset of vertices such that there is an edge between every pair of vertices in this subset
  - Witness: k vertices

P vs NP: NP

- The **class NP** consists of those problems whose answer can be verifiable in polynomial time
  - Exist a polynomial time verification algorithm A, for any input:
    - If the correct answer is YES, then there exists a witness with which the algorithm A can accept
    - If the correct answer is NO, then no witness can make the algorithm A accept
  - The length of witness is polynomial over the length of input

NP-completeness

- So far, we do not know whether these examples belong to P or not
  - 3-coloring problem
  - SAT problem
  - Hamiltonian problem
  - Clique problem
- They are NP-complete problems.
- NP-complete: the hardest problems in NP
  - If we find a polynomial time algorithm for any NP-complete problem, then all problems in NP have polynomial time algorithm, that is, P=NP

Is there any problems outside NP?

- Given $n \times n$ board, two clever players play general Go game. Does the player with black always win?
  - So far, we do not know whether this problem is in NP or not
- Halting problem
  - Not in NP

Millennium Prize

- Birch and Swinnerton-Dyer Conjecture
- Hodge Conjecture
- Navier-Stokes Equations
- **P vs NP**
- Poincaré Conjecture (solved)
- Riemann Hypothesis
- Yang-Mills Theory

First Clay Mathematics Institute Millennium Prize Announced: Prize for Resolution of the Poincaré Conjecture Awarded to Dr. Grigoriy Perelman

- If $P \neq NP$
  - Cryptography!

Cryptography – one way function

- One-way function
  - Cornerstone of cryptography
  - Given x, it is easy to compute $f(x)$: in P
  - Given $f(x)$, it is hard to compute x: not in P

- Is there a one-way function?
  - If it exists, we have $P \neq NP$

Cryptograph - Integer Factorization Problem

- Integer Factorization Problem
  - The most important candidate of one-way function
  - Given p and q, it is easy to compute $n = p \times q$
  - Given n, it is hard to find p and q such that $n = p \times q$
- Based on this problem
  - RSA public key algorithm
  - Ron Rivest, Adi Shamir, Leonard Adleman (1977)