ISSN 1754-3657 (Print)
ISSN 1754-3665 (Online)

CS4FN

Computer Science for Fun

Issue 14

The genius who gave us the future

100 years of Alan Turing

Build a computer out of chocolate

Everyday secret codes

The icebreaker t-shirt

The Alan Turing issue

This year, 2012, marks one hundred years since the birth of Alan Turing. You may not have heard of him before, but he is one of the most important scientists of the last century. He worked on maths, logic, code-breaking and most importantly, he came up with some of the fundamental ideas that make computers work. He was one of the very first computer scientists.

In this issue we'll explore Turing's world-changing life and ideas, and we'll check out the latest research in subjects he cared about. You'll read about computers made from chocolate, the best ways to keep a secret and an animal that can survive being chopped into almost 300 pieces. It's a pretty amazing world out there, made all the more amazing by Alan Turing's work.

Alan Turing's life

Alan Turing was born in London on 23 June 1912. His parents were both from successful, well-to-do families, which in the early part of the 20th century in England meant that his childhood was pretty stuffy. He didn't see his parents much, wasn't encouraged to be creative, and certainly wasn't encouraged in his interest in science. But even early in his life, science was what he loved to do. He kept up his interest in science while he was away at boarding school, even though his teachers thought it was beneath well-bred students. When he was 16 he met a boy called Christopher Morcom who was also very interested in science. Christopher became Alan's best friend, and probably his first big crush. When Christopher died suddenly a couple of years later, Alan partly helped deal with his grief with science, by studying whether the mind was made of matter, and where – if anywhere – the mind went when someone died.

The Turing machine

After he finished school, Alan went to the University of Cambridge to study mathematics, which brought him even closer to questions about logic and calculation. After he graduated he stayed at Cambridge as a fellow, and started working on a problem that had been giving mathematicians headaches: whether it was possible to determine in advance if a particular mathematical proposition was provable. Alan solved it (the answer was no), but it was the way he solved it that helped change the world. He imagined a machine that could move symbols around on a paper tape to calculate answers. It would be like a mind, said Alan, only mechanical. You could give it a set of instructions to follow, the machine would move the symbols around and you would have your answer. This imaginary machine came to be called a Turing machine, and it forms the basis of how modern computers work. See page 10 for more on Turing machines, and how to build one out of chocolates.

Code-breaking at Bletchley Park

By the time the Second World War came round, Alan was a successful mathematician who'd spent time working with the greatest minds in his field. The British government needed mathematicians to help them crack German codes so they could read their secret communiqués. Alan had been helping the government on and off already, but when war broke out he moved to the British code-breaking headquarters at Bletchley Park to work full-time. Based on work by Polish mathematicians, he helped crack one of the Germans' most baffling codes, called the Enigma, by designing a machine (based on an earlier Polish version) that could help break Enigma messages as long as you could guess a small bit of the text (see 'Cribs' on the next page). With the help of British intelligence that guesswork was possible, so Alan and his team began regularly deciphering messages from ships and U-boats. As the war went on the codes got harder, but Alan and his colleagues at Bletchley designed even more impressive machines. They brought in telephone engineers to help marry Alan's ideas about logic and statistics with electronic circuitry. That combination was about to produce the modern world.

Building a brain

The problem was that the engineers and code-breakers were still having to make a new machine for every job they wanted it to do. But Alan still had his idea for the Turing machine, which could do any calculation as long as you gave it different instructions. By the end of the war Alan was ready to have a go at building a Turing machine in real life. If it all went to plan, it would be the first modern electronic computer, but Alan thought of it as "building a brain". Others were interested in building a brain, though, and soon there were teams elsewhere in the UK and the USA in the race too. Eventually a group in Manchester made Alan's ideas a reality.

Troubled times

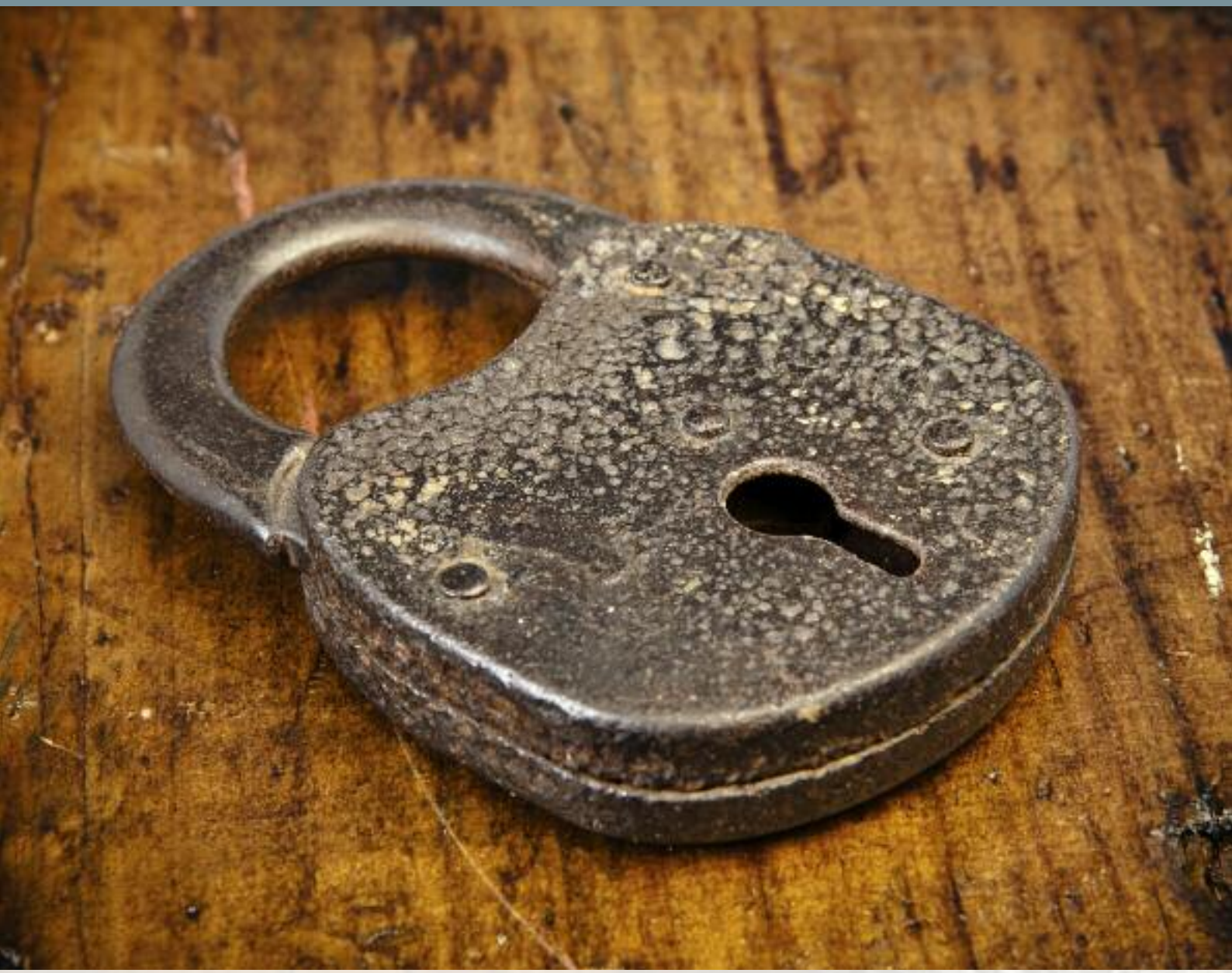
Not long after, Alan went to work at Manchester himself. He started thinking about new and different questions, like whether machines could be intelligent, and how plants and animals get their shape. But before he had much of a chance to explore these interests, Alan was arrested. In the 1950s, gay sex was illegal in the UK, and the police had discovered Alan's relationship with a man. Alan didn't hide his sexuality from his friends, and at his trial Alan never denied that he had relationships with men. He simply said that he didn't see what was wrong with it. He was convicted, and forced to take hormone injections for a year as a form of chemical castration.

Although he had had a very rough period in his life, he kept living as well as possible, becoming closer to his friends, going on holiday and continuing his work in biology and physics. Then, in June 1954, his cleaner found him dead in his bed, with a half-eaten, cyanide-laced apple beside him.

Alan's suicide was a tragic, unjust end to a life that made so much of the future possible. In this issue we're going to celebrate his life and his work, and we'll explore a few amazing and fun scientific discoveries that Alan never got a chance to see.

Cribs

The guessed bits of text used to crack the German codes were called 'cribs'. These were often mundane bits of text that were so frequently used that the codebreakers could guess them. Weather reports were good sources of cribs – the Brits have always been obsessed with the weather. "Nothing to report" turned out to be a useful crib. Who would have thought that reporting nothing could give away the war! One of the most ironic cribs used was perhaps "Heil Hitler" which was added to messages as a matter of course. Dictators can sometimes be too self-important for their own good.



Keeping secrets on the Internet

How do modern codes keep your data safe online? **Ben Stephenson** of the University of Calgary explains.

When Alan Turing was breaking codes, the world was a pretty dangerous place. Turing's work helped uncover secrets about air raids, submarine locations and desert attacks. Daily life in Europe might be safer now, but there are still threats out there. You've probably heard about the dangers that lurk online – scams, identity theft, viruses and malware, among many others. Shady characters want to know your secrets, and we need ways of keeping them safe and secure to make the Internet work. How is it possible that a network with so many threats can also be used to securely communicate a credit card number, allowing you to buy everything from songs to holidays online?

The relay race on the Internet

When data travels over the Internet it is passed from computer to computer, much like a baton is passed from runner to runner in a relay race. In a relay race, you know who the other runners will be. The runners train together as a team, and they trust each other. On the Internet, you really don't know much about the computers that will be handling your data. Some may be owned by companies that you trust, but others may be owned by companies you have never heard of. Would you trust your credit card number to a company that you didn't even know existed?

The way we solve this problem is by using encryption to disguise the data with a code. Encrypting data makes it meaningless to others, so it is safe to transfer the data over the Internet. You can think of it as though each message is locked in a chest with a combination lock. If you don't have the combination you can't read the message. While any computer between us and the merchant can still view or copy what we send, they won't be able to gain access to our credit card number because it is hidden by the encryption. But the company receiving the data still needs to decrypt it and open the lock. How can we give them a way to do it without risking the whole secret?
If we have to send them the code a spy might intercept it and take a copy.

One-way keys

The solution to our problem is to use a relatively new encryption technique known as public key cryptography. (It's actually about 40 years old, but as the history of encryption goes back thousands of years, a technique that's only as old as Victoria Beckham counts as new!) With this technique the code used to encrypt the message (lock the chest) is not able to decrypt it (unlock it). Similarly, the key used to decrypt the message is not able to encrypt it. This may sound a little bit odd. Most of the time when we think about locking a physical object like a door, we use the same key to lock it that we will use to unlock it later. Encryption techniques have also followed this pattern for centuries, with the same key used to encrypt and decrypt the data. However, we don't always use the same key for encrypting (locking) and decrypting (unlocking) doors. Some doors can be locked by simply closing them, and then they are later unlocked with a key, access card, or numeric code. Trying to shut the door a second time won't open it, and similarly, using the key or access code a second time won't shut it. With our chest, the person we want to communicate with can send us a lock only they know the code for. We can encrypt by snapping the lock shut, but we don't know the code to open it. Only the person who sent it can do that.

We can use a similar concept to secure electronic communications. Anyone that wants to communicate something securely creates two keys. The keys will be selected so that one can only be used for encryption (the lock), and the other can only be used for decryption (the code that opens it). The encryption key will be made publicly available – anyone that asks for it can have one of our locks. However, the decryption key will remain private, which means we don't tell anyone the code to our lock. We will have our own public encryption key and private decryption key, and the merchant will have their own set of keys too. We use one of their locks, not ours, to send a message to them.

Turning a code into real stuff

So how do we use this technique to buy stuff? Let's say you want to buy a book. You begin by requesting the merchant's encryption key. The merchant is happy to give it to you since the encryption key isn't a secret. Once you have it, you use it to encrypt your credit card number. Then you send the encrypted version of your credit card number to the merchant. Other computers listening in might know the merchant's public encryption key, but this key won't help them decrypt your credit card number. To do that they would need the private decryption key, which is only known to the merchant. Once your encrypted credit card number arrives at the merchant, they use the private key to decrypt it, and then charge you for the goods that you are purchasing. The merchant can then securely send a confirmation back to you by encrypting it with your public encryption key. A few days later your book turns up in the post.

This encryption technique is used many millions of times every day. You have probably used it yourself without knowing it – it is built into web browsers. You may not imagine that there are huts full of codebreakers out there, like Alan Turing seventy years ago, trying to crack the codes in your browser. But hackers do try to break in. Keeping your browsing secure is a constant battle, and vulnerabilities have to be patched up quickly once they're discovered. You might not have to worry about air raids, but codes still play a big role behind the scenes in your daily life.

A big brain, big number trick

A Turing machine manipulates long sequences of 1s and 0s, made of anything from electronics to chocolates, to make calculations (see page 10). Let's do it bigger. Get a friend to set their phone to calculator mode, and then multiply together any ten single digits. This will create a really, really big number. The friend should keep this big number a secret so you have no idea what it is. But even though you don't know the big number, your big brain will be able to spot something missing.

Get your friend to read out nine digits of their number, in random order, and to hold one of those digits back. That's the number you have to figure out. Oh, and to make it more difficult, that number shouldn't be zero; it's too easy to predict nothing. Your friend reads out their numbers, and after a dramatic pause you correctly reveal the secret digit they have held back. Your brain's like a Turing machine, only clearly bigger and better, as it was able to crack the hidden digit code. Or is it all a trick? (Hint: yes it is.)

Find out how it's done in the magazine+ section of our website, **www.cs4fn.org.**

Hide and see

Encryption is about hiding information in plain sight. A problem with traditional text codes, though, is that it's easy to tell they contain information. If you found a note written in seemingly random letters, you would realise that it's probably a coded message, and set about trying to crack it. So a powerful way to prevent a message being read is to disguise the fact that there is any message at all. That's the idea behind Hide & See, a system designed by Jaakko Tuomivaara for hiding information right on your walls where everyone can see. In his system, information from your phone or an Internet feed gets secretly displayed as changes to the pictures on display in your house.

A picture on your wall of a fair-skinned, freckled woman slowly adds more freckles as your missed phone calls pile up. In a photo on your mantelpiece, a man's shoulder bag changes colour as a friend moves between locations in the real world. In a third picture, some cracks in dried mud are actually a graph of stock market prices. Having this information right in front of you means you could glance at it any time, but no one else would realise it's even there. To them you would just have good taste in photography.

In fact, Hide & See is such an elegant solution to hiding information that it has been displayed alongside art. It was recently on display in the Museum of Modern Art in New York. Hopefully soon it will make the leap to real life, and we can decorate our rooms with beautiful photos and useful secrets.

The icebreaker t-shirt

Alan Turing had a lot of talent to shout about, but actually he was pretty shy. Throughout his life he had a reputation for being a little bit awkward in social situations. On the other hand, he kept a small circle of friends around him, and when he had something to talk about he could rise above his shyness.

Confident people can go straight up and introduce themselves to someone they're interested in getting to know, but for shy people like Turing, this isn't so easy. Making friends is a gradual process of revealing information about yourself, but when we're feeling shy we don't want to expose ourselves to someone new too quickly or too much. Wouldn't it be nice to have some pointers or clues about the person you're talking to before starting a conversation, to know what to talk about or when to take the leap?

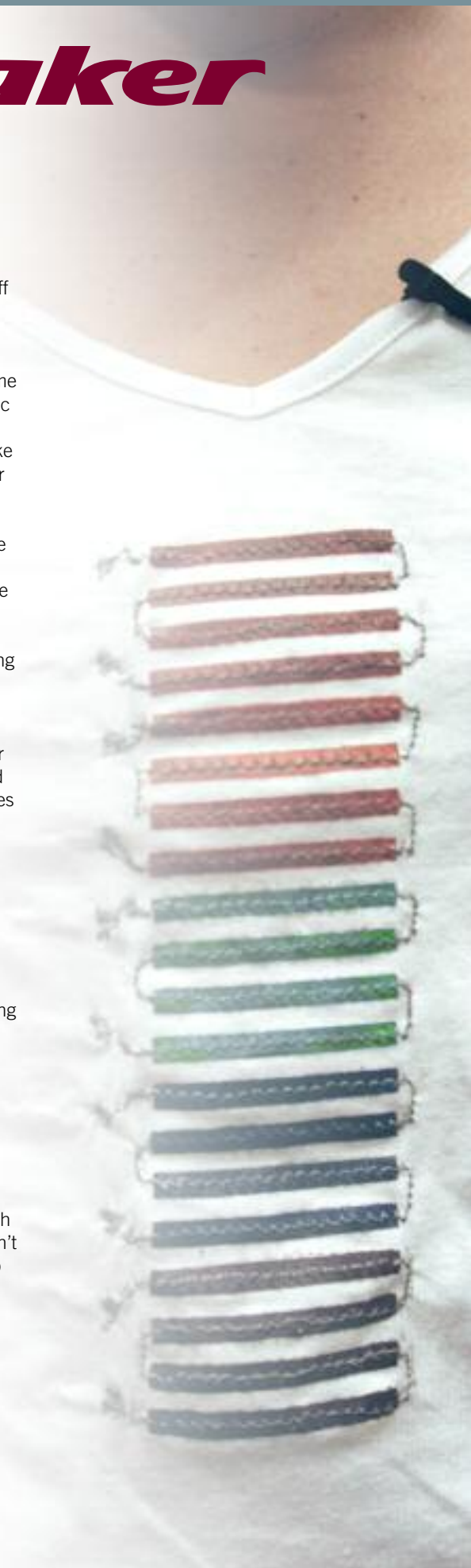
One of the advantages of the Internet is that it can help us to meet new people and connect with others with similar interests. But this doesn't necessarily lead to easier connections in the real world. Nanda Khaorapapong, a PhD student on the Media and Arts Technology programme at Queen Mary, University of London, has been looking at ways to bring some of the benefits of the online world to the real world, and help make these uncertain moments easier.

Designing a computer interface to fit into an everyday situation like meeting someone new poses some interesting problems. The technology has to fit naturally into a situation and it has to be comfortable to use. It must be helpful but not too overwhelming, or else it could take away from the interaction between two people, rather than add to it. After all, she's trying to make face-to-face contact less awkward and more comfortable for shy people. It seemed like the best way to put all of these ideas together – comfort, helpfulness and friendliness – would be to put the technology in a t-shirt. Her first prototype, the icebreaker t-shirt, shows two people wearing them how compatible they are after they shake hands with each other.

Nanda began by taking an RFID tag and reader and embeddeding them at one cuff in a long-sleeved t-shirt. A RFID tag is a computer chip with a radio antenna attached, for sending information when it passes near a reader. They're used in some modern passports, and on London's public transport system in pay-as-you-go cards. In Nanda's t-shirt, when two wearers shake hands, the RFID tags can read each other and recognise each other's identity. Next, a mini processor embedded in the shirt calculates compatibility levels between the wearers by matching their favourite film, music, hobbies and other interests. (In the prototype this is programmed in from a questionnaire, but in future it could be calculated on the fly from social networking sites like Facebook.) These compatibility levels are then shown on the shirt front using a display made of heat-sensitive paints and conductive thread. A black bar chart changes when it is heated up based on the compatibility. A colder match comes out blue or green, but for hotter matches between wearers the bars turn yellow, orange or red.

Lots of wearable computers use rigid displays, but Nanda's soft display means that the t-shirts are still soft and comfy. It's also designed not to give away too much exact information – the colour-coding gives shy users more room to fit the technology to the situation, so they can reveal themselves to others gradually. When Nanda tested her t-shirt with real people doing speed dating, 76% of shy testers said the shirt was helpful when meeting strangers. One said: "It gave a sense of having something in common to begin the conversation, something we both were interested in. Whereas when we didn't have the shirt, I felt we were only trying to make small talk."

If you're occasionally gripped with that awkward feeling around new people, take heart. You're certainly not alone. In fact, you've got some pretty good company in Alan Turing. And hopefully one day, when you meet someone new and interesting you'll be able to get clues about what to say from some helpful t-shirt technology.



Doctor You

When the Doctor in Doctor Who knows his time is up – usually because he's been injured so badly that he's dying – he can regenerate. He transforms into a completely different body. He ends up with a new personality, new looks, even new teeth.

Your body is constantly regenerating itself too. New cells are born to replace the ones that die. Your hair, nails and skin are always growing and renewing. Every year, you lose and regain so much that you could make a pile of dead cells that would weigh the same as your body. And yet with all this change, every morning you look in the mirror and you look and feel the same. No new personality, no new teeth. How does the human body keep such incredible control?

Here's another puzzler. Even though our cells are always being renewed, you can't regrow your arm if it gets cut off. We know it's not impossible to regrow body parts: we do it for small things like cells, and some animals like lizards can regrow tails. Why can we regrow some things but not others?

Creation of the shape

All of those questions are part of a field in biology called morphogenesis. The word is from Greek, and means 'creation of the shape'. Scientists who study morphogenesis are interested in how cells come together to create bodies. It might sound a long way from computing, but Alan Turing became interested in morphogenesis towards the end of his life. He was interested in finding out about patterns in nature – and patterns were something he knew a lot about as a mathematician. A paper he wrote in 1951 described a way that Turing thought animals could form patterns like stripes and spots on their bodies and in their fur (see 'Lines in the sand' on page 17).

Up for the chop

Turing died before he could do much work on morphogenesis, but lots of other scientists have taken up the mantle. One of them is Alejandro Sánchez Alvarado, who works at the Stowers Institute for Medical Research in Kansas City, in the USA. He is trying to get to the bottom of questions like how we regenerate our bodies. He thinks that some of the clues could come from working on flatworms that can regenerate almost any part of their body. A particular flatworm, called *Schmidtea mediterranea,* can regenerate its head and its reproductive organs. You can chop its body into almost 280 pieces and it will still regenerate.

A genetic mystery

The funny thing is, flatworms and humans aren't as different as you might think. They have about the same number of genes as us, even though we're so much bigger and seemingly more complicated. Even their genes and ours are mostly the same. All animals share a lot of the same, ancient genetic material. The difference seems to come from what we do with it. The good news there is that as the genes are mostly the same, if scientists can figure out how flatworm morphogenesis works, there's a good chance that it will tell us something about humans too.

One gene does it all

Alejandro Sánchez Alvarado recently did an experiment on flatworms where he cut off their heads and watched them regenerate. He found that the process looked pretty similar to watching organs like lungs and kidneys grow in humans as well as other animals. He also found that there was a particular gene that, when knocked out, takes away the flatworm's ability to regenerate.

What's more, he tried again in other flatworms that can't normally regenerate whole body parts – just cells, like us. Knocking out that gene made their organs, well, fall apart. That meant that the organs that fell apart would ordinarily have been kept together by regrowing cells, and that the same gene that allows for cell renewal in some flatworms takes care of regrowing whole bodies, Doctor-style, in others. Phew. A lot of jobs for one gene.

Who knows, maybe Time Lords and humans share that same gene too. They're like the lucky, regenerating flatworms and we're the ones who are only just keeping things together. But if it's any consolation, at least we know that our bodies are constantly working hard to keep us renewed. We still regenerate, just in a slightly less spectacular way.

Chocoholic Turing machines

Could you make the most powerful computer ever created ... out of chocolates? It's actually quite easy. You just have to have enough chocolates (and some lollies). It is one of computer science's most important achievements.

Imagine you are in a sweet factory. Think big – think Charlie and the Chocolate Factory. A long table stretches off into the distance as far as you can see. On the table is a long line of chocolates. Some are milk chocolate, some dark chocolate. You stand in front of the table looking at the very last chocolate (and drooling). You can eat the chocolates in this factory, but only if you follow the rules of the day. (There are always rules!)

Without rules there is only chaos. That is not good with chocolate at stake.

The chocolate eating rules of the day tell you when you can move up and down the table and when you can eat the chocolate in front of you. Whenever you eat a chocolate you have to replace it with another from a bag that is refilled as needed (presumably by Oompa-Loompas).

You also hold a single lolly. Its colour tells you what to do (as dictated by the rules of the day, of course). For example, the rules might say holding an orange one means you move left, whereas a red one means you move right. Sometimes the rules will also tell you to swap the lolly for a new one.

The rules of the day have to have a particular form. They first require you to note what lolly you are holding. You then check the chocolate on the table in front of you, eat it and replace it with a new one. You pick up a lolly of the colour you are told. You finally move left, move right or finish completely. A typical rule might be:

If you hold an orange lolly and a dark chocolate is on the table in front of you, then eat the chocolate and replace it with a milk one. Swap the lolly for a pink one. Finally, move one place to the left.

A shorthand for this might be:

if ORANGE, DARK then MILK, PINK, LEFT.

You wouldn't just have one instruction like this to follow but a whole collection with one for each situation you could possibly be in. With three colours of lollies, for example, there are six possible situations to account for: three for each of the two types of chocolate.

As you follow the rules you gradually change the pattern of chocolates on the table. The trick to making this useful is to make up a code that gives different

patterns of chocolates different meanings. For example, a series of five dark chocolates surrounded by milk ones might represent the number 5.

See the box for a set of rules that subtracts numbers for you as a result of shovelling chocolates into your face.

Our chocolate machine is actually a computer as powerful as any that could possibly exist. The only catch is that you must have an infinitely long table!

By powerful we don't mean fast, but just that it can compute anything that any other computer could. By setting out the table with different patterns at the start, it turns out you can compute anything that it is possible to compute, just by eating chocolates and following the rules. The rules themselves are the machine's program.

This is one of the most famous results in computer science. We've described a chocoholic's version of what is known as a Turing machine because Alan Turing came up with the idea. The computer is the combination of the table, chocolates and lollies. The rules of the day are its program, the table of chocolates is its memory, and the lollies are what is known as its 'control state'. When you eat chocolate following the rules, you are executing the program.

Sadly Turing's version didn't use chocolates – his genius only went so far! His machine had 1s and 0s on a tape instead of chocolates on a table. He also had symbols instead of lollies. The idea is the same though. The most amazing thing was that Alan Turing worked out that this machine was as powerful as computers could be before any actual computer existed. It was a mathematical thought experiment.

So, next time you are scoffing chocolates at random, remember that you could have been doing some useful computation at the same time as making yourself sick.

Chocoholic subtraction

A Turing machine can be used to do any computation, as long as you get its program right. Let's create a program to do something simple to see how to do it. Our program will subtract two numbers.

The first thing we need to do is to choose a code for what the patterns of chocolates mean. To encode the two numbers we want to subtract we will use sequences of dark chocolates separated by milk chocolates, one sequence for each number. The more dark chocolates before the next milk chocolate the higher the number will be. For example if we started with the pattern laid out as below then it would mean we wanted to compute 4 – 3. Why? Because there is a group of four dark chocolates and then after some milk chocolates a group of three more.

M M M D D D D M M D D D M M M M ...

Here is a program that does the subtraction if you follow it when the pattern is laid out like that. It works for any two numbers where the first is the bigger. The answer is given by the final pattern. Try it yourself! Begin with a red lolly and follow the table below. Start at the **M** on the very left of the pattern above.

	CURRENT LOLLY	CURRENT CHOCOLATE		NEW CHOCOLATE	NEW LOLLY	MOVE
if	RED	MILK	then	MILK	RED	RIGHT
if	RED	DARK	then	DARK	ORANGE	RIGHT
if	ORANGE	MILK	then	MILK	PINK	RIGHT
if	ORANGE	DARK	then	DARK	ORANGE	RIGHT
if	PINK	MILK	then	MILK	PINK	RIGHT
if	PINK	DARK	then	MILK	GREEN	RIGHT
if	GREEN	MILK	then	MILK	VIOLET	LEFT
if	GREEN	DARK	then	DARK	BLUE	LEFT
if	BLUE	MILK	then	MILK	BLUE	LEFT
if	BLUE	DARK	then	MILK	PINK	RIGHT
if	VIOLET	MILK	then	MILK	VIOLET	LEFT
if	VIOLET	DARK	then	MILK	NONE	STOP

From the above starting pattern our subtraction program would leave a new pattern:

M M M D M M M M M M M M M M M ...

There is now just a single sequence of dark chocolates with only one chocolate in it. The answer is 1!

Try lining up some chocolates and following the instructions yourself to see how it works. It's also explained on the cs4fn website, **www.cs4fn.org**.

Meet the chatterbots

Sitting down and having a nice chat with a computer probably isn't something you do every day. You may never have done it. We mainly still think of it as being a dream for the future. But lots of work is being done to make it happen in the present, and the idea has roots that stretch far back into the past. It's a dream that goes back to Alan Turing, and then even a little further.

The imitation game

Back around 1950, Turing was thinking about whether computers could be intelligent. He had a problem though. Once you begin thinking about intelligence, you find it is a tricky thing to pin down. Intelligence is hard to define even in humans, never mind animals or computers. Turing started to wonder if he could ask his question about machine intelligence in a different way. He turned to a Victorian parlour game called the imitation game for inspiration.

The imitation game was played with large groups at parties, but focused on two people, a man and a woman. They would go into a different room to be asked questions by a referee. The woman had to answer truthfully. The man answered in any way he believed would convince everyone else he was really the woman. Their answers were then read out to the rest of the guests. The man won the game if he could convince everyone back in the party that he was really the woman.

Pretending to be human

Turing reckoned that he could use a similar test for intelligence in a machine. In Turing's version of the imitation game, instead of a man trying to convince everyone he's really a woman, a computer pretends to be a human. Everyone accepts the idea that it takes a certain basic intelligence to carry on a conversation. If a computer could carry on a conversation so well that talking to it was just like talking to a human, the computer must be intelligent.

When Turing published his imitation game idea, it helped launch the field of artificial intelligence (AI). Today, the field pulls together biologists, computer scientists and psychologists in a quest to understand and replicate intelligence. AI techniques have delivered some stunning results. People have designed computers that can beat the best human at chess, diagnose diseases, and invest in stocks more successfully than humans.

A chat with a chatterbot

But what about the dream of having a chat with a computer? That's still alive. Turing's idea, demonstrating computer intelligence by successfully faking human conversation, became known as the Turing test. Turing thought machines would pass his test before the 20th century was over, but the goal has proved more elusive than that. People have been making better conversational chat programs, called chatterbots, since the 1960s, but no one has yet made a program that can fool everyone into thinking it's a real human.

What's up, Doc

On the other hand, some chatterbots have done pretty well. One of the first and still one of the most famous chatterbots was created in 1968. It was called ELIZA. Its trick was imitating the sort of conversation you might have with a therapist. ELIZA didn't volunteer much knowledge itself, but tried to get the user to open up about what they were thinking. So the person might type "I don't feel well", and ELIZA would respond with "you say you don't feel well?"

In a normal social situation, that would be a frustrating response. But it's a therapist's job to get a patient to talk about themselves, so ELIZA could get away with it. For an early example of a chatterbot, ELIZA did pretty well, but after a few minutes of chatting users realised that ELIZA didn't really understand what they were saying.

Where have I heard this before?

One of the big problems in making a good chatterbot is coming up with sentences that sound realistic. That's why ELIZA tried to keep its sentences simple and non-committal. A much more recent chatterbot called Cleverbot uses another brilliantly simple solution: it doesn't try to make up sentences at all. It just stores all the phrases that it's ever heard, and chooses from them when it needs to say something. When a human types a phrase to say to Cleverbot, its program looks for a time in the past when it said something similar, then reuses whatever response the human gave at the time. Given that Cleverbot has had 65 million chats on the Internet since 1997, it's got a lot to choose from. And because its sentences were all originally entered by humans, Cleverbot can speak in slang or text speak. That can lead to strange conversations, though.

A member of our team at cs4fn had an online chat with Cleverbot, and found it pretty weird to have a computer tell him "I want 2 b called Silly Sally".

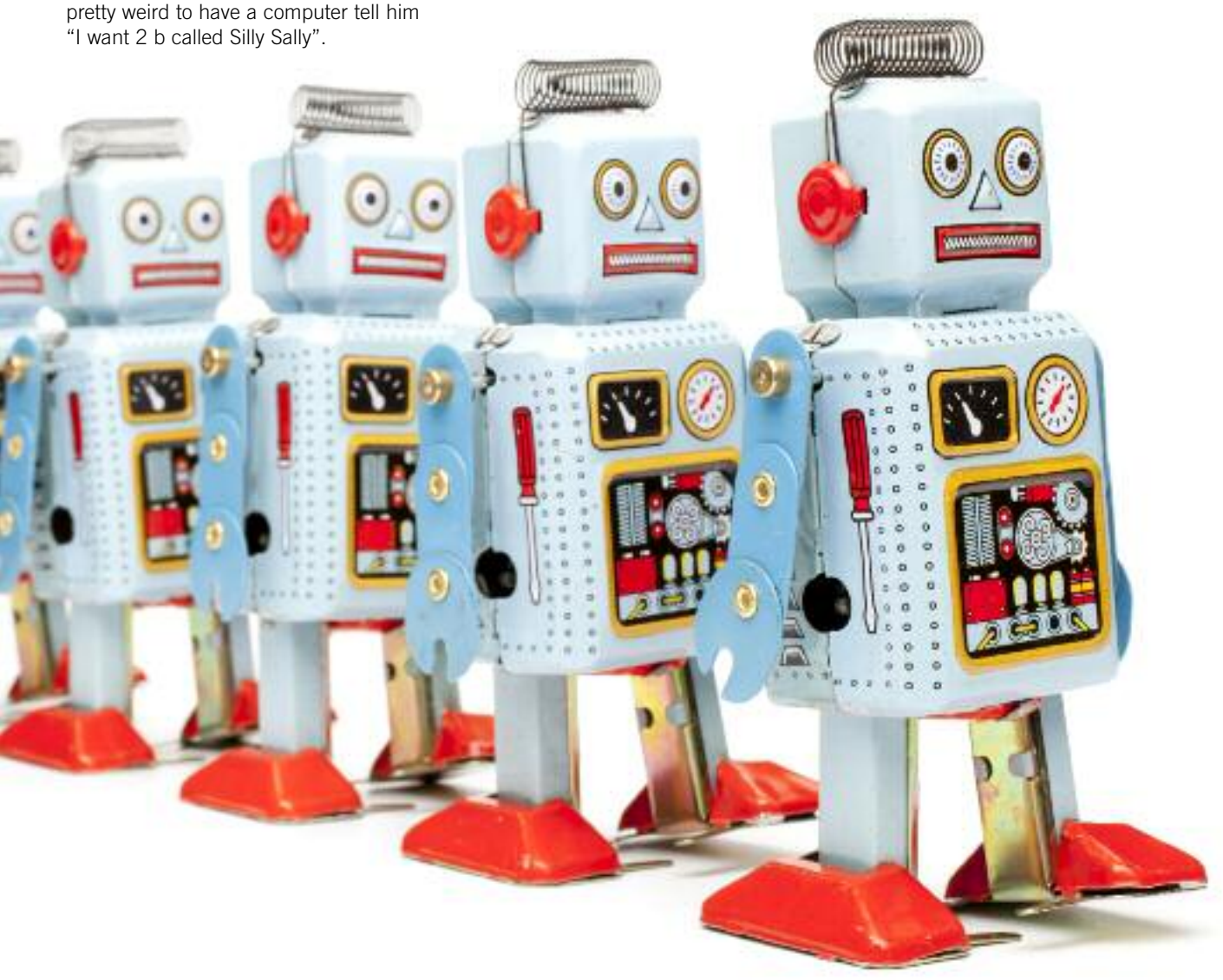
Computerised con artists

Most chatterbots are designed just for fun. But some chatterbots are made for a more sinister intent. A few years ago, a program called CyberLover was stalking dating chat forums on the Internet. It would strike up flirty conversations with people, then try and get them to reveal personal details, which could then be used to steal people's identities or credit card accounts. CyberLover even had different programmed personalities, from a more romantic flirter to a more aggressive one. Most people probably wouldn't be fooled by a robot come-on, but that's OK. CyberLover didn't mind rejection: it could start up ten relationships every half an hour.

Chatterbots may be ready to hit the big time soon. Apple's iPhone 4S includes Siri, a computerised assistant that can find answers to human questions – sometimes with a bit of attitude. Most of Siri's humourous answers appear to be pre-programmed, but some of them come from Siri's access to powerful search engines. Apple don't want to give away their secrets, so they're not saying much. But if computerised conversation continues advancing, we may not be too far off from a computer that can pass the Turing test. And while we're waiting at least we've got better games to play than the Victorians had.

Chat with a computer yourself

Try out chatterbots like ELIZA and Cleverbot! See our magazine+ page at **www.cs4fn.org** for links.



Britain in Alan Turing's time

Alan Turing changed the world, but the world shaped Alan Turing too. In the hundred years since Turing was born and the sixty years since he died, the world has changed so much that many of the things that shaped him seem quite foreign. **Edmund Robinson** of Queen Mary, University of London gives us some examples.

His family

Alan's mother was educated at universities in London and Paris, which was unusual at the time. She came from a distinguished Irish family that included academics and scientists – one of her cousins invented the word 'electron'! Her father became the chief engineer of the Madras Railway in India and was knighted. Alan's father's family were also distinguished: his ancestors were knights from Aberdeenshire in Scotland. Alan's grandfather was a clergyman, and his father was in the Indian civil service. Those were both jobs that were typical for the younger son in a family to do. Even the fact that Alan's parents both lived in India when they met is rooted in history: Alan was born when India was part of the British Empire.

His school

Most British parents in India still wanted their children to grow up in England, especially if their children could go to a good school. Alan's parents were no exception. Alan's school, Sherborne, was and is a good one. It's a big school in a small Dorset village. It was really good at producing Indian civil servants, but really bad at producing scientists. At the time, science wasn't really seen as a respectable job for the graduates of public schools. If you imagine Sherborne as Hogwarts, Turing was sort of like Neville Longbottom. He had a reputation for being a bit clumsy and unkempt. His chemistry experiments had a tendency to blow up. And you would never have thought that by the end of the story both Neville and Alan would have done some very heroic things including helping win a war each.

His love life

Alan was gay, but in those days gay sex was illegal. There were very few parts of British society where being gay was OK, and luckily when Alan went off to university he found one. Both the head of his college at Cambridge and its most famous fellow were openly gay or bisexual. Unfortunately, Alan still faced prejudice throughout his life. While he was living in Manchester in the 1950s the police found out that he was dating a man. He was convicted of indecency, lost his security clearance and was forced to take hormones to 'cure' him. Only in 2009 did the British government recognise the injustice of what Alan went through, and apologise.

His death

Alan died in 1954 as a result of eating a poisoned apple that he had laced with cyanide. Suicide was illegal at the time and considered to be an offence against God and the Queen. His mother was sure that he had just been sloppy with his chemicals. He may have chosen an ambiguous way to kill himself in part to spare her the social shame of the time. His choice may also have been due to a fairy tale. His favourite film was supposedly the Disney cartoon Snow White and the Seven Dwarfs and he especially liked the scene where the evil Queen dips the apple into a potion she has brewed to kill Snow White. Unfortunately there was no handsome prince to wake Alan though. His work changed all our lives. If only he had lived he undoubtedly would have gone on to change the world in other ways too.

Computers that read emotions

One of the ways that computers could be more like humans – and maybe pass the Turing test – is by responding to emotion. But how could a computer learn to read human emotions out of words? **Matthew Purver** of Queen Mary, University of London tells us how.

Have you ever thought about why you add emoticons to your text messages – symbols like :-) and :-@? Why do we do this with some messages but not with others? And why do we use different words, symbols and abbreviations in texts, Twitter messages, Facebook status updates and formal writing?

In face-to-face conversation, we get a lot of information from the way someone sounds, their facial expressions, and their gestures. In particular, this is the way we convey much of our emotional information – how happy or annoyed we're feeling about what we're saying. But when we're sending a written message, these audio-visual cues are lost – so we have to think of other ways to convey the same information. The ways we choose to do this depend on the space we have available, and on what we think other people will understand. If we're writing a book or an article, with lots of space and time available, we can use extra words to fully describe our point of view. But if we're writing an SMS message when we're short of time and the phone keypad takes time to use, or if we're writing on Twitter and only have 140 characters of space, then we need to think of other conventions. Humans are very good at this – we can invent and understand new symbols, words or abbreviations quite easily. If you hadn't seen the :-D symbol before, you can probably guess what it means – especially if you know something about the person texting you, and what you're talking about.

But computers are terrible at this. They're generally bad at guessing new things, and they're bad at understanding the way we naturally express ourselves. So if computers need to understand what people are writing to each other in short messages like on Twitter or Facebook, we have a problem. But this is something researchers would really like to do: for example, researchers in France, Germany and Ireland have all found that Twitter opinions can help predict election results, sometimes better than standard exit polls – and if we could accurately understand whether people are feeling happy or angry about a candidate when they tweet about them, we'd have a powerful tool for understanding popular

opinion. Similarly we could automatically find out whether people liked a new product when it was launched; and some research even suggests you could even predict the stock market. But how do we teach computers to understand emotional content, and learn to adapt to the new ways we express it?

One answer might be in a class of techniques called semi-supervised learning. By taking some example messages in which the authors have made the emotional content very clear (using emoticons, or specific conventions like Twitter's #fail or abbreviations like LOL), we can give ourselves a foundation to build on. A computer can learn the words and phrases that seem to be associated with these clear emotions, so it understands this limited set of messages. Then, by allowing it to find new data with the same words and phrases, it can learn new examples for itself. Eventually, it can learn new symbols or phrases if it sees them together with emotional patterns it already knows enough times to be confident, and then we're on our way towards an emotionally aware computer. However, we're still a fair way off getting it right all the time, every time.

See if you can catch the computer out by playing with the Chatterbox Wheel Of Emotion game. Look for a link in the magazine+ section of our website, **www.cs4fn.org**.

Lines in the sand

Towards the end of his life, Alan Turing got interested in patterns in nature. He wondered how it was that so many unusual shapes, like spots, stripes and spirals, could appear on plants and animals. He thought the answer might be in chemistry. A certain kind of reaction, in which two chemicals can transform into one another, and then spread out over a larger area, could end up producing elaborate patterns. Over time, scientists have found clues that Turing might have been right.

An inventor in New York City called Michael Dubno doesn't wait for nature to create patterns. He's made an artwork called the Sand Table that draws complicated patterns within itself. The glass-topped table contains a pan filled with sand and one steel ball bearing a little smaller than a clementine. Underneath the sand is a series of motors that move a powerful magnet around. When the magnet moves, the ball bearing moves through the sand and creates complicated patterns and drawings. The whole thing is controlled by a computer program. The user can tell the magnet to draw abstract shapes like spirals, snowflakes and mazes, make words or draw animals.

You can build your own sand table if you fancy a challenge! See the link in the magazine+ section of **www.cs4fn.org** to find out how.

Software for justice

The end of Alan Turing's life was marked by injustice, but the branch of science he founded is helping improve justice in our time. Here's how.

A jury is given misleading information in court by an expert witness. An innocent person goes to prison as a result. This shouldn't happen, but unfortunately it does and more often than you might hope. It's not because the experts or lawyers are trying to mislead but because of some tricky mathematics. Fortunately, a team of computer scientists at Queen Mary, University of London are leading the way in fixing the problem.

The Queen Mary team, led by Professor Norman Fenton, is trying to ensure that forensic evidence involving probability and statistics can be presented without making errors, even when the evidence is incredibly complex. Their solution is based on specialist software they have developed.

When a match may not be a match

Many cases in courts rely on evidence like DNA and fibre matching for proof. When police investigators find traces of this kind of evidence from the crime scene they try to link it to a suspect. But there is a lot of misunderstanding about what it means to find a match. Surprisingly, a DNA match between, say, a trace of blood found at the scene and blood taken from a suspect does not mean that the trace must have come from the suspect.

Forensic experts talk about a 'random match probability'. It is just the probability that the suspect's DNA matches the trace if it did not actually come from him or her. Even a one-in-a-billion random match probability does not prove it was the suspect's trace.

Worse, the random match probability an expert witness might give is often either wrong or misleading. This can be because it fails to take account of potential cross-contamination, which happens when samples of evidence accidentally get mixed together, or even when officers leave traces of their own DNA from handling the evidence. It can also be wrong due to mistakes in the way the evidence was collected or tested. Other problems arise if family members aren't explicitly ruled out, as that makes the random match probability much higher. When the forensic match is from fibre or glass, the random match probabilities are even more uncertain.

Probability problems

The potential to get the probabilities wrong isn't restricted to errors in the match statistics, either. Suppose the match probability is one in ten thousand. When the experts or lawyers present this evidence they often say things like: "The probability that the trace came from anybody other than the defendant is one in ten thousand." That statement sounds OK but it isn't true.

The problem is called the prosecutor fallacy. You can't actually conclude anything about the probability that the trace belonged to the defendant unless you know something about the number of potential suspects. Suppose this is the only evidence against the defendant and that the crime happened on an island where the defendant was one of a million adults who could have committed the crime. Then the random match probability of one in ten thousand actually means that about one hundred of those million adults match the trace. So the probability of innocence is ninety-nine out of a hundred! That's very different from the one in ten thousand probability implied by the statement given in court.

A map to justice

Norman Fenton's work is based around a theorem, called Bayes' theorem, which gives the correct way to calculate these kinds of probabilities. The theorem is over 250 years old but it is widely misunderstood and, in all but the simplest cases is very difficult to calculate properly. Most cases include many pieces of related evidence – including evidence about the accuracy of the testing processes. To keep everything straight, experts need to build a model called a Bayesian network. It's like a graph that maps out different possibilities and the chances that they are true. You can imagine that in almost any court case, this gets complicated awfully quickly. It is only in the last 20 years that researchers have discovered ways to perform the calculations for Bayesian networks, and written software to help them. What Norman and his team have done is develop methods specifically for modelling legal evidence as Bayesian networks in ways that are understandable by lawyers and expert witnesses.

Norman and his colleague Martin Neil have provided expert evidence (for lawyers) using these methods in several high-profile cases. Their methods help lawyers to determine the true value of any piece of evidence – individually or in combination. They also help show how to present probabilistic arguments properly.

Change that is slow in coming

Unfortunately, although scientists accept that Bayes's theorem is the only viable method for reasoning about probabilistic evidence, it's not often used in court, and is even a little controversial. Norman is leading an international group to help bring Bayes's theorem a little more love from lawyers, judges and forensic scientists. Although changes in legal practice happen very slowly (lawyers still wear powdered wigs, after all), hopefully in the future the difficult job of judging evidence will be made easier and fairer with the help of Bayes's theorem.

If that happens, then thanks to some 250 year-old maths combined with some very modern computer science, fewer innocent people will end up in jail. Given the innocent person in the dock could one day be you, you will probably agree that's a good thing.

Back (page) in the day

Alan Turing's contribution to computer science is legendary, but others have helped historically and geographically the world over to advance computing. Here are just a sample of some of them.

Made in Greece

Sponge divers, ancient loot and the Antikythera mechanism

In the early 1900s a group of sponge divers, taking a dip while a storm passed, discovered the wreck of an ancient ship on the seabed of Point Glyphadia on the Greek island of Antikythera. This historic discovery, believed to be a shipwreck from around 200 BC carrying the loot of a Roman general, turned out to contain computing treasures too. Along with the usual haul of statues of philosophers' heads and discus throwers, a small disc containing an intricate mechanism of at least 30 cogwheels was uncovered. Archaeologists believe the Antikythera mechanism, as it came to be known, was used by ancient Greeks to calculate the position of the moon and stars for a given date. It's also the oldest known hand-held computing device.

Motto: a computer in the hand owes much to this wreck

Made in Arabia

The cure for the common code

Ya'qūb ibn Ishāq al-Kindī (c. 801–873 AD), often known as 'the philosopher of the Arabs', was an outstanding mathematician and philosopher. Like Turing many centuries later, he was a code breaker. He is credited as being the first to develop the technique of frequency analysis to break secret codes. Al-Kindi realised that if he knew how often particular letters turned up normally in writing, he could use this to match particular symbols in the secret code. Common letters would occur as often in normal writing as in secret code, so he could crack the code by simply counting the number of times letters appeared in both. Al-Kindi was a renaissance man about 500 years before the renaissance: he also published numerous works on medicine and music.

Motto: clever codes can be frequently broken

Made in Scotland

Scotsmen, bones, and the end of the world

Born in Merchiston, Edinburgh, in 1550, John Napier was a bit of a problem at school. His nobleman father was told in a letter from his uncle, "I pray you, sir, to send John to the schools; over to France or Flanders, for he can learn no good at home." But when young John grew up he found learning really excited him. He became a world-class mathematician, astronomer, physicist and astrologer. Not only did he make the use of the decimal point commonplace, but he also created a new type of abacus, a mechanical computing device, based on his study of earlier Arabic mathematics. Napier's bones, as his abacus became known, allowed the rapid multiplication and division of numbers by moving around appropriately labelled wooden rods. The bones turned multiplication into simple addition, and division into subtraction, opening up a whole new world of applications. Like all of us, though, Napier didn't always get it right. As an astrologer his study of the biblical book of Revelation led him to believe that the end of the world would occur in 1688 or 1700.

Motto: always predict the end of the world to be after you're dead

Made in India

Inventing the importance of nothing

Computers live in a binary world of 1s and 0s, but where did 0 come from? We owe the big something that is nothing to the Indian mathematician and astronomer Brahmagupta (598–668 AD). Brahmagupta was the first person to use zero as a number: he invented nothing! He also founded the modern rule that two negative numbers multiplied together equals a positive number. Like other Indian scholars at the time, he wrote his books in elliptical verse, so his work was not only mathematical but poetic.

Motto: nothing can turn out to be a really big something

Made in America

A star computer, period

Henrietta Swan Leavitt worked as a 'computer' in 1893 at the Harvard Observatory. From around the mid-17th century the name computer referred to a person rather than a machine: someone who carried out mathematical calculations as their day job. Henrietta was employed to count the images of stars on astronomical photographs. While she counted she also thought, and her thoughts helped change the way we understand our universe. Henrietta noticed that some stars would appear, go away, and then come back again. Rather than being fixed points in the night sky, they varied in the amount of light they shone over a period of days, months or years. She had helped to discover a class of star called a Cepheid variable, stars that are used today to help us calculate the distances between galaxies.

Motto: computers need people too

cs4fn is supported by industry including **Google**, **Microsoft** and **ARM**.

The Chartered Institute for IT

EPSRC Engineering and Physical Sciences Research Council

COMPUTING AT SCHOOL

Queen Mary University of London

For a full list of our university partners see www.cs4fn.org

www.cs4fn.org