Complexity theory

Key terms

- Polynomial time algorithm
- *P* class
- *NP* class
- Decision problem
- Reduction
- More concepts …

Common *NP* problems

- SAT (Satisfiability)
- Hamiltonian
- TSP (Travelling Salesman Problem)

.

. They are known to be *NP*–complete problems.

.

Theorems

Properties

Problem 1

Is Kruskal's algorithm in the *P* class?

Problem 2

"Because Hamiltonian is an *NP*–complete problem, it cannot be solved in polynomial time."

Is this statement true / false?

Problem 3

Assume: $\Pi_1 \sim \Pi_2$

"YES instance of Π_1 is transformed into a NO instance of Π_2"

Is this statement true / false?

Problem 4

University of Veszprém has n societies and organisations $c_1, \ldots, c_i, \ldots, c_n$.

The largest contains m members (a person can be a member of multiple clubs). $1 \le |c_i| \le m, \forall i$

The Rector of the University wishes to hold a dinner to all clubs, and to invite at least one member from each club. $\forall i\ \exists j: g_j \in c_i$

The place can accommodate only k guests. The problem is as follows: construct a guest list of k persons.

Can you provide him with a general method, which he can apply in the future to solve the problem himself? Is the problem NP–complete?

Problem 5

Prove: if a set is recursive, then it is recursively enumerable.

Prove: if a set is recursive, then so is its complement, too.

Problem 6

Consider the HALTING problem.

Consider also the CHANGE problem: it determines whether a given variable changes value (or not) during the execution of a procedure (function) $q()$.

Prove: HALTING ≤ CHANGE

Problem 7

Prove: the P class is closed under complement.

Prove: SAT can be checked in polytime.

Problem 8

Given coNP = $\{ \overline{L} : L \in NP \}$. That is, co$NP$ is the set of languages whose complement is in NP. Prove that if $P = NP$ then $NP =$ coNP. You are given that P is closed under complement.

Problem 9

Given a 3×3 square board filled with eight square tiles numbered from 1 to 8 as shown below.

1	2	3
4	5	6
7	8	

This is the initial pattern.

Given a final pattern $a_1, \ldots, a_8$. Design a back tracking algorithm to obtain the final pattern from the initial.

Challenge: optimise!

Problem 10

TRUE / FALSE?

- Problems in *NP* cannot be solved in polytime.
- Problems in *NP* can be solved in exponential time.
- If SAT ≤ X, then X is *NP*–complete.
- We know that Independent_Set ≤ Clique. If the Clique ≤ Vertex_Cover, then Independent_Set ≤ Vertex_Cover.

Problem 11

Given the following distances between four towns, is there a TSP tour of length 50 or less? Justify.

	B	C	D
A	11	12	15
B		15	11
C			16

Problem 12

Which is larger (for large n), $n^{\log n}$ or $(\log n)^n$? Justify.

Problem 13

Explain the difference between a deterministic and a non–deterministic TM.

Problem 14

Write down a sequence of the letters A to L in such an order that the following lines sorted in this way would give a configuration i for which $f\,i$ is minimal.

```
procedure simulated_annealing
begin
      A) c_{M+1} := h(c_M) ;
      B) repeat
      C) if accept then UPDATE( config. i → config. j );
      D) PERTURB( config. i → config. j , compute Δf ij := f j - f i );
      E) until STOP_CRITERION = true
      F) M := M + 1 ;
      G) INITIALIZE;
      H) if Δf ij < 0 then accept
      I)   else if exp(- Δf ij / c ) > random[0,1) then accept
      J) repeat
      K) until EQUILIBRIUM_IS_APPROACHED_SUFFICIENTLY_CLOSELY
      L) M := 0 ;
end.
```