Algorithms

Tractable and Intractable Algorithms

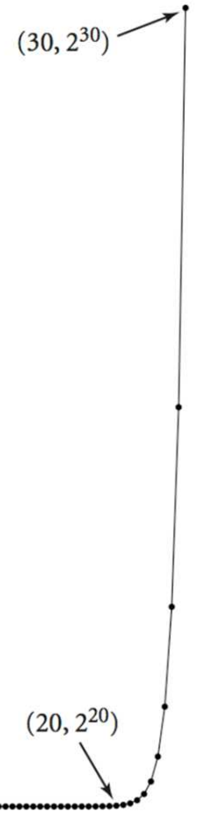
Paolo Camurati and Stefano Quer
Dipartimento di Automatica e Informatica
Politecnico di Torino

Exponential growth

- Exponential growth dwarfs technological change
- Example
  - The **Travelling Salesman Problem Algorithm** on **n** points needs **n!** steps using **brute force**
  - Suppose
    - We have a giant parallel computing device …
    - With as many processors as electrons in the universe …
    - Where each processor has power of today's supercomputers …
    - And each processor works for the life of the universe…

Exponential growth

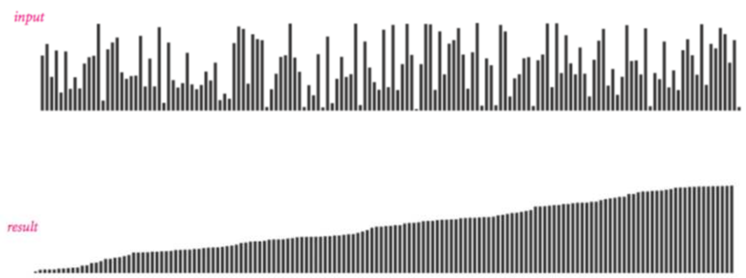
Quantity	Value
Electrons in universe	10^{79}
Instruction per seconds (supercomputers)	10^{13}
Age of universe (seconds)	10^{17}



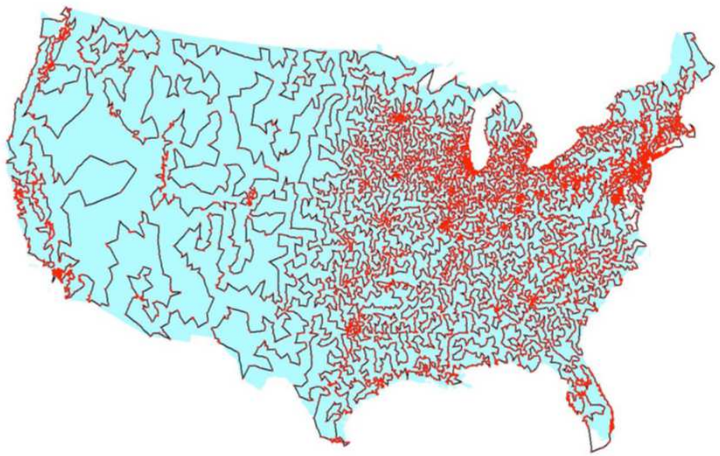
- Then
  - $1000! >> 10^{1000} >> 10^{79} \cdot 10^{13} \cdot 10^{17}$
- The parallel machine will not help solve 1000 TSP problems via brute force

Exponential growth

- Which problems can be solved in practice?
  - Those with poly-time algorithms
- Which problems have poly-time algorithms?
  - Not so easy to know !



Many known poly-time algorithms for sorting



No known poly-time algorithms for TSP

Growth

- Which of these problems have poly-time algorithms?
  - LSOLVE: Given a system of linear equations, find a solution
    - Gaussian elimination solves N-by-N system in N^3 time
  - LP: Given a system of linear inequalities, find a solution
    - Ellipsoid algorithm is poly-time
  - ILP: Given a system of linear inequalities, find a 0-1 solution
    - No poly-time algorithm known or believed to exist !
  - SAT: Given a system of boolean equations, find a binary solution
    - No poly-time algorithm known or believed to exist !

The P Class

- Decidable and tractable decision problems
  - There exists a polynomial algorithm that solves them (Edmonds-Cook-Karp thesis, 1970s)
  - That is, P problems are solvable in polinomial time
    - An algorithm is polynomial iff, working on n data, given a constant c>0, it terminates in a finite number of steps upper-bounded by n^c
    - In practice c should not exceed 2
  - Problems in P are supposed to be tractable

Most of the problems we are going to consider are in P

The NP Class

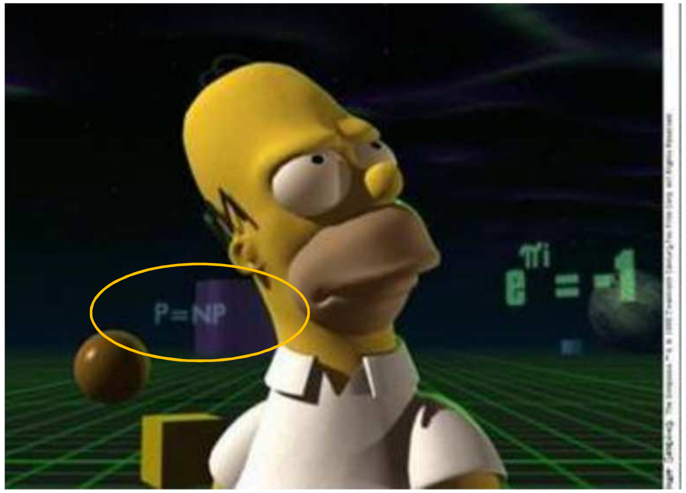
- Nondeterministic machine can guess the desired solution to a problem
- Example
  - int v[N] = {0};
  - Initializes entries to 0
  - A nondeterministic machine may inizialize entries to the final solution
- NP problems are problems solvable in poly time on a nondeterministic machine

The NP Class

- NP stands for Non-deterministic Polynomial
- There exist decidable problems for which
  - We have exponential algorithms, but we don't know any polynomial algorithms
  - However we can't rule out the existence of polynomial algorithms
- We have polynomial verification algorithms, to check whether a solution (certificate) is really such
  - Sudoku, satisfyability of a boolean function, factorization, graph isomorphism

P versus NP

- Thus
  - P = class of search problems solvable in poly-tyme
  - NP = class of all search problem
- Does P = NP ?



Copyright © 1990, Matt Groening

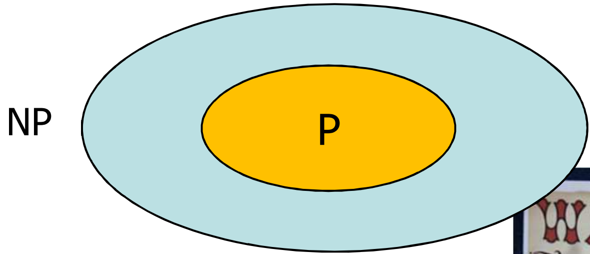
Copyright © 2000, Twentieth Century Fox

P versus NP

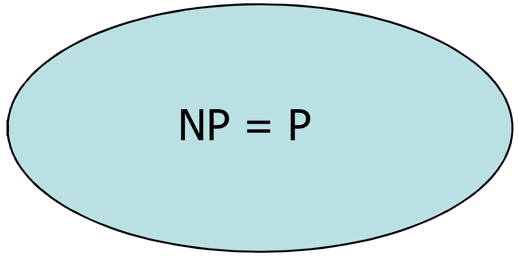
- We know that $P \subseteq NP$
- We don't know whether P is a proper subset of NP or it coincides with NP

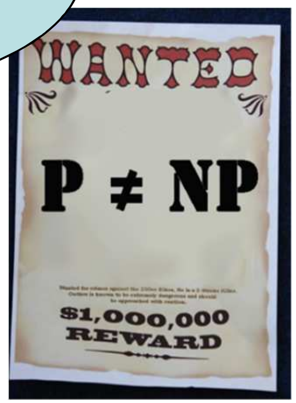
❖ It is probable that P is a proper subset of NP

Probable



Improbable





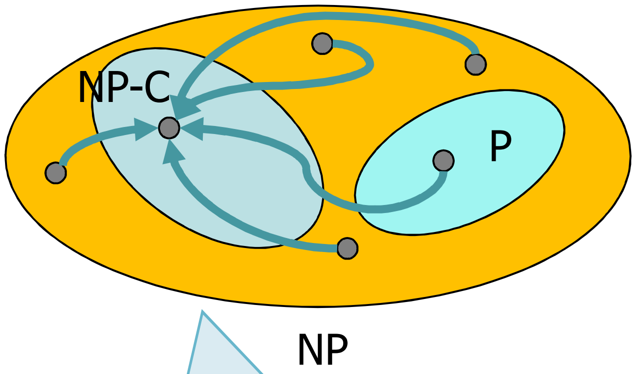
Millennium prize
$1 million for resolution

The NP-C Class

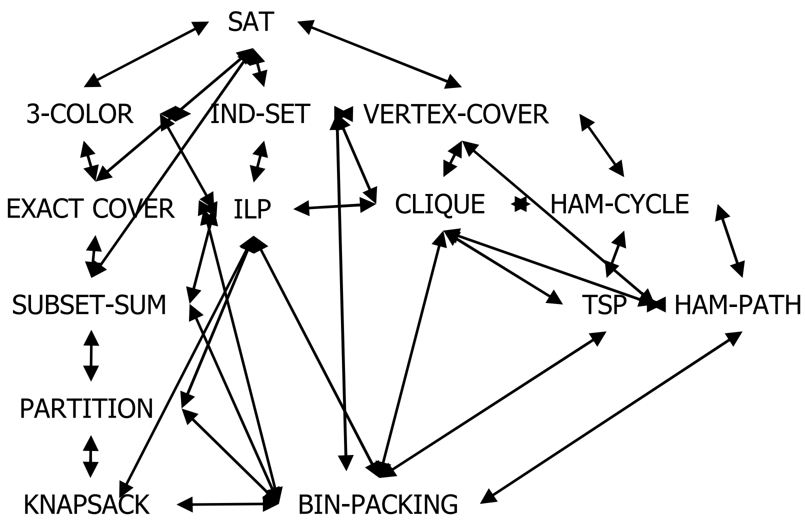
- Definition
  - An NP problem is NP-complete if every problem in NP poly-time reduce to it
- Problems in NP-C are the hardest within NP

The NP-C Class

- A problem is NP-complete if
  - It is NP
  - Any other problem in NP may be reduced to it by means of a polynomial transformation



NP and NP-C problems are often considered as intractable



P versus NP versus NP-C

- If we find a polynomial algorithm for any problem in this class, we could find polynomial algorithms for all NP problems, through transformations
- This is **highly improbable** !
- The existence of the NP-C class makes it probable that P $\subset$ NP

- Example of NP-C problem
  - Satisfyiability
    - Given a Boolean function, find if there exists an assignment to the input variables such that the function is true.
  - Hamilton Cycle, Clique, Graph Connectivity, Primality, Determinant

The NP-H Class

- A problem is NP-hard if every problem in NP may be reduced to it in polynomial time (even if it does not belong to NP)
- Any other problem in NP may be reduced to it by means of a polynomial transformation
  - Permanent of a matrix