Solution to Homework 39

Problem. Describe, in detail, at least three different schemes that use serious but still hypothetical physical processes to solve NP-complete problems in polynomial time.

Comment. The lectures contain several such schemes. In this solution, we present three of them, but if you present some other two schemes, this will also be a correct answer to this question.

Solution. Let us first describe two schemes based on using time machines.

First scheme. The first scheme is very straightforward: we set a computer to solve the corresponding instance of an NP-complete problem by exhaustive search.

This is possible, since by definition, NP-complete problems belong to the class NP, and in each such problem we need:

- given a string x,
- find a string y that satisfies some feasible property $C(x, y)$ and for which the length $\text{len}(y)$ is bounded by some polynomial of the length of x:

$$\text{len}(y) \le P_\ell(\text{len}(x)).$$

So, in principle, we can solve each such instance by trying all possible string y of such limited length. This requires exponential (i.e., not feasible) time, but it is doable.

If we have a time machine, then we can set up the computer so that when it finishes computations, it should use a time machine to send this result back to us, to the time when we started the computations. In this setting, we solve the problem and get the result right away – i.e., clearly, in polynomial time.

Second scheme. The second scheme is based on the fact that, as is well known (at least to those who read science fiction or watch science fiction movies) time travel can lead to paradoxes: e.g., a time traveler can travel into the past and kill his grandfather before his father was conceived – thus, he himself will not exist. Since the time traveler exists, this means that in each such attempt, something prevented him from killing his grandfather. And if we try to ensure that all usual causes of failure are taking care of, there must be some low-probable event that prevented him from killing – e.g., a meteorite falls on his head when he tries

to shoot. In general, the existence of the time machine means that attempts to use it can trigger events with very low probability.

Suppose now that we want to solve an instance of a propositional satisfiability problem, i.e., that:

- we have a propositional formula $F(v_1, \ldots, v_n)$ with n Boolean variables $v_1, \ldots, v_n$, and
- we want to find the values of these variables v_i that make the formula true.

To solve this problem:

- we run n times a random number generator that generate 0 and 1 with probability 1/2 each;
- we plug in the resulting values $r_1, \ldots, r_n$ into the formula F;
- if the formula is satisfied by these values, i.e., if $F(r_1, \ldots, r_n)$ is true – we get the desired solution;
- if the formula F is not satisfied by the values r_i, i.e., if the value $F(r_1, \ldots, r_n)$ is false, we set up a time machine to trigger an event with very low probability $p_0 \ll 1$.

Now, nature has two choices:

- it can generate the satisfying vector, which happens for random number generators with probability 2^{-n}, or
- it can generate a different vector, in which case an event with probability p_0 is triggered.

According to statistical physics, events with higher probability usually occur. So if $p_0 < 2^{-n}$, nature will prefer the first option – and thus, provide the desired solution.

The third scheme. The third scheme is based on the hypothesis that elementary particles are actually gateways to different worlds. In this case, if we, e.g., have a propositional formula $P(x_1, \ldots, x_n)$ with n variables $x_1, \ldots, x_n$, we select two particles, and send, to each of these worlds, a propositional formula with $n-1$ variables: $P(x_1, \ldots, x_{n-1}, 0)$ and $P(x_1, \ldots, x_{n-1}, 0)$. Each of these worlds, in its turn, sends two formulas with $n-2$ variables to other worlds, etc. After n iterations, all if needed for each of finally selected 2^n worlds is to check whether the given expression holds for a given set of values $x_1, \ldots, x_n$. Once a world finds that a formula is true for the corresponding tuple x_i, it sends this tuple back. In n steps, we thus either get a satisfying vector or – if no such tuples came – an information that the propositional formula is never true.