

From DocDel <docdel@Princeton.EDU>

Sent Friday, January 6, 2006 10:30 am

To fineanx@Princeton.EDU

Subject Document Delivery Pull Slip

THIS ITEM WAS REQUESTED BY: Engineering on 1/6/2006 10:27:36 AM

Please scan and finish this article for Article Express located at Annex B. Thank you.

ILLiad Transaction Number: 244678

This request is for: Sanjeev Arora (arora)

Delivery Method: Hold for Pickup

Electronic Delivery: Yes

Call Number: Oversize 9000.846q

Branch Location: Annex B

Journal Title: Scientific American

Article Author: HR Lewis and CH Papadimitriou

Article Title: Efficiency of Algorithms

Journal Vol: 238 Journal Issue: 1

Journal Month: Journal Year: 1978

Article Pages: 96--? 110

This request has been sent by ss - Borrowing.

ILLiad Transaction Number: 244678

Thank you,

Shauna Shaw

Engineering Library

The Efficiency of Algorithms

Some mathematical problems can be solved only by methods too slow for even the fastest computers. More efficient methods have not been found, but neither has it been proved that there are no better methods

by Harry R. Lewis and Christos H. Papadimitriou

Suppose you were asked to plan an itinerary for a traveling salesman who must visit a number of cities. You are given a map on which the distances between the cities are marked and you are asked to find the shortest route that passes through all the cities and returns to the starting point. An approach to this problem that is certain to give the correct answer is to trace all the possible routes, measure their length and pick the shortest one. If the tour included more than a few cities, however, hundreds or thousands of routes would have to be checked. If there were 100 cities, then even the fastest computers would require weeks of calculation to find the shortest path.

In the search for a quicker solution you might try some less rigorous methods. One idea that seems reasonable is always to visit nearby cities before going on to those farther away. You would soon discover, however, that this procedure does not invariably yield the correct answer. Other shortcuts also fail. In fact, the best methods known for solving the problem are not much better than the obvious but laborious procedure of checking all the possible itineraries. Mathematicians now suspect that this problem and many others like it may forever remain beyond our ability to solve in any efficient way. That speculation itself, however, is unconfirmed; although no faster methods of solution have been found, neither has it been proved that faster methods do not exist.

In the problem of the traveling salesman's tour it is not the solution for a particular set of cities that is of the greatest importance but a general method for finding the solution for any cities. Such a method is called an algorithm; it is a precisely stated procedure or set of instructions that can be applied in the same way to all instances of a problem. If the problem is to be solved with the aid of a computer, an algorithm is indispensable, because only those procedures that can be stated in the explicit and unambiguous form of an algorithm

can be presented to a computer. Instructions that are vague or that rely on intuition are unacceptable.

An example of an algorithm is the procedure taught in the schools for the subtraction of whole numbers. If each of the steps in this procedure is applied correctly one at a time, the algorithm will always yield the correct result. What is more, once the algorithm has been learned or stored in the memory of a computer or embodied in the circuitry of an electronic calculator, it can be applied to an infinite set of subtraction problems. With this one algorithm the difference between any two whole numbers can be determined.

In principle any problem for which an algorithm can be devised can be solved mechanically. It may therefore seem surprising that there are problems for which algorithms exist but for which we so far have no practical general solution. The algorithms for solving these problems always give a correct answer, but they often require an inordinate amount of time. The problem of the traveling salesman's tour is among these intractable tasks.

The efficiency of computer algorithms is a topic of obvious practical importance. It is also of interest in more formal areas of mathematics. There are some problems in mathematics and logic for which no algorithm can ever be written, and there are many others for which efficient, fast algorithms are already known. Between these two groups is a third class of problems that can always be solved in principle but for which there are only inefficient (and therefore largely unusable) algorithms. For some of these difficult problems mathematicians have been able to demonstrate that efficient algorithms can never be designed. For many of the most important problems, however, there is only the suspicion that good algorithms are impossible.

A given problem can have more than one algorithm for its solution. For example, children in Europe learn a procedure for subtraction slightly different

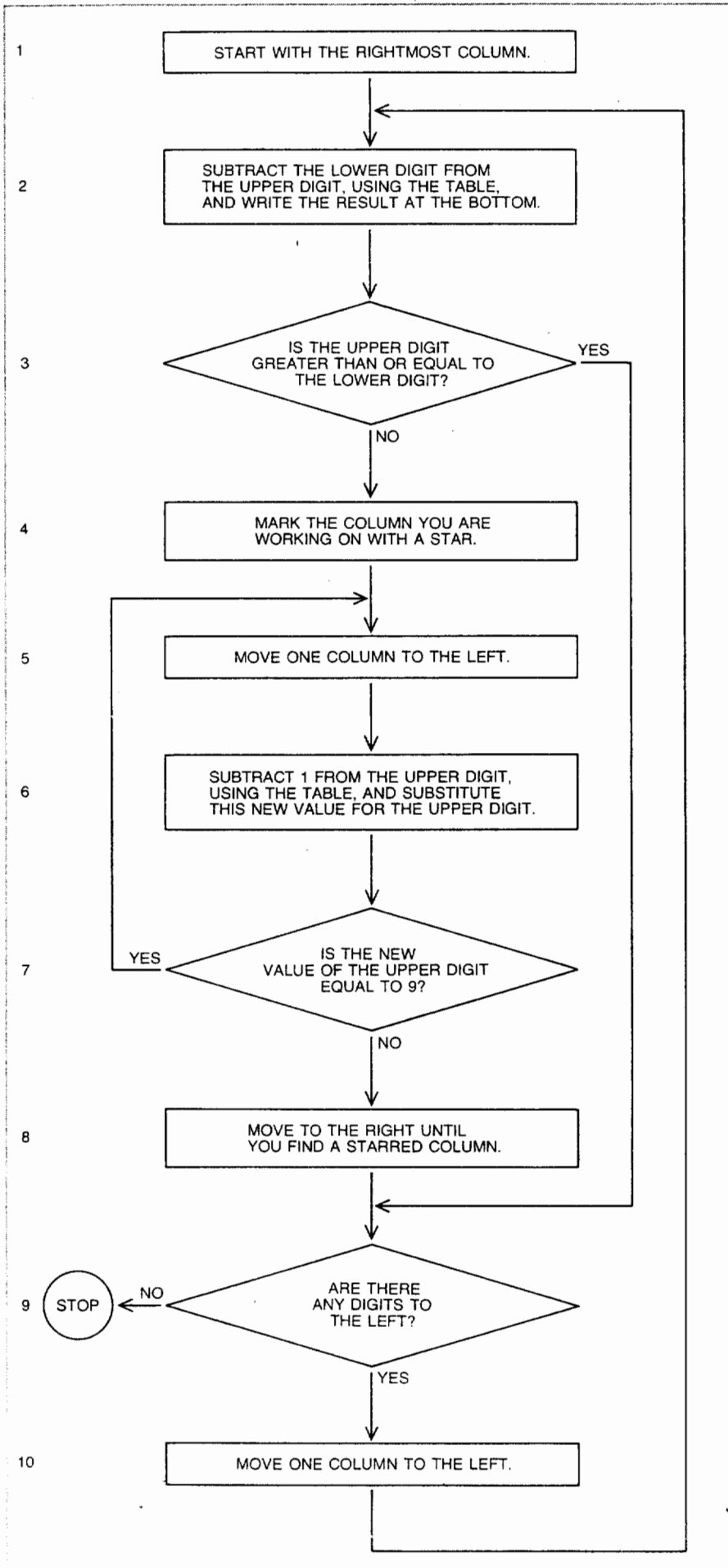
from the one taught in the U.S. Both of the subtraction algorithms, however, give the same result in the same amount of time. That is not invariably the case with different algorithms for solving the same problem. One celebrated problem that can be solved by either a "fast" algorithm or a "slow" one is the problem of the Königsberg bridges.

In the 18th-century German city of Königsberg (which is now the Russian city of Kaliningrad) a park was built on the banks of the river Pregel and on two islands in the river. Within the park seven bridges connected the islands and the riverbanks. A popular puzzle of the time asked if it was possible to walk through the park by a route that crossed each of the bridges once and only once.

For the solution of the problem the size and shape of the islands and the length of the bridges are immaterial; the only essential information is the pattern of interconnections. This information can be presented compactly in the mathematical structure known as a graph, which is merely a set of points with lines drawn to join them. In the case of the Königsberg park each of the riverbanks and each of the islands is condensed to a single point and each of the bridges is represented by a line between two points. Thus the graph consists of four points and seven lines. If the lines are labeled, any path through the park can be specified by a simple listing of labels.

The obvious approach to the problem is to list all the paths that cross all the bridges and to eliminate from consideration those that cross any bridge more than once. This is the technique of exhaustive search, similar to the one employed in the problem of the traveling salesman. When the mathematician Leonhard Euler was presented with the problem of the Königsberg bridges, he recognized the limitations of the technique and found another method. In recognition of his contribution a path that traverses each line of a graph exactly once is now called an Eulerian path.

Euler wrote: "The particular problem of the seven bridges of Königsberg



that meets these conditions actually has an Eulerian path requires a somewhat more complicated argument, which we shall not present here, but Euler was able to give a rigorous proof.

It is an easy matter to express Euler's solution to this problem in an algorithm that could be executed by a computer. The first requirement, connectivity, can be established by marking some point of the graph, then similarly marking all points connected to it by lines, then all the points connected to the newly marked points, and so on. The graph is connected if at the end all points have been marked. The second requirement is just as easily tested: the machine is instructed to examine each point of the graph and count the number of lines that terminate at that point. If no more than two of the points have an odd number of lines, the graph has an Eulerian path. The park at Königsberg met the first condition but not the second, and so there was no Eulerian tour of the seven bridges.

Euler's method is unquestionably the more economical approach to the problem of the Königsberg bridges: it requires that each point and line of the graph be listed just once, whereas the exhaustive search is not completed until every path that crosses all the bridges has been listed. The number of such paths is much larger than the number of points and lines in the graph. In that sense Euler's method is the better algorithm, but how much better? How can

		DIMINUEND									
		0	1	2	3	4	5	6	7	8	9
SUBTRAHEND	0	0	1	2	3	4	5	6	7	8	9
	1	9	0	1	2	3	4	5	6	7	8
	2	8	9	0	1	2	3	4	5	6	7
	3	7	8	9	0	1	2	3	4	5	6
	4	6	7	8	9	0	1	2	3	4	5
	5	5	6	7	8	9	0	1	2	3	4
	6	4	5	6	7	8	9	0	1	2	3
	7	3	4	5	6	7	8	9	0	1	2
	8	2	3	4	5	6	7	8	9	0	1
	9	1	2	3	4	5	6	7	8	9	0

2	*	3	*
3	0	4	7
-	1	4	3
	1	6	0
			9

DIMINUEND
 SUBTRAHEND
 DIFFERENCE

ALGORITHM for the subtraction of whole numbers defines an explicit procedure that can be followed without any need for intuition and even without an understanding of the significance each step has to the operation as a whole. The algorithm, which is assumed to incorporate the table of differences shown here, can be applied to an infinite number of subtraction problems. Other algorithms are equally effective. A method of subtraction taught in European schools, for example, differs in the treatment of borrowing where it specifies that 1 should be added to the lower digit instead of being subtracted from the upper one.

the d
one
Fo
seve
enot
pose
brid
othe
were
is be
new
list
c
path
sear
each
list
i
erati
resu
num
path
In
tion:
sis f
eval
effi
that
er at
at w
rith
to m
rith
slow
to b
mai
tial
A
ined
met
pon
ber
the
Gro
a r
wh
the
sim
Am
reac
tiple
n).
all
hav
gro

M
The
line
igna
ity.
lem
met
the
are
suc
nor
tha
F
exp
exc
the
The

the difference be measured, and how can one tell if the difference is significant?

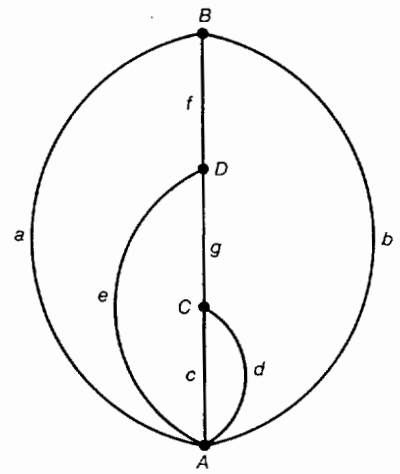
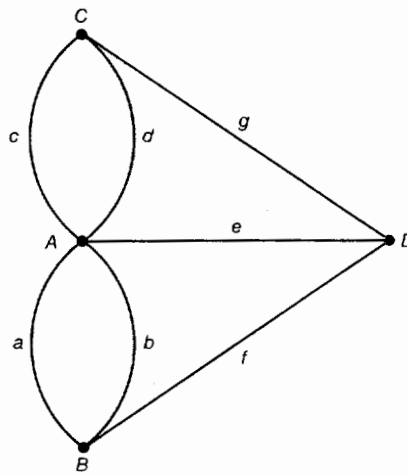
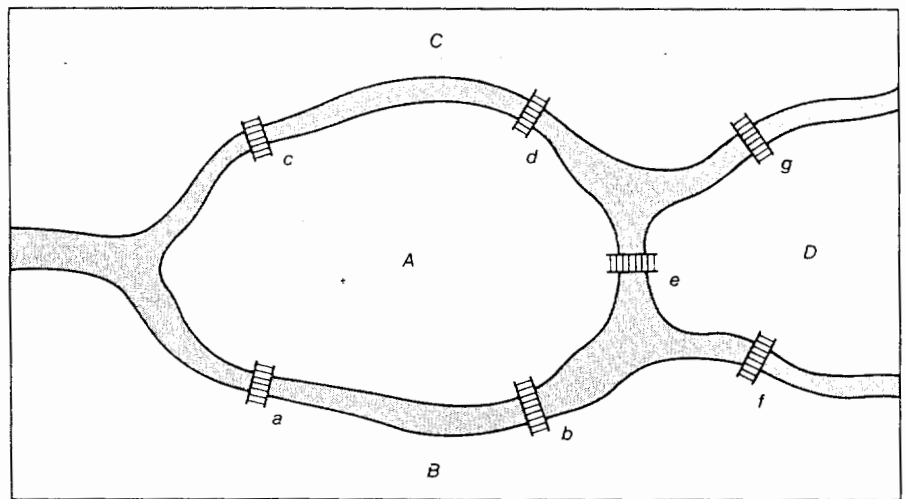
For a graph with only four points and seven lines both techniques are fast enough to be considered practical. Suppose, however, that more islands and bridges were added to the park, or in other words that more points and lines were added to the graph. If the problem is being solved by Euler's method, each new point merely adds one item to the list of points that must be checked. If the paths are to be examined by exhaustive search, on the other hand, then with each new point and line the size of the list is multiplied by some factor. A moderate increase in the size of the graph results in an explosive increase in the number of paths. Ultimately the list of paths must become prohibitively long.

In this comparison of the two solutions to Euler's problem there is the basis for a completely general method of evaluating the speed or practicality or efficiency of any algorithm. We imagine that the algorithm is supplied with larger and larger inputs, and we note the rate at which the execution time of the algorithm increases. In this way it is possible to make unequivocal judgments of algorithms. Exhaustive search not only is a slower method; in general it is too slow to be of any value. Euler's method remains practical for problems of essentially unlimited size.

As the size of the graphs being examined increases, the lists produced by the method of exhaustive search grow exponentially. Each time some fixed number of points and lines are added to the graph the size of the list doubles. Growth of this kind can be described by a mathematical function such as 2^n , where n is some measure of the size of the graph. Many other functions have similar or even higher rates of growth. Among them are n^n and $n!$ (which is read as " n factorial" and signifies n multiplied by all the integers between 1 and n). For the purposes of this discussion all these functions can be regarded as having the same property of exponential growth.

Mathematical functions of another kind are known as polynomials. The simplest members of this class are linear functions, such as $3n$, which designate a simple relation of proportionality. The time needed to solve the problem of the Königsberg bridges by Euler's method increases as a linear function of the size of the graph. Other polynomials are n^2 , n^3 and so on, and the sums of such functions. What distinguishes polynomials from exponential functions is that n never appears in an exponent.

For sufficiently large values of n any exponential function will overtake and exceed any polynomial function. It was the certainty of this result that dismayed Thomas Malthus when he compared the



abcdf	abcdg	abcgd	abcfg	abdcef	abdceg	abdgec	abdgf	A: abcde = 5
acdbfe	acdbfg	acdefb	acdeg	acgebf	acged	acgfbd	acgfbe	B: abf = 3
adceg	adgebfg	adgcb	adgfb	adgfb	adgfb	aefbcd	aefbcd	C: cdg = 3
aegdbf	aegdc	abegcd	abegdc	abef	adcbfe	adcbfg	adcefb	D: efg = 3
aefbdg	aegcbf	aegcd						

EULER'S PROBLEM asks whether there is a path through a graph that traverses each line exactly once. In this context a graph is defined as any collection of points with lines connecting them. The problem was first stated in terms of a walking tour through a park in the 18th-century German city of Königsberg (top); the path sought was required to cross each of the seven bridges in the park exactly once. The park can be represented as a graph in at least two equivalent ways. One approach to the problem is to list all the paths through the graph that continue as far as they can without repeating a line. Even for a small graph, however, there are many such paths; the sample listing shown includes only those paths that begin with line a . A more efficient algorithm was discovered by Leonhard Euler. A graph has a path that traverses each line once, he showed, if every point of the graph (with two possible exceptions) is at the junction of an even number of lines. Such a path is now called an Eulerian path. Counting the lines meeting at each point shows that the graph of the Königsberg park has no Eulerian path.

exponential rate of population growth with the polynomial rate of increase in the food supply. For small values of n a given polynomial function may well exceed a given exponential one, but there is always a value of n beyond which the exponential function is the greater. The exact form of the polynomial makes little difference, except in changing the point at which the polynomial function is overtaken.

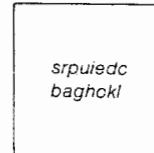
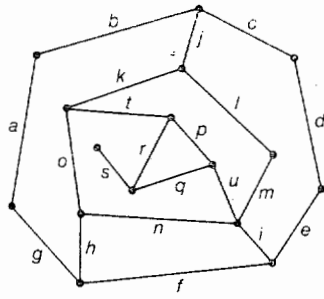
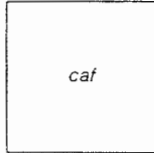
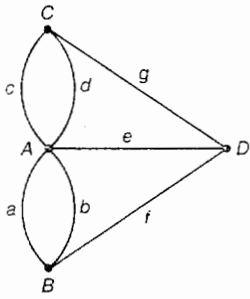
It is now generally agreed by computer scientists that algorithms whose execution time increases exponentially as a function of the size of the input are not

of practical value. We shall call algorithms of this kind "exponential time" algorithms, or simply inefficient algorithms. The only algorithms that are considered fast or efficient enough for general application are "polynomial time" algorithms.

Of course, even among efficient algorithms some are faster than others, but for the purposes of this discussion it is important only to distinguish polynomial-time algorithms as a class from exponential-time algorithms. Moreover, this system of classification has the advantage that it makes the speed of an al-

has what we was ler's thm ater. can point king then ewiy ph is have ment ne is f the than er of path. first l so even the the s: it the the until lges such er of that lgo-can

hole that tion sig- as a in- ere, ub- ual- ght s in ifies in- one.



4,294,967,297

6700417
641
6700417
26801668
40202502
4294967297

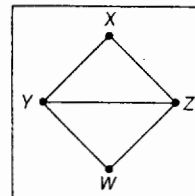
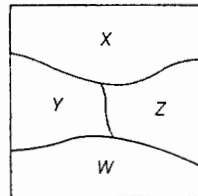
PROBLEMS IN THE CLASS NP ask a yes-or-no question that often can be answered only through a time-consuming, inefficient procedure, but when the answer is known to be yes, that fact can be demonstrated by a short proof. For the present there is no efficient way to determine whether a graph has a Hamiltonian path, but if it does have one, a brief "certificate" can be issued to prove it. The certificate simply lists the lines of the path in the order they are traversed. Another problem in the class NP asks whether a number is compos-

ite, that is, whether it can be written as the product of two other numbers. No efficient way of answering this question is known; indeed, in one case it took almost 100 years to show that a number is composite. In 1640 Pierre de Fermat proposed that $4,294,967,297$, which is equal to $2^{32} + 1$, is a prime number, and he was not proved wrong until Euler discovered the factors of the number in 1732. Once a number is known to be composite, however, that fact can be demonstrated by exhibiting a multiplication that gives the number as an answer.

designated by the letters NP, signifying "nondeterministic polynomial." The class NP encompasses all the problems in P, or in other words P is a subset of NP. In addition NP includes problems whose status is less certain. They are all solvable problems in principle; they have algorithms, but for now only exponential-time algorithms are known. They may also have polynomial-time algorithms (in which case NP and P are identical), or they may prove to be permanently intractable, with only inefficient solutions.

The problems considered here, and all problems classified in this way, can be described as an infinite set of similar questions each of which can be answered yes or no. For problems that are formally unsolvable, such as the problem of predicting whether a Turing machine will halt, these questions simply cannot be answered by any algorithmic procedure. For problems of the class P the questions can invariably be answered, whether the answer turns out to be yes or no, by an efficient procedure. In order for a problem to qualify for the

class NP there need not be an efficient means of answering the yes-or-no questions. What is required is that whenever the answer is yes there be a short and convincing argument proving it. Hamilton's problem, for example, meets this condition. It is not possible to tell by any efficient means known today whether a graph has a Hamiltonian path, but if it does, then the path itself can be exhibited. Hence for every Hamiltonian graph it is possible to issue a "certificate" that proves its membership in this special class of graphs. Such a



- ∧ AND
- ∨ OR
- ¬ NOT

$((X \text{ is red}) \vee (X \text{ is blue}) \vee (X \text{ is green})) \wedge ((Y \text{ is red}) \vee (Y \text{ is blue}) \vee (Y \text{ is green})) \wedge ((W \text{ is red}) \vee (W \text{ is blue}) \vee (W \text{ is green})) \wedge ((Z \text{ is red}) \vee (Z \text{ is blue}) \vee (Z \text{ is green})) \wedge$

Every country is colored some color.

$((\neg(X \text{ is red}) \vee \neg(X \text{ is blue}) \vee \neg(X \text{ is green})) \wedge ((\neg(Y \text{ is red}) \vee \neg(Y \text{ is blue}) \vee \neg(Y \text{ is green})) \wedge ((\neg(Z \text{ is red}) \vee \neg(Z \text{ is blue}) \vee \neg(Z \text{ is green})) \wedge ((\neg(W \text{ is red}) \vee \neg(W \text{ is blue}) \vee \neg(W \text{ is green})) \wedge$

No country is colored two colors.

$((\neg(X \text{ is red}) \vee \neg(Y \text{ is red})) \wedge ((\neg(X \text{ is blue}) \vee \neg(Y \text{ is blue})) \wedge ((\neg(X \text{ is green}) \vee \neg(Y \text{ is green})) \wedge$

X and Y are not the same color.

$((\neg(X \text{ is red}) \vee \neg(Z \text{ is red})) \wedge ((\neg(X \text{ is blue}) \vee \neg(Z \text{ is blue})) \wedge ((\neg(X \text{ is green}) \vee \neg(Z \text{ is green})) \wedge$

X and Z are not the same color.

$((\neg(Y \text{ is red}) \vee \neg(Z \text{ is red})) \wedge ((\neg(Y \text{ is blue}) \vee \neg(Z \text{ is blue})) \wedge ((\neg(Y \text{ is green}) \vee \neg(Z \text{ is green})) \wedge$

Y and Z are not the same color.

$((\neg(Y \text{ is red}) \vee \neg(W \text{ is red})) \wedge ((\neg(Y \text{ is blue}) \vee \neg(W \text{ is blue})) \wedge ((\neg(Y \text{ is green}) \vee \neg(W \text{ is green})) \wedge$

Y and W are not the same color.

$((\neg(W \text{ is red}) \vee \neg(Z \text{ is red})) \wedge ((\neg(W \text{ is blue}) \vee \neg(Z \text{ is blue})) \wedge ((\neg(W \text{ is green}) \vee \neg(Z \text{ is green})) \wedge$

W and Z are not the same color.

PROPOSITIONAL CALCULUS serves as a universal language for problems in the class NP. The problem considered here is that of applying three colors to a map or its equivalent graph. A sentence in the propositional calculus is made up of statements, such as "X is red," joined by the logical connectives "and," "or" and "not." If two statements are joined by a logical "and," the sentence is true only if both statements are true; a logical "or" requires that at least one of the statements be true, and "not" signifies that a statement is false. The sentence representing the map-coloring problem has three parts: the first two establish that every country has exactly one color and the

third part lists the countries that cannot have the same color. The map-coloring problem is thereby reduced to the problem of determining whether this sentence can be satisfied, that is, whether it is possible to assume some statements to be true and others false in such a way that there are no contradictions. Since all problems in NP could be expressed as sentences in the propositional calculus, an efficient general method for solving the satisfiability problem could be applied to all those problems. No such method is known. The sentence given here can be satisfied by assuming, for example, that only the statements "X is red," "Y is blue," "Z is green" and "W is red" are true.

