abstract

COMPUTER SCIENCE/DISCRETE MATH II
Topic:

Speaker:

Affiliation:

Date:

Time/Room:

One natural strategy for proving a time lower bound for a problem is proof-by-contradiction: assume that there is a great algorithm for the problem, and apply this algorithm as a subroutine to solve other problems until the algorithms get so great that they do impossible things. Typically the contradiction is with a time hierarchy proved by diagonalization: for instance, one may use the great algorithm to show that all time t algorithms can be simulated in $t^{0.9}$ time. For this reason the strategy is sometimes called "indirect diagonalization."

Since the early 80's, such algorithmic arguments have been used to prove separation results and lower bounds for important problems. I will survey some of them. Possible topics include:

Paul, Pippenger, Szemeredi, and Trotter's result that that NTIME$[n]$ $\neq$ DTIME$[n]$ for multitape Turing machines

Fortnow's result that NL is not equal to any unbounded level of the polynomial time hierarchy

Satisfiability cannot be solved in $n^{2 \cos(\pi/7) - o(1)}$ time and $n^{o(1)}$ space simultaneously

How to transfer these arguments to modular counting problems.For example the above lower bound for satisfiability also holds for counting satisfying assignments modulo 6.

How to formulate the search for lower bounds as a computer search. Namely, the hard work in these proofs can sometimes be replaced by a series of linear programming instances.