

P versus NP

Your Friend, P

P stands for "Polynomial time"

P is the set of problems that we can solve with a program in $O(n^c)$ time for a fixed c , where n = the length of the input

Addition, Multiplication, Stable Pairings
 $O(n)$ time $O(n^{1.58})$ time $O(n^3)$ time

Algorithm for Stable Pairings: $O(n^3)$ time

For each day that some boy gets rejected, do:

- **Morning** $< n^2$ days
 - Each girl stands on her balcony
 - Each boy proposes under the balcony of the best girl that remains on his list
- **Afternoon (for those girls with at least one suitor)**
 - To today's best suitor: "Maybe, come back tomorrow"
 - To any others: "No, I will never marry you"
- **Evening**
 - Any rejected boy crosses the rejecting girl off his list

If no boy was rejected: Each girl marries the last boy to whom she said "maybe"

Addition, Multiplication, and Stable Pairings

These problems are routinely solved in practice, all the time.

This is possible because algorithms for them are considered to be very efficient.

P is the set of problems that we can solve "efficiently" with a computer program

Addition, Multiplication, Stable Pairings
 $O(n)$ time $O(n^{1.58})$ time $O(n^3)$ time

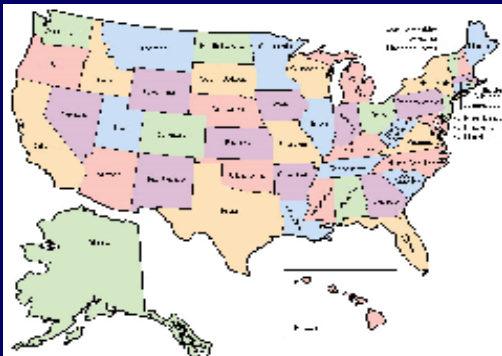
What is NP?

To answer this question, we will break out our coloring books.

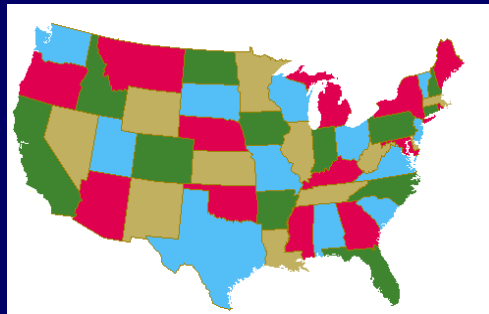
Proud to Color American Maps...



Proud to Color ...With Five Colors



Proud to Color ... With Four Colors

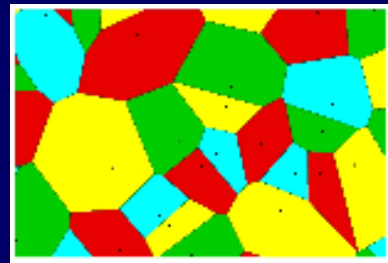


The Wild Wild West



Suppose Nevada is **B**.
Then the cycle **Oregon**→**Idaho**
→**Utah**→**Arizona**→**Cali**→**Oregon**
must use only **R** and **G**.
Suppose Oregon is **R**.
Then Idaho and Cali are both **G**,
Utah and Arizon both **R** ...
that doesn't work!
But if Oregon is **G**, the same kind
of reasoning holds.
Three colors are not enough!

The Four Color Theorem:

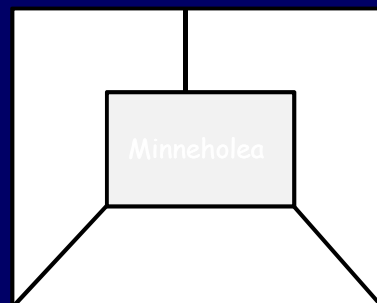


Every map can be colored with only four colors

The Four Color Theorem

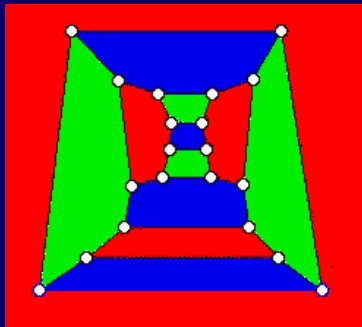
1852: Guthrie, coloring the counties of England, noticed that four colors sufficed.
1853: Kempe proved it
1864: Heawood found a bug in the proof...
1880: Tait proved it
1891: Peterson found a bug...
1922: Franklin proved it for 25 countries
1976: Appel and Haken "proved" it
Their proof is considered correct... but part of the proof uses a computer and cannot be verified by hand!
1996: Robertson, Sanders, Seymour, Thomas:
Somewhat shorter computer search, easier to check...
OPEN PROBLEM: IS THERE A SIMPLE PROOF??

When Can We Get Away With Three?



MAP OF THE KINGDOM OF DONUT-ISTAN

When Can We Get Away With Three?



COLORING PLANAR MAPS

FOUR COLOR THEOREM: EVERY MAP IS 4-COLORABLE

FACT: SOME MAPS ARE 3-COLORABLE, SOME ARE NOT

Let's suppose we represent an uncolored map as a black-and-white bitmap file.

THE 3-COLORING PROBLEM

Given an uncolored map M as input, print "yes" exactly when M can be colored with three colors.

THE 3-COLORING PROBLEM

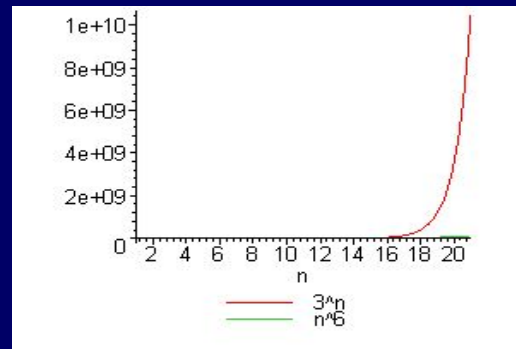
Given an uncolored map M as input, print "yes" exactly when M can be colored with 3 colors.

IS THIS PROBLEM COMPUTABLE?

YES! Solve by "Brute Force"

Every map M is a finite file of 0's and 1's.
There are a finite number of countries, C
For all 3^C ways to give a color to each country,
fill in those colors and check if that works.
If you ever find one that works, print "yes"!

But 3^n Grows Really Fast!



OK, But Maybe You Can Do Better?

Is the 3-Coloring Problem in **P**?

Well, maybe I don't know how to find a coloring that works, but if you guess a coloring for me, then I can tell you when **your coloring works!**



NP = "Neat Proofs"

Recall that **P** stands for "Polynomial Time"

P is the set of problems that can be solved by some program in $O(n^c)$ time for a fixed c , where n = the length of the input

Addition, Multiplication, Stable Pairings

$O(n)$ time $O(n^{1.58})$ time $O(n^3)$ time

NP = "Nondeterministic Polynomial Time"

NP is the set of problems where **any proposed solution can be "checked"** by some program in $O(n^c)$ time for a fixed c . The program gets not only the problem but a proposed solution to it.

NP = "Neat Proofs"

NP stands for "Nondeterministic Polynomial Time"
 NP is the set of problems that we can **check a possible solution to**, in $O(n^c)$ time for a fixed c

3-COLORING is in NP:

If I give you a map M and then color all the regions with 3 colors, you can *tell* if the coloring works or if it does not!
 Just find two regions that are adjacent but have the same color. You can eyeball this.

NP

All problems with efficient verification algorithms of given solutions

For every such problem, **finding** a solution (of bit length n) takes $\leq 2^n$ steps: try all possible solutions & verify each.

Can we do better than "brute force"?
 Do all NP problems have efficient algs?

P versus NP

P: Problems for which solutions can be efficiently *found*
NP: Problems for which solutions can be efficiently *checked*

Fact: $P \subseteq NP$ [finding implies checking]
Conjecture: $P \neq NP$ [finding is much harder than checking]

"P=NP?" is a central question of math, science, and technology!

What is in NP?

Mathematician: Given a statement, *find* a proof
Scientist: Given data on some phenomena, *find* a theory explaining it.
Engineer: Given constraints (size, weight, energy) *find* a design (bridge, medicine, phone)

In many intellectual challenges, *verifying* that we found a good solution is an easy task!
 (if not, we probably wouldn't start looking)

If $P=NP$, these all have an automatic *finder*

General Sudoku

		8	6								
										6	
			4	8					2	3	
	5			9							8
4	9					2	1				
2			4		7						
3	6			2	9						
1											
						5	1				

3

1		2	3	4		12	6			7	
	8			7		3		9	10	6	11
	12		10		1	13	11			14	
3		15	2		14		9			12	
13			8		10	12	2		1	15	
	11	7	6			16		15			5
			10	5	15		4	8			11
16		5	9	12		1					8
	2						12	5	8		3
13			15	3		14	8		16		
5	8		1		2			13	9	15	
		12	4	6	16	13	7				5
	3		12			6	4	11		16	
	7		16	5	14		1			2	
11	1	15	9			13				14	
	14			11	2			13	3	5	12

4

General Sudoku

y			b	a	c	x	n		h		t	r	i		d	e								
w	t	s	u	j	h	v		d	q	c		o	k	b	n	a	w	p						
w	h	e	m	a	n		l	u	k	p		r	y	s	x	d	q	c	o	j	l	b		
b		j		p	s		t				l	m	v	n	g	h	a	q		r	x	y		
x	o	i	d		i	p		r	e			f	u	j	w	y	m		h	s	c			
w	q	u					i	e	x	b	o	m	a		n	h	k	c	s					
n	c			w	x	u	s	f	q		l		e	m	k	v				j	a			
a	l	x	f	c	l		m	v	k	w	q			j	d	g	b	h						
s		b	v		h	k	p	o	b	u	f	j	n		q	t		j	d	i	m	r	q	
		b	d		m	r	v				j	h	p		a	g	y	w			t	u		
y	p		e	t	a	m		v	h	o	b	x		l	t	s	q	u	w	g	r	c	d	k
q	g	j		e	s	r	h	c						f	k		x		y	l	a	o		
	u	t	k		n	o		l	r	m	q	y		b	a	v	j			l	p	h		
x	r	w	p		y	k	l		l	e	j			o	m		t	q	v	u				
s	n	b	q	c		g	w	k	a	u	t	p	y		o	r	x		j	m				
j	n	s	q	v	x	y	h		u	t	p	o	g	i	m		f	d		w	l	k	r	
u	w	b	t		l	e	r	p	o	m		c	d	f	k	v				s	q		e	
			h		m	s	c	f	q	j		k	n	g	w	b			l	v	u	e		
			o	e	d	l	k	n	q		a	w	u		j	a	l		h	b	p	m		
l	k			v	j	t	w		a	s	h					u	r	q	c	d	f	n		
			g		d	y	r	w		c	l		i	n	p	v	a	f	e		q			
l	v	x	p		t	b		d	n	r		w			g		s	a	h	y	l			
		l	k	w	c	g	q	x	h		d	n	r		a	u	i	d	e	s		m	f	v
		a	y	r		d	f	e	n	x	k	s	h		b	u	p							
q	i		r	s		m	i	v		w				h	x	t	y							

5

Universality: NP-completeness

3-Coloring solvers can solve any NP problem

Cook-Levin '71: NP-complete problems **exist!**

SAT is NP-complete. We can encode any NP problem as a simple logic problem.

Karp '72: NP-complete problems **abound!**

21 problems in logic, optimization, algebra,...

Today: >3000 problems in all sciences, **equivalent**

Yato '03: **General Sudoku** is NP-complete

$P=NP$ iff **Sudoku** has an efficient algorithm

Universality: NP-completeness

NP-complete problems:

If one of them is easy, then all are!

If one of them is hard, then all are!

3-Coloring,

Sudoku,

Minesweeper,

And thousands more problems

NP-complete

NP-complete

NP-complete

