# P vs NP

**Chaskin Saroff, Alexander Jansing**
State University of New York at Oswego
Department of Mathematics
Oswego, NY, USA
csaroff@oswego.edu, ajansing@oswego.edu

**Abstract**

*The abstract should be a brief overview of the project. Keep specialized notation to a minimum; try to explain to a general audience what the project is about in a way to draw a viewer's attention. Some say that the abstract of a poster should have a larger font size, to catch a viewer's eye as he/she passes along but if you don't want to have a larger font size then delete the sizing command at the start of this textbox. You can also make your abstract horizontally skinnier by decreasing the number in the parbox qualifier.*

## INTRODUCTION

Algorithmic time complexity has interesting applications to Number Theory and much more. How long does it take to sort a list of a million numbers?

Can large primes be factored quickly? If the solution to a problem can be verified in polynomial time, can it be solved in polynomial time? [2].

If P=NP, then cryptographic security systems, as they stand now, are broken. It is as easy to decode encrypted messages as it to encode them.

Solving this problem is interesting and valuable because of its implications to data security as well as speeding up computers.

## 1. HISTORY

The P vs NP question is famous for being one of the most accessible millennium prize questions[5]. It began in 1936 when Alan Turning developed a theoretical computational model, which later became a useful theoretical model for computation. Although, the original model soon failed as the needs for time and memory were not accounted for[3].
Throughout the 1960's, 70's, and 80's the understanding of time complexity developed rapidly. Several mathematicians and computer scientists proved many problems could be solves in polynomial time and created notations that illustrated the time (and sometimes, space) complexity of their calculations. In the 1970's **NP**-completeness became one of the most insightful and fundamental theories in mathematics and furthered out knowledge of the combinatorial difficulties of many problems that seem to lack solutions that can be found in polynomial time[3].

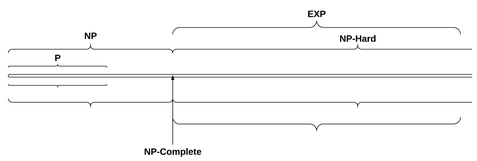
## 2. NOTATION

Some good rules for mathematical notation:

- P = Set of all problems which are solvable in polynomial time.
- NP = Set of all problems whose solutions can be verified in polynomial time.
- NP-Hard = Set of all problems that are at least as hard as the problems in NP.
- NP-Complete = Set of all problems that are in both NP and NP-Hard.

## 3. FIGURES

Blanks are bad for business. Fill your space with figures, if you have nothing else.



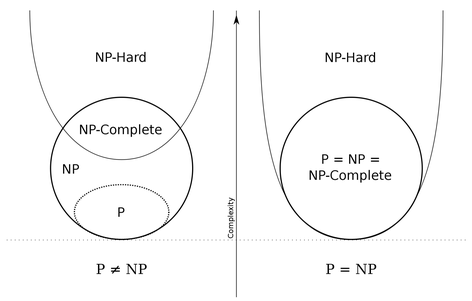




**Figure 1:** *Euler Diagram showing P vs NP vs NP-Hard*

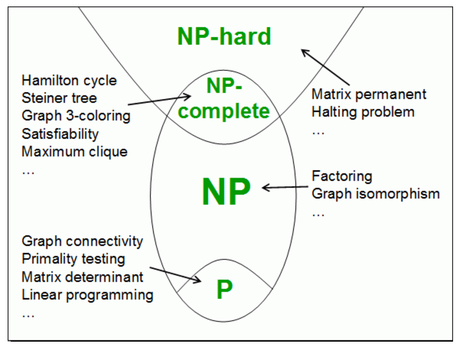



**Figure 2:** *P, NP, NP-Hard Problems*

Don't forget that you may need to scale your images so that they fit in the columns. An oversized image will wreak havoc on the appearance of your poster.

## 4. The Multicols Environment

The {4} after the multicols command tells the compiler to divide the width of the poster into four equal columns, and adjust the length of the poster as necessary. If you have less content or wide figures then you might be better off with fewer columns – three is fine, but 2 starts to make things look sparse.

The thing is, you need to have enough material to fill up the poster and prevent a giant white spot at the bottom. That's just awful-looking!

## 5. MORE SECTIONS!

The more we know about our degradation function H and our noise function $N$, the better our estimate of $f(x, y)$ will be. In the spatial domain, we represent our degraded function as:

$$g(x, y) = h(x, y) * f(x, y) + \eta(x, y) \tag{1}$$

where $h(x, y)$ is the spatial representation of the degradation function.

## 6. References with BibTeX

Open the references.bib file and check out the sample citation entries. If you require different kinds of entries, search "Bibtex entries" on the internet and a whole array of options will be available. Each entry has a name that you'll use in the tex file when you're typing to call up that citation.

Once your references are each formatted as a BibTeX entry, you can cite them using the cite{} command. in the tex tile. There are several examples here in this document. If you don't actually have a citation in the text for something, you can use the command "nocite" in front of the bibliography style command in your file.

You need to cite data sources and any specialized software that you've used (GAP, Sage, R). Usually specialized software programs have information on their webpages about how to properly cite the software.

Now you need to compile your tex file. First, make sure references.bib and your tex file are in the same folder. Then latex compile your tex source file. If you're using TeXShop or TeXWorks, then the next step is to Bibtex compile your tex source file. Then latex compile your tex source file TWICE more. It's crazy! If you're using other software, sometimes you just need to latex compile your tex source file 3-4 times. When in doubt, the internet can help you fgure out the procedure.

## 7. ACKNOWLEDGEMENTS

Professor Elizabeth Wilcox, for her review and guidance throughout the research and development of this project

## References

[1] MS Windows NT kernel description. http://web.archive.org/web/20080207010024/http://www.808multimedia.com/winnt/kernel.htm. Accessed: 2010-09-30.

[2] A. Chermak and A. Delgado. A measuring argument for finite groups. *Proc. AMS*, 107(4):907 – 914, 1989.

[3] Lance Fortnow. The computational complexity column. NEC Labratories America.

[4] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.5*, 2014.

[5] Michael Sipser.