Project work on Theory Computational Complexity (1 cu)

time vs. space complexity of Turing machines, complexity classes, NP-completeness

The project work follows directly the TFCS course and it can be performed in the same problem-based way as we practised in the course. So you may discuss together about the problems in the groups. However return the results of your own studies as your problem report! In addition to the problems the project contains 10 exercise tasks and a learning diary. The problems give 50% of total points, and both exercises and the learning diary give 25% of points. If you get stuck in bad difficulties, come and ask help!

Deadline **Fri 10.5. before 14.00**

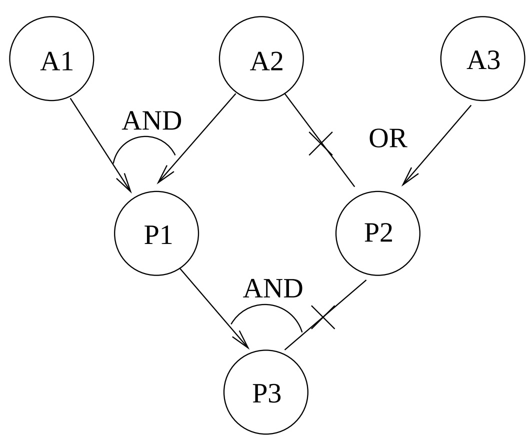
Select three of the following problems!

Case 1

Let's consider the following kind of belief networks: The system is given a set of axioms $A_1, A_2, ..., A_n$ (root nodes) and beliefs $P_1, P_2, ..., P_k$ (other nodes). Each axiom or belief can be either true or false. The truth values of beliefs are derived from axioms or already derived beliefs by AND-, OR- and NOT-operations. For example rules

$A_1\ AND\ A_2 \rightarrow P_1$
$NOT A_2\ OR\ A_3 \rightarrow P_2$
$P_1\ AND\ NOT P_2 \rightarrow P_3$

can be represented as a graph



The system can be used for two kind of reasoning. First you may ask, if the given belief P is true or false, when the axioms are given. Second you may ask, which axioms must hold that given belief P would be true. What are the time requirement classes of both types of reasoning (in the worst case)? If the same graph is used for several queries, what is the best way to implement the reasoning? (Hint: Could you change the time into space?)

Case 2

Maya and Math are playing with blocks of different height. Both of them are building their own towers, but then they begin to argue: are the blocks shared equally? Can both of them build equally high tower?

Help the children and design a nondeterministic Turing machine, which solves children's so called PRT-problem (partitioning problem): You are given n positive integers $x_1, x_2, ..., x_n$. Can the numbers (blocks) be divided into two sets such that the sum of both sets is equal? I.e. is there a partitioning into two index sets I and J such that $\Sigma_{i \in I} x_i = \Sigma_{j \in I} x_j$? (N.B.! The machine has to answer just "Yes" or "No".) Into which time requirement class does the problem belong?

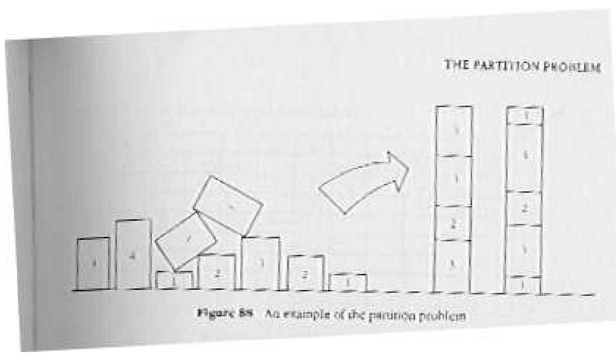


Figure 88 An example of the partition problem

Case 3

You would like to take a lot of books with you to your tour, but there is an absolut weight limit of 30 kg for the luggage in the airplane. You decide to leave everything else but books home, but still you have too many books and have to prune some of them. So you decide to optimize the importance-weight ratio of the books you take with you. You give every book a measure of importance s_i and weigh it (weight w_i). How can you select the optimal collection of books? I.e. you should select from the book collection $1, 2, ..., N$ a subset I uch that $\Sigma_{i \in I} w_i \leq 30$ and $\Sigma_{i \in I} s_i$ is as large as possible.

How hard problem is it?

Case 4

A group of outer space aliens arrives Earth and offers people as a sign of their good will an algorithm, which solves SAT-problem in polynomial time $q(n)$. How could you use the gift?

Consider especially an NP-complete problem CAT[1], which can be solved by a nondeterministic Turing machine in time $p(n)$. Coding CAT-problem as SAT (i.e. as logical formulas, which describe the behaviour of CAT-machine) requires polynomial time $(p(n))^3$. What time does it now take to solve CAT? What about other NP-complete problems?

Exercises

1. Eight competitors have taken part in a programming competition, in which they had to solve a mysterious MYST-problem as efficiently as possible. Put the competitors in order by their time requirement classes (O classes). The time requirements of their programs are following:

   $n \log n$
   n^8
   $n^{1+\epsilon}$
   $(1+\epsilon)^n$
   $(n^2 + 8n + \log^3 n)^4$
   $\frac{n^2}{\log n}$
   $(1-\epsilon)^n$
   1

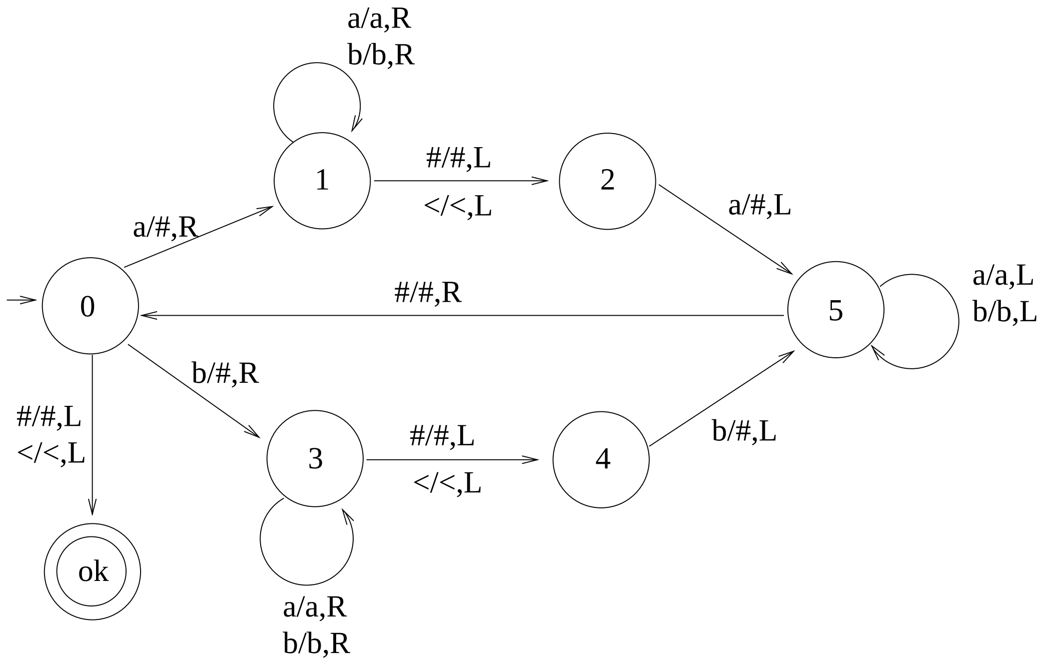
   in which ϵ is constant $0 < \epsilon < 1$.

2. Construct a pushdown automaton and a Turing machine, which recognize the language $L = \{a^n b^n | n \geq 0\}$. What are the time and space requirements of both machines? Are they comparable?

3. Give the time requirement (as a function of input length) in the worst case for the following standard type Turing machine, which recognizes language $\{ww^R | w \in \{a,b\}^*\}$.

   (Hint: The worst cases for time requirement are those inputs, which belong to the language. Consider, how many "sweeps" the machine does on such input, and how many moves it makes during one sweep.)

[1] N.B.! CAT is an imaginary problem, not yet known.



4. (a) Prove that all regular languages belong to the class $DTIME(n+1)$
   (b) Prove that all context-free languages belong to the class P. (Hint: how fast parsing algorithms do you know?)

5. Let C be any language class (especially interesting are the classes $C = NP$, $PSPACE$, etc.). We say that the language A is C-complete, if A belongs to class C and any other language $B \in C$ can be polynomially reduced to A. (Denote: $B {\leq_m}^p A$.) Why we require that the time requirement of the reduction function is polynomial? Why it is not enough that for example the space requirement is polynomial?

6. Prove the following claims:
   a) If A is NP-complete language and $A \in P$, then $P = NP$.
   b) If A is NP-complete language, $B \in NP$ and $A {\leq_m}^p B$, then B is also NP-complete.

   (Hint: Properties of polynomial reductions.)

7. We know that a nondeterministic Turing machine can always be transformed to a deterministic one. Why this doesn't satify to prove that $P = NP$?

8. Prove that the class NP is closed under union and cut! (Hint: Suppose that we have nondeterministic Turing machines M_A and M_B, which recognize the langauges A and B in polynomial time, and construct the corresponding union- and cut-machines.) The class is not known to

be closed under complement. What difficulty do we meet if we try to prove that property?

9. You are a travelling mathematician Paul ErdÃűs giving lectures all around world. You'd like to plan a shortest possible tour between the universities but without visiting any city twice. You have a very efficient algorithm for Hamilton Circle problem. Could you utilize that program in planning your route? Justify your answer!

   (Hint: Polynomial reductions. Add the missing paths to the HC-graph, but give them so large weights that the mathematician's route cannot go though them.)

10. Which is your favourite of the NP-complete problems? In which practical situations you can meet it?