

Combinatorial Search, second week

The second question (out of six) of the exam will be *Cook's theorem and the complexity of variants for SAT*. Three random students present this question.

Consider the following ways of representing finite Boolean functions $f : \{0, 1\}^n \rightarrow \{0, 1\}$:

1. Truth tables.
2. CNF-formulae.
3. DNF-formulae.
4. Unrestricted Boolean formulae with operations $\{\wedge, \vee, \neg\}$.
5. Unrestricted Boolean formulae with operations $\{\wedge, \vee, \neg, \oplus, \Leftrightarrow\}$.
6. Nested "if-then-else-expressions" such as "if (if x_1 then x_2 else x_3) then (if x_2 then 1 else x_1) else 0.
7. Boolean decision trees (a decision tree where leaves contain either 0 or 1 and each internal node contains the name of a variable and has two successors, one for each truth value of the variable).
8. Boolean circuits with operations (gates) $\{\wedge, \vee, \neg\}$.
9. Boolean circuits with operations (gates) $\{\wedge, \vee, \neg, \oplus, \Leftrightarrow\}$.
10. Deterministic finite automata accepting exactly those strings x for which $f(x) = 1$.
11. Turing machines accepting exactly those strings x for which $f(x) = 1$.
12. Turing machines accepting those strings x for which $f(x) = 1$ and rejecting those strings x for which $f(x) = 0$.
13. Pairs (M, t) so that t is a natural number and M is a Turing machine accepting those strings x for which $f(x) = 1$ in at most t steps.

For those objects (e.g., Turing machines) that are not strings, assume some standard representation as strings.

- For which pairs of representations can you convert from one to the other by a polynomial time algorithm?
- For which pairs of representation do you know/can you argue that this is *not* the case?
- Which of these representations do you know/can you argue to be polynomially equivalent?

For each of these representations, consider the corresponding variant of the satisfiability problem (given a function using this representation, can the variables be assigned truth values so that the output is 1?).

- Which of these satisfiability problems can you put in **NP**?
- Which of these problems can you prove *not* to be in **NP**?
- Which of these satisfiability problems can you put in **P**?
- Which of these problems can you prove *not* to be in **P**?
- Which polynomial time reductions can you find among these problems?
- Which of these problems can you show to be **NP**-hard? (assuming Cook's theorem known)
- Which of these problems can you show to be **NP**-complete?
- Which of these problems can you show not to be in **P** under the assumption that **P** \neq **NP**?