How hard can it be?

Deliver the solution to an everyday puzzle and you could win the biggest prize in mathematics, says **Ian Stewart**

EVER since a Babylonian scribe decided to teach his students arithmetic by setting them problems using the formula "I found a stone but did not weigh it..." mathematicians have celebrated the hidden depths of apparently everyday problems. They have found inspiration in slicing pies, tying knots and spinning coins. But even mathematicians have been surprised by the depth of the mystery that lurks behind an innocent question about postage stamps.

Suppose that your post office sells stamps with just two values: 2 cents and 5 cents. By combining these values, you can make up almost any whole number of cents. For example, to post a letter costing 9¢, you could stick one 5¢ stamp and two 2¢ stamps on the envelope. Two values that you cannot achieve are 1¢ and 3¢ – and in fact these are the only impossible amounts. You can produce any even amount using 2¢ stamps – given a big enough envelope – and any odd value from 5¢ upwards, using one 5¢ stamp and multiple 2¢ stamps.

This example is typical. Given an unlimited supply of stamps, there is always some key value above which any total can be achieved by sticking the right combination of stamps on the envelope. This is also true if you have more than two denominations of stamp available.

But the million-dollar question is this: with n denominations of stamps available, what is that key value? The first person to consider a simple version of this question was James Joseph Sylvester in 1883 (to be precise, he was dealing with coins, but for our purposes we'll stick with stamps). Sylvester came up with a simple formula for finding this key value when dealing with just two denominations (see "Pushing the envelope", page 48).

In its general form, the postage-stamp problem really could be a million-dollar question: the Clay Mathematics Institute in Cambridge, Massachusetts, is offering exactly that amount to anyone who can solve a problem that is logically equivalent to it. We now have tantalising new hints that the postage-stamp problem – and therefore, perhaps, the related million-dollar enigma – might not be as daunting as it appears. So considering how to pay for posting our mail might lead to a breakthrough in one of the most significant mathematical problems of the 21st century.

It will never compute

The issue centres around the cost of solving a problem – not in dollars and cents, but in computational effort. We measure the difficulty of a calculation by the number of basic computational steps needed to complete it: for a particular size of problem – often measured in terms of the number of digits in the number to be crunched – what is the "running time" of the algorithm concerned? If the problem concerns 50-digit numbers rather than 25-digit numbers, say, how much longer does the algorithm take to get the answer? What about 100-digit numbers, or any number of digits? It should be noted that this running time is an abstract notion, related but not equivalent to the actual time taken by any given computer.

In broad terms, a computational method is practical – "efficient" or "easy", if you prefer to look at it that way – if the running time grows in step with some fixed power of the number of digits required to pose the question. For example, an algorithm for testing a number n to see whether it is prime may have a running time linked to the sixth power of the number of digits of n.

Such algorithms are said to be "class P", where the "P" stands for "polynomial". Algorithms that run in polynomial time are relatively stable: they do not get wildly slower with small increases in the size of the input. In contrast, non-P algorithms are generally impractical – "inefficient" or "hard" – and become unmanageable with relatively small increases in input size. It's not quite that straightforward, because some non-P algorithms are pretty efficient until the input size gets very big indeed, while some P algorithms depend on a parameter which is so large that they couldn't actually run within a human lifetime. Nevertheless, the distinction between P and non-P seems to be the most basic and important distinction in problems about the efficiency of algorithms – a way to formalise the intuitive ideas of "easy to compute" versus "hard to compute".

Are there any such things as truly hard problems? Yes, several kinds. The obvious ones are hard for a simple reason, such as printing out the answer takes too long. A good example is "print all ways to rearrange this list of symbols". With the 52 symbols in a pack of cards, the list would contain 80,658,175,170,943,878,571,660,636,856,403,766,975,289,505,440,883,277,824,000,000,000,000 arrangements, and you'd have to print the lot. These types of problem have to be excluded, which we do by introducing another class of algorithm, confusingly called NP, which run in "nondeterministic polynomial" time. A problem is NP if any proposed solution can be checked to determine whether it is right or wrong, in polynomial time – that is, in reasonable time. A rough analogy is solving a jigsaw puzzle. However long it takes to work out how to fit the pieces together – the nondeterministic aspect – a brief glance at the result usually reveals whether it is correct.

All this classification has led to a rather fundamental question, and whoever cracks it will take the Clay prize: is NP really any different from P? To put it plainly, if it is easy to check the accuracy of any proposed ▶

solution to a problem, must there be an easy way to solve the problem in the first place?

The smart money says NP problems need not be P: even if it is easy to check any proposed solution to a problem, you can't solve that problem efficiently by making repeated guesses and checking them in turn, because the sheer number of possibilities is too large. Think of opening a combination lock by trying every combination in turn. A single satisfying "click" greets the correct answer, but if you are dealing with a sophisticated lock you could spend a lifetime trying successive combinations. Guessing at a computer password is another example.

Even without the Clay prize as motivation, most mathematicians would sell their mothers into slavery to find out whether NP is distinct from P because it is such a baffling and fundamental problem. The truly tantalising thing about this conundrum is that it is an example of an "NP-complete" problem. NP-complete problems are a subset of NP problems and are special in that if an efficient solution to any of them can be found, then that same solution can be used to solve any NP problem efficiently. In other words, finding an efficient way to solve any NP-complete problem means we have shown that all NP problems are effectively P. The matter isn't entirely esoteric either: the problem is related to some issues in banking security, for example, so there is good reason to pursue a solution. Which brings us back to our very own problem – postage stamps.

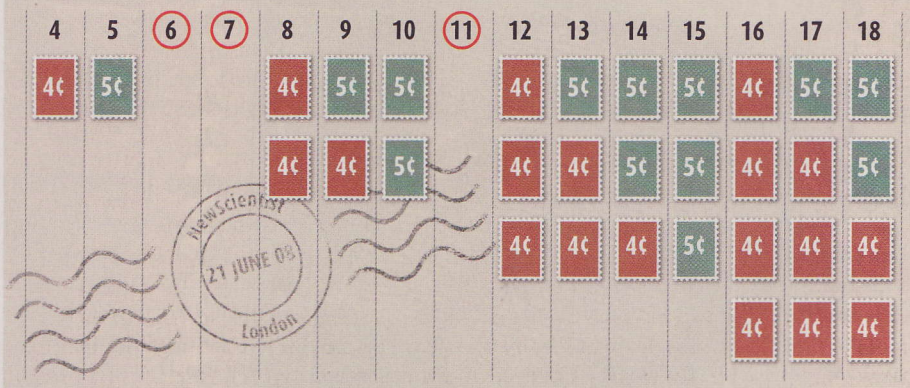
It was Jorge Ramirez-Alfonsin of the Pierre and Marie Curie University in Paris, France, who proved in 1996 that the general version of the stamp problem – with an unlimited number of stamp denominations – is NP-complete. So if there is an efficient method to solve the postage-stamp problem, there is an efficient method to solve any NP problem.

Well, is there an efficient method? Not yet, but there is a new reason for optimism, at least for a simplified version of the problem in which there is a limit on the number of stamp denominations allowed – three, four, 10, 100, whatever – though with such a limit in place, however large, the problem is no longer NP-complete. What's more, simple formulae for the postage-stamp problem of the type that Sylvester found no longer apply. But more complex algorithms do and in principle they run in polynomial time, so if we consider such specific examples of our NP-complete problem, they become class P.

As the number of stamp denominations increases, so will the power of the input size that determines the running time. Suppose, for instance, that for three stamps you could find an algorithm whose running time is proportional to the third power, or cube, of the total number of digits in the stamps' values; for four stamps you could find an algorithm whose running time is proportional to the fourth power of the number of digits and so on. In general, for n denominations of stamp you could find an algorithm whose running time is proportional to the nth power of the number of digits. This is good news: in this scenario, broadly speaking, each separate 3-stamp, 4-stamp or 1000-stamp problem would be class P. Indeed, Ravi Kannan, now at Yale University, showed in 1992 that there is an "efficient" solution for each number of stamps.

It is not all good news, however. Ramirez-Alfonsin's theorem shows that we cannot

Pushing the envelope



Suppose that you have two types of stamp: 4¢ and 5¢. Which totals can you achieve, and what is the biggest total you cannot obtain? Clearly you can get:

4
5
9 = 4+5
10 = 5+5
12 = 4+4+4
13 = 4+4+5
14 = 4+5+5
15 = 5+5+5
16 = 4+4+4+4
17 = 4+4+5+4

and so on. The numbers 1, 2, 3, 6, 7 and 11 are impossible, but as soon as we can achieve four consecutive whole numbers (12, 13, 14, 15) then we can add extra 4s to them to get 16, 17, 18 and 19, then 20, 21, 22 and 23, and so on – with no gaps. So every value from 12 upwards can be achieved. The largest impossible total is thus 11 (see Diagram).

Now, 11 is equal to $(4 \times 5) - 4 - 5$. This pattern is universal. Given stamps of denominations x and y, with no common factor (other than of course 1), the largest total that cannot be achieved is exactly $xy - x - y$. So, for instance, if the two values are 99¢ and 101¢, then you can get every total larger than, but not equal to, $(99 \times 101) - 99 - 101$, which is 9799¢.

"Even if the general method your bank uses to encrypt account information is secure, the implementation might not be"

combine these individual methods to obtain one algorithm that is efficient no matter how many stamps we have. The whole lot lumped together into the general *n*-stamp problem would not be class P, because there is no upper limit to the powers that could occur. Furthermore, Kannan's theory turns out to be one of those cases where the theoretical concept of efficiency does not equate to "practical": gigantic exponents or multipliers undo much of his good work. As a result, no one has even considered using his method to program a computer to solve the problem.

The latest twist in this tale is more encouraging, though. Stan Wagon of Macalester College in St Paul, Minnesota, and colleagues can now solve 4-stamp problems that involve 100-digit numbers in about 1 second on a fast desktop computer, and 10-stamp problems with 10-digit numbers in two days. Better still, Bjarke Roune at the University of Aarhus, Denmark, has used computational algebraic geometry to solve 4-stamp problems with 10,000-digit values in a few seconds, and 13-stamp problems with 10-digit values in a few days.

While all of this leaves the general *n*-stamp problem unperturbed, it proves that an NP-complete problem can be solved in many practical situations. Imagine if one "nasty" combination of stamps in every trillion possibilities would take forever to solve, but that all the others take only a few minutes. The probability of encountering one of the nasty cases in any particular instance would be negligible. In this scenario, although the generalised problem is NP-complete and unavoidably hard, most specific examples of it could be much easier with the right approach. We already know that something like this happens for the travelling salesman problem – which aims to calculate the shortest return journey through *n* cities, visiting each city only once – and for some problems in mathematical economics. Now we're seeing it again for postage stamps. The same might occur in other NP-complete problems.

There are practical implications to all of this. Even if the method that your bank uses to encrypt your account information is NP-complete "in general" – which is more than can currently be proved for most practical encryption systems – the particular version that your bank is using might nevertheless be insecure. That is, perhaps, not a good enough reason to rush off to check your bank statement, but it could make us rethink the meaning of secure encryption.

The most exciting possibilities thrown up by the latest approaches to the postage-stamp problem renew hope for solving many problems that previously seemed unassailable. By excluding the rare worst-case scenarios and focusing our attention on the typical ones, we might lick them yet. ●

Ian Stewart's latest book is *Why Beauty Is Truth* (Basic Books)