



Algorithms



Paolo Camurati
Dip. Automatica e Informatica
Politecnico di Torino



Algorithm

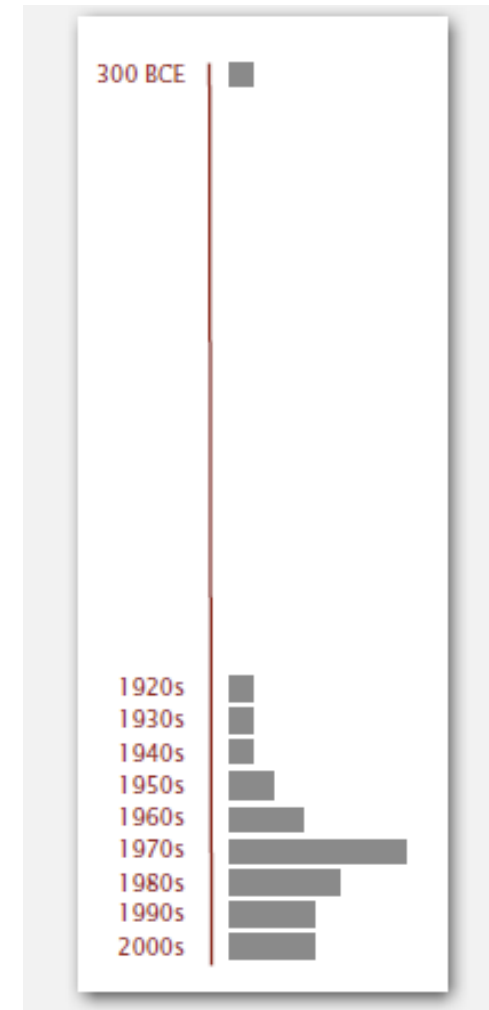
Finite sequence of instructions that

- Solve a problem
- Satisfy the following criteria
 - They receive input values
 - They produce output values
 - They are clear, unambiguous and executable
 - They terminate after a finite number of steps
- They work on data structures

Algorithm: from al-Huarizmi, Persian IX cent. DC mathematician

Why study algorithms?

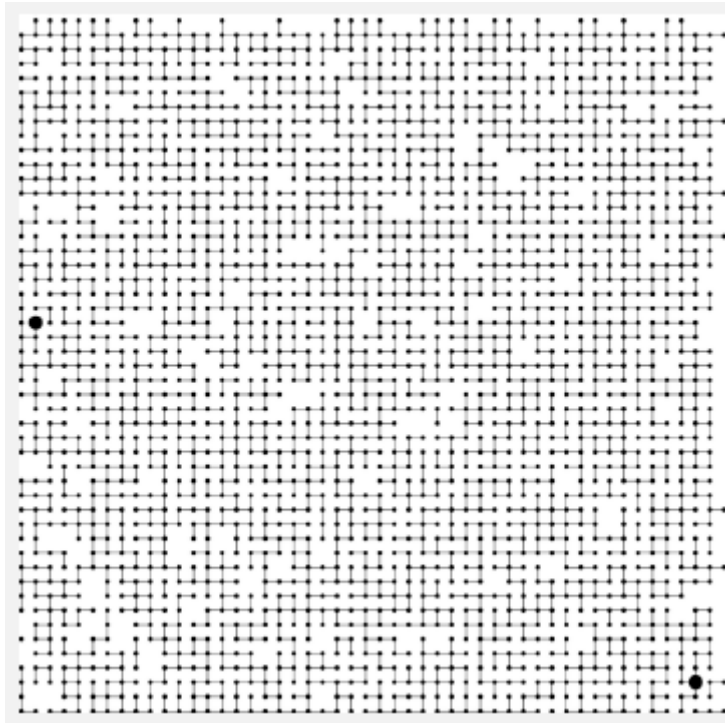
- Algorithms have ancient roots
 - Euclid (IV cent, B.C.)
 - Formalization by Church and Turing (XX cent., '30s)
 - Recent developments



Why study algorithms?

- To do something otherwise impossible
 - Are the two dark dots connected in this network (network connectivity)?

Network
connectivity





Why study algorithms?

- To solve problems in many domains
 - Internet: Web search, packet routing, distributed file sharing
 - Biology: human genome
 - Computers: CAD tools, file systems, compilers
 - Graphics: virtual reality, videographics
 - Multimedia: MP3, JPG, DivX, HDTV
 - Social Networks: recommendations, news feed, advertisement
 - Security: e-commerce, cell phones
 - Physics: particle collision simulation ...



Why study algorithms?

- For intellectual stimulation
- To become a proficient programmer

“ I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships. ”

— Linus Torvalds (creator of Linux)



Why study algorithms?

- To unlock the secrets of life and of the universe by creating models
 - In many sciences computational models are replacing mathematical ones

$$E = mc^2$$

$$F = ma$$

$$F = \frac{Gm_1m_2}{r^2}$$

$$\left[-\frac{\hbar^2}{2m} \nabla^2 + V(r) \right] \Psi(r) = E \Psi(r)$$

Mathematical formulae

```
for (double t = 0.0; true; t = t + dt)
  for (int i = 0; i < N; i++)
  {
    bodies[i].resetForce();
    for (int j = 0; j < N; j++)
      if (i != j)
        bodies[i].addForce(bodies[j]);
  }
```

Computational model



Why study algorithms?

- For fun
- For money
- To increase speed
- To process more data
- To satisfy intellectual curiosity



Types of problems

- Decision problems
 - Problems with a yes/no answer
 - Examples
 - Given 2 integers x and y , does x exactly divide y ?
 - Given a positive integer x , is it prime?
 - Given a positive integer n , do 2 positive and > 1 integers p and q exist such that $n = pq$?
 - Determine whether a number is prime

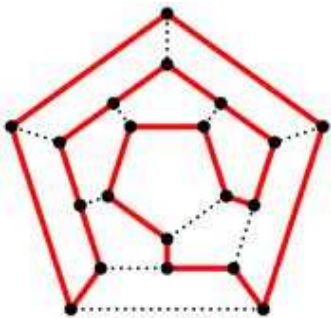
Types of problems

- Search problem

- Does a valid solution exist and which one is it?

- Examples

- Hamiltonian cycle: given an undirected graph, does a simple cycle spanning all vertices exist? Which one is it?



- Which one is the k-th prime number

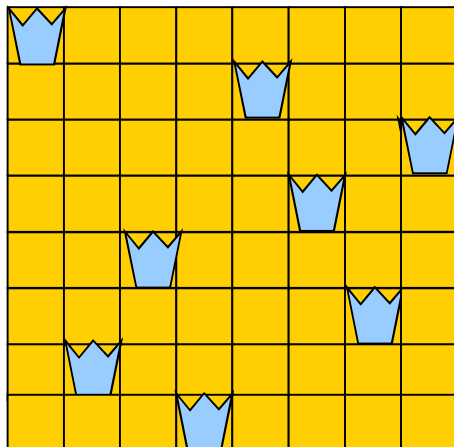
- Given an array of integers, sort it in ascending order

Types of problems

■ Verification problems

- Given a solution (certificate), make sure that it is really one
- Examples
 - Sudoku
 - The eight queen problem

8 queens



Sudoku

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

Types of problems

■ Optimization problems

- If a solution exists, which one is the best one?

- Examples

- Given a weighted directed graph, which is the shortest simple path, if it exists, between nodes i and j ?





Decision Problems

- Decision problems may be
 - Decidable
 - There exists an algorithm that solves them
 - Undecidable
 - There is no algorithm that solves them
- Decidable decision problems may be
 - Tractable, i.e., solvable in “reasonable” time
 - Intractable, i.e. non solvable in “reasonable” time



The P Class

Decidable and tractable decision problems

\Leftrightarrow

\exists a polynomial algorithm that solves them
(Edmonds-Cook-Karp thesis, '70)

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

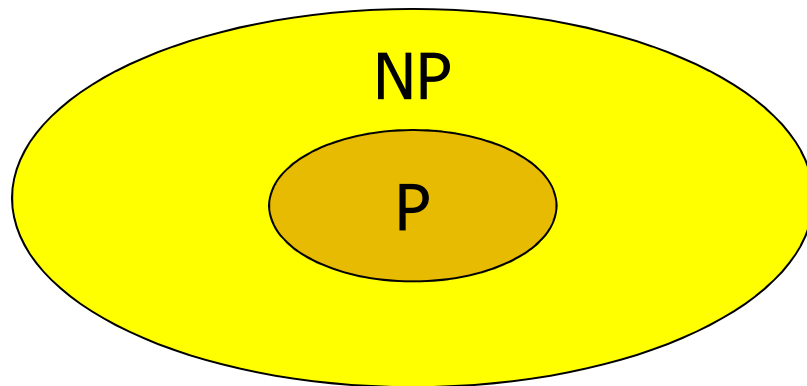


The NP Class

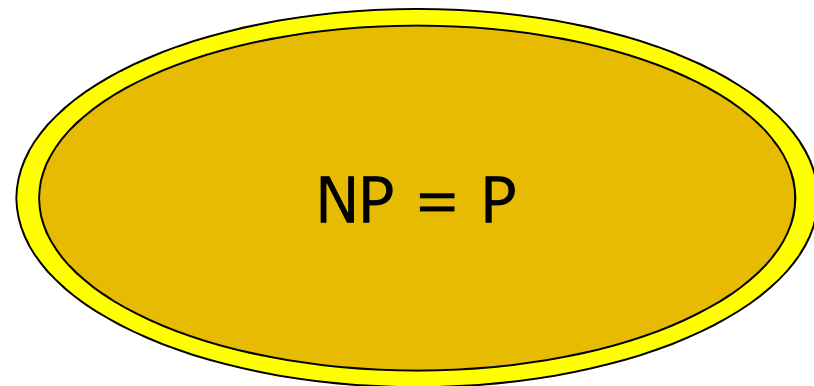
- NP stands for Non-deterministic Polynomial
- There exist decidable problems for which
 - We have exponential algorithms, but we don't know any polynomial algorithms
 - However we can't rule out the existence of polynomial algorithms
- We have polynomial **verification** algorithms, to check whether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

P versus NP

- $P \subseteq NP$, but 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

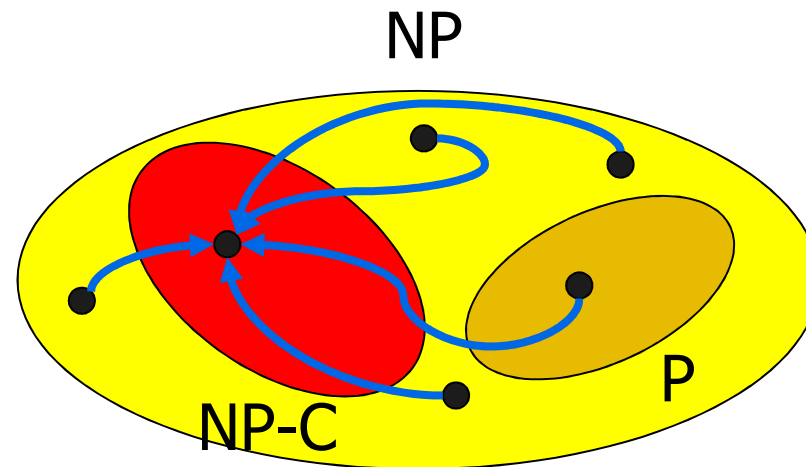


Improbable

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





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

HIGHLY IMPROBABLE!

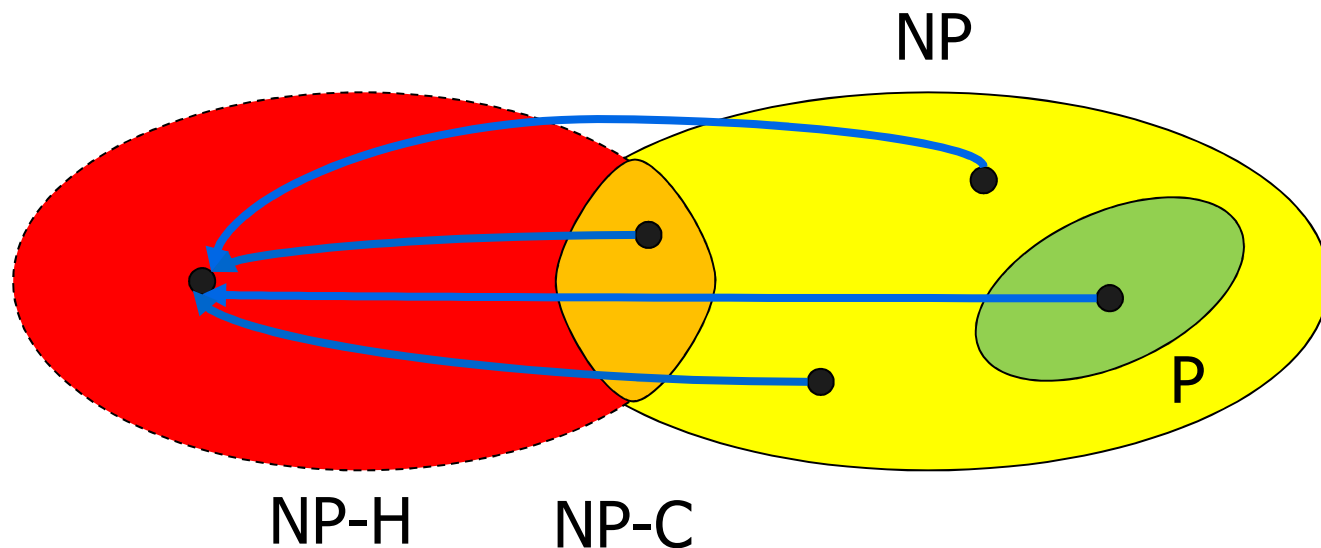
The existence of the NP-C class makes it probable that $P \subset NP$

Example of NP-C problem: satisfiability

given a Boolean function, find if there exists an assignment to the input variables such that the function is true.

The NP-H Class

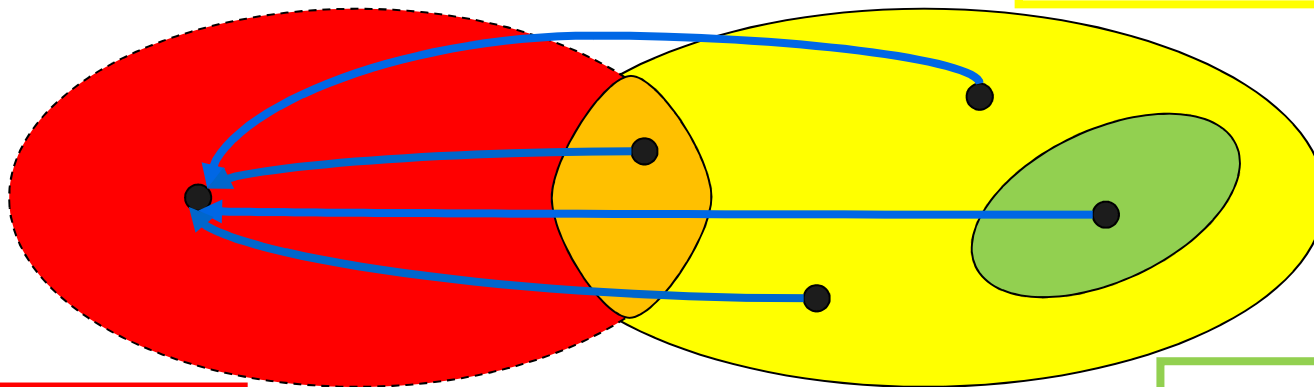
- A problem is NP-**hard** if every problem in NP may be reduced to it in polynomial time (even if it doesn't belong to NP)
- Any other problem in NP may be reduced to it by means of a polynomial transformation



The NP-H Class

NP:

- Factorization
- Graph isomorphism



NP-H:

- Permanent of a matrix

NP-C:

- Satisfiability
- Hamilton Cycle
- Clique

P:

- Graph Connectivity
- Primality
- Determinant