Coping with NP-completeness: Introduction

Alexander S. Kulikov

Steklov Institute of Mathematics at St. Petersburg
Russian Academy of Sciences

Advanced Algorithms and Complexity
Data Structures and Algorithms

- Your boss asked you to implement a program that solves efficiently a certain search problem
- If you are lucky enough, the problem can be solved by some known technique like dynamic programming, linear programming, flows (though it is usually still not immediate to notice this)
- Alas, this happens rarely

After two weeks of unsuccessful attempts to implement an efficient program, you come to your boss' office. "I can't find an efficient algorithm, I guess I'm just too dumb."

Michael R. Garey and David S. Johnson.
Computers and Intractability: A Guide to the Theory of NP-Completeness. 1979.

Perhaps there is just no efficient algorithm for your search problem. "I can't find an efficient algorithm, because no such algorithm is possible!"

Michael R. Garey and David S. Johnson.
Computers and Intractability: A Guide to the Theory of NP-Completeness. 1979.

- But currently, we don't have a proof that a certain search problem has no efficient (that is, polynomial) algorithm
- Note that such a proof would resolve the **P** vs **NP** question
- Instead of showing that there is no efficient algorithm for your program, you show that it is one of the hardest search problems
- That is, you show that your problem is **NP**-complete

“I can’t find an efficient algorithm, but neither can all these famous people!”

Michael R. Garey and David S. Johnson.
Computers and Intractability: A Guide to the Theory of NP-Completeness. 1979.

OK, now you know that your problem is **NP**-complete meaning that it is unlikely that there exists an efficient algorithm for solving it. Should you give up?

Keep your head up!

It is just the beginning of a fascinating adventure!

Next Parts

If $\mathbf{P} \neq \mathbf{NP}$, then there is no polynomial time algorithm that finds an optimal solution to an **NP**-complete problem in all cases.

	poly time	optimal solution	all cases
special cases	✓	✓	✗
approximation algorithms	✓	✗	✓
exact algorithms	✗	✓	✓