

# **CSAFN**

**Computer Science for Fun**

**Annual Issue 2**

## ***The women are here***



***“Computing’s too important  
to be left to men”***

***- Karen Spärk-Jones***

# ***Too important to be left to men***

Women have been at the forefront of computer science and electronic engineering from the outset. Does that surprise you? It shouldn't but if it does it's probably due in great part to the power of stereotypes. At the moment too many girls have been believing the stereotypes, which just leads to a self-fulfilling prophecy. Fortunately many ignore them and are continuing to be as successful as the women of history.

Here we celebrate some of the great women of the past, highlight the work of current top researchers and also profile some students who are set to continue the trend. Women may not always shout about their achievements, but boy do they do them well!

After all as leading computer scientist Karen Spärk-Jones said:  
*"Computing's too important to be left to men".*

## ***The woman of the future?***

Thomas Edison is a household name, a great man, a great thinker, an extraordinarily prolific inventor with over a 1000 patents to his name. He invented the light bulb, now the iconic image for 'bright idea'. Think of great inventors, great scientists and most people find it much easier to list men than women. Why is that? Try it. Before you read on, name some great female scientists or engineers?

There are lots but most find them hard to think of. Edison had a reason for that, as he described in an article in Good Housekeeping in 1912 called 'The Woman of the Future':

*"Direct thought is not at present an attribute of femininity. In this woman is now centuries, ages, even epochs behind man"*

This is of course total garbage. The trouble is men keep saying stupid things like that.

What, for example, was the female scientist Marie Curie doing around the time Edison was writing his ridiculous statement? Well actually it was the year before in 1911 that she had won the second of her Nobel prizes. She is of course one of the few people to have won two Nobel prizes and one of only two to win the prize in two different subjects. Not bad for someone "epochs behind men".

Whilst Edison was amassing his wealth and patents, Curie did not patent the radium-isolation process she invented. Who was the smarter? His aim was to be rich and famous. Hers was to allow scientific research to continue without the barriers patents put up. Who was the greater thinker? Perhaps women do things differently.

It is easy for famous people, even famous scientists, to make grand pronouncements that aren't based on science, which people then take more seriously than they should. The message of science, though, is clear:

Do **NOT** believe things just because of the person who says it. Look for the evidence. If there isn't enough to be sure, look for more.

So what of women and computer science. Most people would say it is a male-dominated subject. The truth is much more interesting. Women have played pivotal roles from the start. The first programmer was a woman: Ada Lovelace. Grace Hopper's work was central in the move from low-level programming languages to high-level ones. Fran Allen was awarded the 2006 Turing Prize, the most prestigious prize a computer scientist can win, for her work laying the foundation for modern optimising compilers. Great female thinkers are still in the thick of things as they have been throughout. Why shouldn't they? Computer Science requires great thinkers: innovative people who are creative, social team players, just like most other subjects. Both men and women can have those attributes.



## ***A gendered timeline of technology***

**825 Muslim scholar Al-Khwārizmī** kicks it all off with a book on algorithms – recipes on how to do computation pulling together work of Indian mathematicians. Of course back then it's people who do all the computation, as electronic computers won't exist for another millennium.

**1587 Mary, Queen of Scots** loses her head because the English Queen, Elizabeth I, has a crack team of spies that are better at computer science than Mary's are. They've read the Arab mathematician Al-Kindi's book on the science of cryptography so they can read all Mary's messages.

**1818 Mary Shelley** writes the first science fiction novel on artificial life, *Frankenstein*.

**1842 Ada Lovelace and Charles Babbage** work on the analytical engine. Lovelace shows that the machine could be programmed to calculate a series of numbers called Bernoulli numbers, if Babbage can just get the machine built. He can't. It's still Babbage who gets most of the credit for the next hundred-plus years (see page 9).

**1854 George Boole** publishes his work on a logical system that remains obscure until the 1930s, when Claude Shannon discovers that Boolean logic can be electrically applied to create digital circuits.

*Watch for more of our gendered timeline throughout this annual.*

# Using the dark side

Shadows are strange. They're everywhere but usually we don't notice them. They just sit there, moored to the thing that casts them, ignored by us as we go about our daily lives. These seemingly useless dark patches in our visual world move around with us, change colour with the weather, hide things, and move with the light. **Hannah Dee**, of the Grenoble Institute of Technology in the French Alps, tells us why she is bringing them into the limelight.

I'm a computer vision researcher, so I write programs that try to make sense of images and video. I'm really interested in shadows. I'm interested in how we can use shadows to help computers interpret images automatically, and how the human perceptual system uses shadows too.

## **The recipe for a shadow**

If you're going to teach computers how to use shadows, you've got to know why shadows look the way they do. A shadow has a simple recipe with just three ingredients – light, surface and object. When the object comes between the light and the surface the shadow is cast. The characteristics of those three ingredients determine how the shadow appears.

The sharpness of the edges of a shadow (the penumbra) depends upon how big the light source is: small lights cast sharp shadows, large lights cast fuzzy shadows. The shape of a shadow depends on the object that's casting it and the shape of the surface that it's cast upon. If there are any other light sources you might get extra shadows, but if the other light is really fuzzy and spread out you won't. However, the colour of the fuzzy light will determine the colour of the shadows. The indirect light reflected off the blue sky explains why shadows seem bluish on a sunny day.

## **The way you move**

That takes care of what a shadow looks like, but humans also get information from how shadows move. How a shadow moves depends on the movement of those three

shadow ingredients, light, surface, and object, as well. Mostly, we think of shadows moving when the object moves (shadow puppets, for example) and we can use these moving shadows to tell us about the moving object. Sundials are an example of using a moving light source (the sun) to get a shadow to tell us about something in the world.

## **What does a light source see?**


That's not all the shadow tells you! The position and shape of a shadow gives us clues about the location of the object and its relation to the surface the shadow is cast upon. This is because the shadow is the area hidden from the light by the object. (Leonardo da Vinci knew this back in the 15th century and said "No luminous body ever sees the shadows that it generates".) We can say that the outline of the shadow is the outline of the object from the point of view of the light. That is, if you were where the light was, the silhouette of the object would be exactly the shape of the shadow, but you'd not be able to see the shadow because the object would be exactly in the way.

Because the shadow gives you a version of the world from the point of view of the light source, it's possible for a shadow to tell you about things you can't actually see. This can be used to terrifying effect in horror films, where the monster is creeping up out of shot, and the only evidence of their presence is the shadow cast on the wall.

## **What use are shadows?**

Psychologists have spent a lot of time over the last ten years investigating how we see shadows, and what the human eyes and brain use them for. It seems that we use shadows for some things in our perception but not others. There's some evidence that the brain processes shadows very quickly, uses them to help work out the relationships between objects in the world, and then discards them. This is probably





why we don't tend to notice them that much. Our brains use shadows a lot for working out where things are in depth – for 3D vision – and for tasks like grasping and picking up objects. We don't seem to use shadows for things that depend on small details, and we don't seem to use them to work out much about the surface or light source.

Mostly we use shadows to tell us about the objects that are casting them. Humans never, except in optical illusions and tricks, mistake a shadow for an object.

### ***A shadowy nemesis***

Computer vision systems try and make sense of the world from analysing images and videos, and many people try to model computer vision systems on the human visual system. This seems a fairly sensible starting place, because it is one of the only examples we have that vision can work! Shadows, however, are problematic for computer vision systems. They keep getting mistaken for objects, which as we know is a mistake that humans do not make.

Computer vision researchers have been busy developing shadow detection and shadow removal techniques, so we can treat shadows as background and not as objects. How do you detect a shadow? Well, it's about the same colour (hue) as the background, but a bit darker. So either you make your background detection system less fussy about dark stuff, or you explicitly detect shadows and then lump them in with the background. This gets most of the shadows, but not all, so researchers are constantly coming up with new ways to improve the detection. One example of an enhanced technique is based on noticing

that, if the light stays the same, a piece of ground will look the same colour each time it's in shadow. So if you can remember that, you've got a head start in detecting the shadow.

### ***The shadow robots***

In computer vision and robotics, a few researchers are now beginning to exploit shadows as sources of information (not just as problematic dark patches to be ignored). Some Japanese researchers are fitting robots with bright lights so that the shadows they cast can tell them about the shape of the pipe or corridor they're going down. And I'm working in an international team trying to help a robot use shadows the way humans do. My Brazilian collaborators Paulo Santos and Valquiria

Fenelon have a robot, and we're using shadows to help it navigate around its environment and to guess the relationship between the robot itself, objects around it, and a light source. Just like humans do when we watch a horror film and see the shadow of the monster...

# *How to win love by not playing cool*

You know that feeling of awkwardness when you're talking to someone you fancy? That special mix of exhilaration and horror as you sweat, smile and stammer your way through an encounter with your beloved? It might be tempting to think that if you could just be cool and not give anything away, you'd be on your way to snagging the girl or boy of your dreams. Actually, as Vanessa Carpenter has found out, maybe what we need is more potential embarrassment, not less.

Vanessa works as an artist and designer for a group called Illutron and GeekPhysical in Copenhagen, Denmark. The projects she's done with her colleagues Mads Høbye and Dzl Møbius all use computers as a way to get people to socialise. One of the best ways to break the ice is to give people a chance to open up with one another. To share a moment that's a little bit personal. Maybe even a little bit...embarrassing.

Take a look at a few of Vanessa's projects and find out what awkwardness could do for your pulling power. Before you do, though, we should explain that when we say Vanessa works with computers, we're not talking about computers like the ones that sit on desks. She works in 'physical computing', building interactive gadgets that work within the real world. Designers in physical computing make computerised electronic gadgets with things like sensors, motors and displays. Here you'll see Vanessa's icebreaking designs at work in a toilet door, a corset and projected onto walls.

## **The ladies' and men's room mixup**

It started with nightclubs. To get ideas, Vanessa's group went to observe people in clubs and watched them struggle with a big problem. "They're with their friends and it's supposed to be this social place to meet someone, but it's almost impossible to approach anyone," she explains. "It's really hard to do the pickup or even make new friends in a nightclub because everyone's suspicious." You have to talk to someone to get to know them, but there's never a good excuse – leading to billions of bad chat-up lines. In awkward situations, though, talking is essential to fix things. So how do you create an awkward situation in a nightclub?

What Vanessa's group did was to replace the usual signs on the toilet doors with electronic displays whose symbols could change. Sometimes guys would head for what they thought was the right loo, only to see a group of girls come out. Groups of people outside and inside the toilets talked to one another to try and figure out what was going on. Sometimes the people who had figured out the game would try and guide other groups to the right place, or cheekily enter the 'wrong' toilet on purpose. As long as there were some people who didn't know what was happening, the awkward situation produced lots of playful interaction between the girls and guys in the club.



Image: Jonas Eriksson

## **Your heart, but not on your sleeve**

Pretend you're on a date. Your heart's pounding, you might be getting clammy hands, but you hope your date will never know. "So how do we get past that," Vanessa asks, "stop playing cool and start being honest here?" What she designed was a corset that tells the person wearing it when her heart rate rises, by getting tighter. Vanessa got a corset and put a heart rate monitor for runners inside it. Then she attached it to a circuit board that monitored the signal it got from the heart. When the wearer's heartbeat went above 75 beats a minute, the computer pumped air into pockets around the corset, which kept it tight until her heart rate went down again. That did two things: it told the wearer 'hey, your body thinks you're doing something interesting' and it gave them a sort of hugging feeling from the tightened corset.

Vanessa tested it by wearing it to a party, where, she says, "what ended up happening was kind of beautiful". Because of the music and talking in the party no one could hear the pump working, so Vanessa could be the only one who knew when her heart rate was up. Except when she and a friend left to get some supplies, the friend could hear the corset filling up as well. "So then he was aware every time my heart rate went up," says Vanessa, laughing. "So that was really embarrassing, because it was every five minutes and it was like, 'oh dear'." But once again, the situation was saved by awkwardness. Embarrassment turned into flirting, and pretty soon Vanessa and her friend were a couple.



Image: Jonas Eriksson

## **Projecting your feelings**

Sure, the corset worked when Vanessa was excited about being around someone, but it also inflated when she drank a Red Bull and went on the dance floor. So Vanessa and Illutron began working on a better system that works on more than just heart rate. Now they have one that looks at heart rate, body temperature, and even changes in your skin's electrical response. All those signals get sent to a laptop, and if you're at one of Vanessa's events, projected onto the wall.

You might think it would be embarrassing to have your body chemistry projected up onto a wall for everyone to see, but it turns out the effect can be exactly the opposite. At one party, Vanessa says, there were three shy guys sitting together, hiding from the crowd. She asked them if they'd mind putting on the monitors. They weren't sure at first but they agreed. Immediately, Vanessa says, their heart rates went through the roof, but after a few minutes they settled down...and suddenly they got confidence. "Then," says Vanessa, "three guys who were hiding in a corner are now going up to all the prettiest girls in the room and saying 'hey, that's my heart rate on the wall. Isn't that cool?' so now they have an excuse to talk to people." Once again, Vanessa and her group manage to fight awkwardness with awkwardness.



# Predicting cancer cures

**The most exciting research often comes about when people from different subjects work together, and it can come from unexpected directions. How might computer scientists help in the discovery of drugs to fight cancer?**

The obvious way is in building faster super-computers to number-crunch the problems. However, there are more subtle ways. Ideas from one area can have a big impact on others, if the researchers are creative enough to see the links. Professor Muffy Calder, a computer scientist at Glasgow University, discovered one intriguing link while working with cancer specialists. It turns out that the problems of understanding how drugs act on our cell chemistry are very similar to those in understanding communication networks. Tools for one can be used for the other.

Muffy and her team are aiming to understand the biochemical 'pathways' in which signals pass through from the membrane of cells into their nucleus. These 'pathways' are just a series of chemical reactions where different protein molecules are created and destroyed. To develop cancer drugs, scientists need to have a better understanding of how this happens. It will help them predict the way the reactions are affected by drugs.

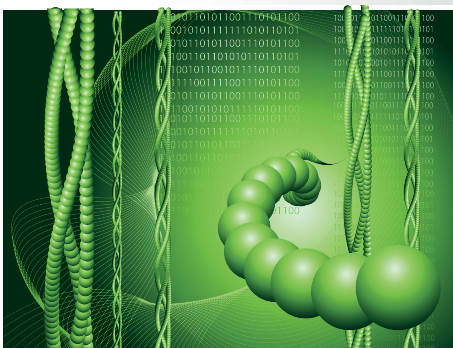
The pathways are normally modelled using complex maths built into tools that simulate the processes involved – allowing virtual experiments to be done on the computer instead of on real cells. These simulations are used to suggest actual experiments and to help understand the results. Muffy's team realised that the diagrams used by biochemists to illustrate the reactions are well known to computer scientists – they look just like the 'producer-consumer'

networks already used to analyse telecom networks. That means tools already developed for analysing telecom networks can be used to analyse the biological networks. What advantage does this give? Proof can be used rather than just simulation. Simulation allows you to check what happens in particular individual situations: the ones you simulate. Unfortunately other situations may or may not give the same results: you don't know unless you simulate them too. With the mathematical proof-based tools, general properties of the biological system as modelled can be shown to always be true. You can, for example, work out the probability that too much of a particular protein will be produced.

Previous work like this had a focus on proving properties at a molecular level: what will happen if single molecules react? Muffy realised that when developing drugs it's not what single molecules do that

and not enough reagent to trigger a reaction. This is a similar idea to one computer scientists use to reason about hardware circuits, thinking in terms of high and low voltage levels rather than about individual electrons.

The work opens up a whole new approach for developing drugs. The ultimate aim is to provide predictive tools for biochemists. They will suggest what effects different drugs might have on the processes taking place in cells and so suggest experiments to perform. If the team of computer scientists and cancer specialists do manage this, they will have handed biochemists a powerful new tool to help in the fight against cancer.



ultimately matters, but being able to predict how test tubes of reagents (substances that react) behave. That means you have to model something slightly different: what's known as the molar level, the test-tube level. It turns out that to prove properties interesting to the biochemists only two levels of concentration of reagents matter – high and low – corresponding to enough





# *The pen, the paper and the poet's daughter*

In the 1800s 'computers' were teams of people employed to calculate tables of numbers used, for example, when navigating. Charles Babbage changed that when he invented the first general purpose programmable computer. He called it the 'Analytical Engine' and it was intended to eradicate human error from the tables by taking over the whole job. Babbage is famous as the grandfather of computing as a result, though few people know of the other person who was vital to the success of the Analytical Engine.

The key thing to notice is that the Analytical Engine was the first programmable computer. Babbage's main concern was in creating the hardware but of course in doing so he also created a completely new job – that of computer programmer. Without programs for it to run, his machine was useless.

So Babbage's computers needed programs and for that he needed a woman called Ada Lovelace. She was the daughter of the poet Lord Byron and an accomplished mathematician, having been kept well away from her father, who was seen as a corrupting influence!

---

***You don't need a famous poet dad or even a computer to teach yourself to program.***

---

Unfortunately Babbage struggled to get funding for his ideas, not least because of his confrontational personality, and he never built the Analytical Engine. While waiting for the machine to be built, Ada kept herself busy. Even though the Analytical Engine didn't exist, she wrote programs for it based on Babbage's plans, and even tested them on paper even though she couldn't run them on the machine. The same technique is still used today by programmers to help get rid of bugs in their code at an early stage.

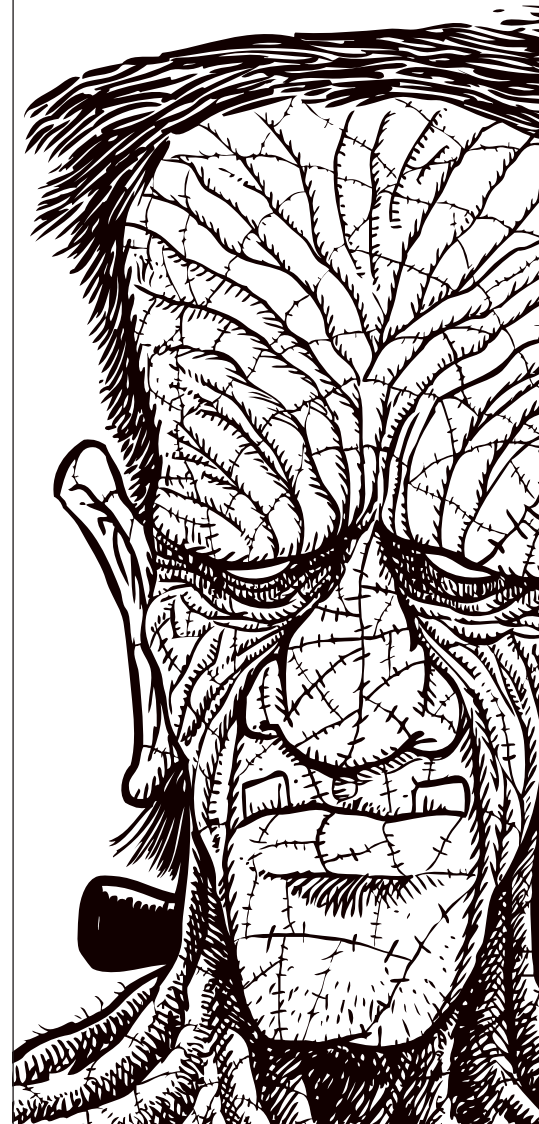


Some credit Ada with writing the very first program, though it is not entirely clear whether Ada or Charles actually did that. The earliest written record of programming, though, is of Ada correcting a faulty program written by Babbage.

Ada may be less well known than Charles but at least she actually finished her part of the work – unlike him.

## ***Frankenstein's Monster***

Around the time Ada Lovelace was writing programs, Mary Shelley, a friend of her father's, was writing a novel. In *Frankenstein*, inanimate flesh is brought to life. Life it may not be, but engineers are now doing pretty well in creating humanoid machines that can do their own thing. The 21st century is undoubtedly going to be the age of the robot. Maybe it's time to start thinking about the consequences in case they gain a sense of self.



# Cognitive crash dummies

The world is heading for catastrophe. We're hooked on power hungry devices: our mobile phones and iPods, our Playstations and laptops. Wherever you turn people are using gadgets, and those gadgets are guzzling energy – energy that we desperately need to save. We are all doomed, doomed...unless of course a hero rides in on a white charger to save us from ourselves.

Don't worry, the cognitive crash dummies are coming!

Actually the saviours may be a professor of human-computer interaction, Bonnie John, and a grad student, Annie Lu Luo. They're both from Carnegie Mellon University in the US and it's their job to figure out how well gadgets are designed.

If you're designing a bridge you don't want to have to build it before finding out if it stays up in an earthquake. If you're designing a car, you don't want to find out it isn't safe by having people die in crashes. Engineers use models – sometimes physical ones, sometimes mathematical ones – that show in advance what will happen. How big an earthquake can the bridge cope with? The mathematical model tells you. How slow must the car go to avoid killing the baby in the back? A crash test dummy will show you.

Even when safety isn't the issue, engineers want models that can predict how well their designs perform. So what about designers of computer gadgets? Do they have any models to do predictions with? As it happens, they do. Their models are called 'human behavioural models', but think of them as cognitive crash dummies. They are mathematical models of the way people behave, and the idea is you can use them to predict how easy computer interfaces are to use.

One very successful model is called 'GOMS'. When designers want to predict which of a few suggested interfaces will be the quickest to use, they can use GOMS to do it.

## Send in the GOMS

Suppose you are designing a new mobile phone. There are loads of little decisions you'll have to make that affect how easy the phone is to use. You can fit a certain number of buttons on the phone, but what should you make the buttons do? How big should they be? You can also use menus, but how many levels of menus should a user have to navigate before they actually get to the thing they are trying to do? More to the point, with the different variations you have thought up, how quickly will the person be able to do things like send a text message or reply to a missed call? These are questions GOMS answers.

To do a GOMS prediction you first think up a task you want to know about – sending a text message perhaps. You then write a list of all the steps that are needed to do it with your new design. Not just the button presses, but hand movements from one button to another, thinking time, time for the machine to react, and so on. In GOMS, your imaginary user already knows how to do the task, so you don't have to worry about spending time fiddling around or making mistakes. That means that once you've listed all your separate actions, GOMS can work out how long the task will take just by adding up the times for all the separate actions. Those basic times have been worked out from lots and lots of experiments on a wide range of devices. They have shown, on average, how long it takes to press a button and how long users are likely to think about it first.

## GOMS in 60 seconds?

GOMS has been around since the 1980s, but hasn't been used much by industrial designers yet. The problem is that it is very frustrating and time-consuming to work out all those steps for all the different tasks for a new gadget. That is all starting to change, thanks to a team of researchers led by Bonnie John. Her team have developed a tool called CogTool. You make a mock-up of your phone design in it, and tell it which buttons to press to do each task. CogTool then works out where the other actions, like hand movements and thinking time, are needed and makes predictions using GOMS.

Bonnie John came up with an easier way to figure out how much human time and effort a new design uses, but what about the device itself? How about predicting which interface design uses less energy? That is where Annie Lu Luo, comes in. She had the great idea that you could take a GOMS list of actions and instead of linking actions to times you could work out how much energy the device uses for each action instead. By using GOMS together with CogTools, a designer can now find out whether their design is the most energy efficient too.

So it turns out you don't need a knight on a white charger to help your battery usage, just Annie Lu Luo and her version of GOMS. Mobile phone makers saw the benefit of course. That's why Annie walked straight into a great job on finishing university.



# Sorry to bug you

In the 2003 film *The Matrix Reloaded*, Neo, Morpheus, Trinity and crew continue their battle with the machines that have enslaved the human race in the virtual reality of the Matrix. To find the Oracle, who can explain what's going on (which, given the twisty plot in the Matrix films, is always a good idea), Trinity needs to break into a power station and switch off some power nodes so the others can enter the secret floor. The computer terminal displays that she is disabling 27 power nodes, numbers 21 to 48. Unfortunately, that's actually 28 nodes, not 27! A computer that can't count and shows the wrong message!

Sadly there are far too many programs with mistakes in them. These mistakes are known as bugs because back in 1945 Grace Hopper, one of the female pioneers of computer science, found an error caused by a moth trapped between the points at Relay 70, Panel F, of the Mark II Aiken Relay Calculator being tested at Harvard University. She removed the moth, and attached it to her test logbook, writing 'First actual case of bug being found', and so popularised the term 'debugging' for testing and fixing a computer program.

Grace Hopper is famous for more than just the word 'bug' though. She was one of the most influential of the early computer pioneers, responsible for perhaps the most significant idea in helping programmers to write large, bug-free programs.

As a Lieutenant in the US Navy reserves, having volunteered after Pearl Harbor, Grace was one of three of the first programmers of Harvard's IBM Mark I computer. It was the first fully automatic programmed computer.

She didn't just program those early computers though, she came up with innovations in the way computers were programmed. The programs for those early computers all had to be made up of so-called 'machine instructions'. These are the simplest operations the computer can do such as to add two numbers, move data from a place in memory to a register (a place where arithmetic can be done in a

subsequent operation), jump to a different instruction in the program, and so on. Programming in such basic instructions is a bit like giving someone directions to the station but having to tell them exactly where to put their foot for every step. Grace's idea was that you could write programs in a language closer to human language where each instruction in this high-level language stood for lots of the machine instructions – equivalent to giving the major turns in those directions rather than every step.

The ultimate result was COBOL – the first widely used high-level programming language. At a stroke her ideas made programming much easier to do and much less error-prone. Big programs were now a possibility.

For this idea of high-level languages to work though you needed a way to convert a program written in a high-level language like COBOL into those machine instructions that a computer can actually do. It can't fill in the gaps on its own! Grace had the answer – the 'compiler'. It is just another computer program, but one that does a specialist task: the conversion. Grace wrote the first ever compiler, for a language called A-0, as well as the first COBOL compiler. The business computing revolution was up and running.

High-level languages like COBOL have allowed far larger programs to be written than is possible in machine-code, and so

ultimately the expansion of computers into every part of our lives. Of course even high-level programs can still contain mistakes, so programmers still need to spend much of their time testing and debugging. As the Oracle would no doubt say, "Check for moths, Trinity, check for moths".



# *The multi-million pound greeting*

There is real money to be made out there in the virtual world – if you are willing to put in the effort to develop appropriate skills.

You don't have to be young or a geek either. 62-year-old grandmother Jacquie Lawson turned a hobby into a multi-million pound business. She is a trained illustrator having originally studied art at St Martins School of Art in London. She bought her first computer in 1998. Despite struggling at the start she taught herself to draw computer animations using Macromedia Flash.

Just for fun she made an animated Christmas e-card and sent it to friends. Her skill as an illustrator combined with her artistic flair meant that suddenly she was inundated with people wanting them from around the world – a wonderful example of viral marketing. She set up a business, launched the [www.jacquielawson.com](http://www.jacquielawson.com) e-card website and is now the market leader – with double the visitors of its nearest rival.

---

***“The Internet is such a fantastic medium. It ought to be better.”***

---

As Jacquie says about the Internet: "It's such a fantastic medium. It ought to be better". She believes there is a lot of rubbish on the Internet – which means there is scope for skilled, creative people to make a difference by focusing on detail in what they do. Quality can stand out.

So develop the basic skills, have a great idea, throw in some business savvy...but most of all do it for fun, if you want to end up with a successful business.



## ***A gendered timeline of technology***

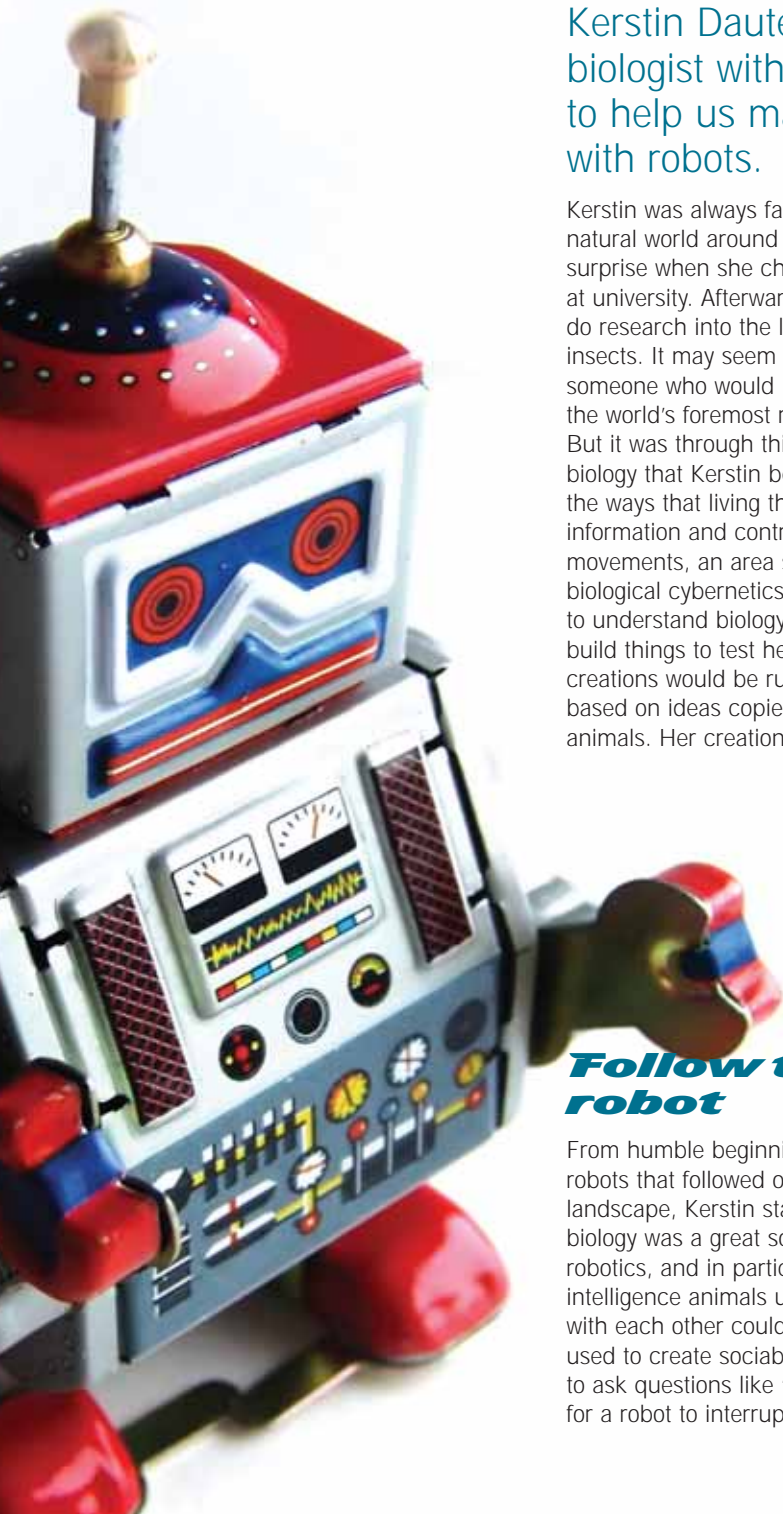
**1856 Statistician (and nurse) Florence Nightingale** returns from the Crimean War and launches the subject of data visualisation to convince politicians that soldiers are dying in hospital because of poor sanitation (See page 31).

**1912 Thomas Edison** claims "woman is now centuries, ages, even epochs behind man", the year after Marie Curie wins the second of her two Nobel prizes (See page 2).

**1927 Metropolis**, a silent science fiction film, is released. Male scientists kidnap a woman and create a robotic version of her to trick people and destroy the world. The robotic Maria dances nude to 'mesmerise' the workers. The underlying assumptions are bleak: women with power should be replaced with docile robots, bodies are more important than brains, and working class men are at the whim of beautiful gyrating women. Could the future be more offensive? (See page 26 and 27)

**1943 Thomas Watson**, the CEO of IBM, announces that he thinks: "there is a world market for maybe 5 computers". It's hard to believe just how wrong he was!

# Future friendly



Kerstin Dautenhahn is a biologist with a mission: to help us make friends with robots.

Kerstin was always fascinated by the natural world around her, so it was no surprise when she chose to study biology at university. Afterwards she went on to do research into the leg reflexes in stick insects. It may seem a strange start for someone who would later become one of the world's foremost robotics researchers. But it was through this fascinating bit of biology that Kerstin became interested in the ways that living things process information and control their body movements, an area scientists call biological cybernetics. This interest in trying to understand biology made her want to build things to test her understanding. Her creations would be run by computers but based on ideas copied from biological animals. Her creations would be robots.

## ***Follow that robot***

From humble beginnings building small robots that followed one another over a hilly landscape, Kerstin started to realise that biology was a great source of ideas for robotics, and in particular that the social intelligence animals use to live and work with each other could be modelled and used to create sociable robots. She started to ask questions like 'What's the best way for a robot to interrupt you if you are

reading a newspaper?' and perhaps most importantly 'When would a robot become your friend?' Now at the School of Computer Science at the University of Hertfordshire where she is a professor of artificial intelligence, she leads a world famous research group looking to try and build friendly robots with social intelligence.

## ***Good robot/ Bad robot***

Kerstin, like many other robotics researchers, is worried that most people tend to look on robots as being potentially evil. If we look at the way robots are portrayed in the movies that's often how it seems: it makes a good story to have a mechanical baddie. But in reality robots can provide a real service to humans, from helping the disabled, assisting around the home and even becoming friends and companions.

---

***This fall in acceptability is called the 'uncanny valley'***

---

The baddie robot ideas tend to dominate in the west, but in Japan robots are very popular and robotics research is advancing at a phenomenal rate. There has been a long history in Japan of people finding mechanical things that mimic natural things interesting and attractive. It is partly this cultural difference that has made Japan a world leader in robot research. But Kerstin and others like her are trying to get those of us in the west to change our opinions by building friendly robots and looking at how we relate to them.



Kerstin and her team

## ***Polite robots roam the room***

Kerstin decided that the best way to see how people would react to a robot around the house was to rent a flat near the university, and fill it with robots. Rather than examine how people interacted with robots in a laboratory, moving the experiments to a home with bookcases, biscuits, sofas and coffee tables made it real. She and her team looked at how to give their robots social skills. What was the best way for a robot to approach a person?

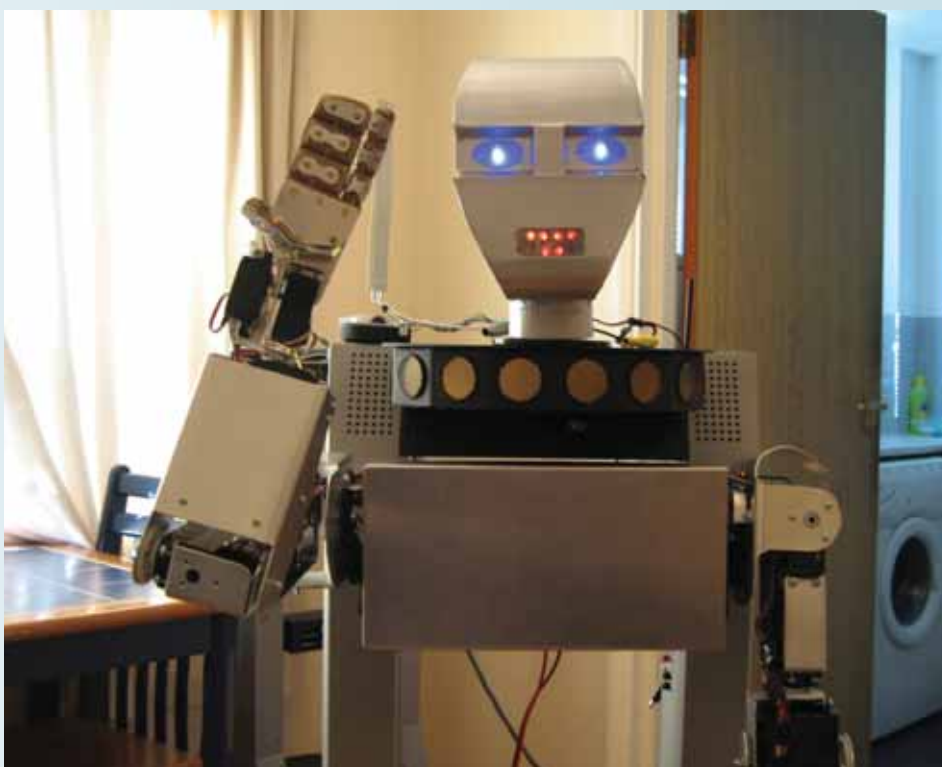
At first they thought that the best approach would be straight from the front, but they found that humans felt this too aggressive, so the robots were trained to come up gently from the side. The people in the house were also given special 'comfort buttons', devices that let them indicate how they were feeling in the company of robots.

Again interesting things happened. It turned out that not all, but quite a lot of people were on the whole happy for these robots to be close to them, closer in fact than they would normally let a human approach. Kerstin explains, "This is because these people see the robot as a machine, not a person, and so are happy to be in close proximity. You are happy to move close to your microwave, and it's the same for robots." These are exciting first steps as we start to understand how to build robots with socially acceptable manners. But it turns out that robots need to have good looks as well as good manners if they are going to make it in human society.

## ***Looks are everything for a robot?***

How we interact with robots also depends on how the robots look. Researchers had found previously that if you make a robot look too much like a human being, people

*Continued on page 16*



Images: Adaptive Systems Research Group, University of Hertfordshire

Continued from page 15



Images: Adaptive Systems Research Group, University of Hertfordshire



expect it to be a human being, with all the social and other skills that humans have. If it doesn't have these, we find interaction very hard. It's like working with a zombie, and it can be very frightening. This fall in acceptability of robots that look like, but aren't quite, human is what researchers call the 'uncanny valley'. People prefer to encounter a robot that looks like a robot and acts like a robot. Kerstin's group found this effect too, so they designed their robots to look and act the way we would expect robots to look and act, and things got much more sociable. But they are still looking at

how we act with more human-like robots and have built KASPAR, a robot toddler, which has a very realistic rubber face capable of showing expressions and smiling, and video camera eyes that allow the robot to react to your behaviours. He possesses arms so can wave goodbye or greet you with a friendly gesture. He's very lifelike and hopefully as KASPAR's programming grows and his abilities improve, he will emerge from the uncanny valley to become someone's friend.

## **Autism – mind blindness and robots**

The fact that most robots at present look like and act like robots can help in supporting children with autism. Autism is a condition that prevents you from developing an understanding of how to interact socially with the world. A current theory to explain the condition is that those who are autistic cannot form a correct understanding of others' intentions. It's called mind blindness. For example if someone came into the room wearing a hideous hat and asked you 'Do you like my lovely new hat?' you would probably think, 'I don't like the hat, but he does, so I should say I like it so as not to hurt his feelings'. You have a mental model of your friend's state of mind (that he likes his hat). An autistic person is likely to respond 'I don't like your hat', if this is what he feels. Autistic people cannot create this mental model so find it hard to make friends and generally interact with people, as they can't predict what people are likely to say, do or expect.

## **Playing with robot toys**

It's different with robots: many autistic children have an affinity with robots. Robots don't do unexpected things. Their behaviour is much simpler, because they act like robots. Using robots Kerstin's group have been examining how we can use this interaction with robot toys to help some autistic children to develop skills to allow them to interact better with other people. By controlling the robot's behaviours some of the children can develop ways to mimic social skills, which may ultimately improve their quality of life. There is no final conclusion yet, but some promising results, and this work continues to be one way to try and help those suffering with this socially isolating condition.

## **Working together**

It's only polite that the last word goes to Kerstin:

*"I firmly believe that robots as assistants can potentially be very useful in many application areas. For me as a researcher, working in the field of human-robot interaction is exciting and great fun. In our team we have people from various disciplines working together on a daily basis, including computer scientists, engineers and psychologists. This collaboration, where people need to have an open mind towards other fields, as well as imagination and creativity, is necessary in order to make robots more social."*

In the future, when robots become our workmates, colleagues and companions it will be in part down to Kerstin and her team's pioneering effort as they work towards making robots future friendly.



# Power play



How do you get power to a tiny island located in perilous waters? A group who call themselves the Nerd Girls have taken up the challenge and aim to demonstrate the excitement of engineering at the same time.

They are trying to find ways to supply Thacher Island with power and drinking water. It is home to the last operating twin lighthouses in the USA. As well as the lighthouses there are two cottages on the island that need power: one for the keeper's house and the other for guests. It's really important that the island has power but it hasn't been easy to get it there.

The island does have power coming via a cable running the couple of kilometres from the coastal town of Rockport on the mainland, under the Atlantic Ocean and onto the Island. Unfortunately the fierce local currents frequently disrupt the supply, and even when there is power, it's less than expected because water seeps into the power line.

This is where the Nerd Girls, a group of female researchers and students from Tufts University, come on the scene. They have created alternative energy systems to run everything on the island: the lights in the lighthouse, the water system, the refrigerators, and even a small electric golf cart that is used to haul materials around the island.

Exploiting wind energy wasn't possible because the island is small and home to wildlife. Herring gulls and Great black-backed gulls nest there in the spring, so anything that disrupts them or their chicks was out of the question. As a result the Nerd Girls decided to focus on developing solar energy as the alternative to that badly damaged and oft-repaired power line from

Rockport. The Nerd Girls installed a small, solar-powered beacon for one of the Lighthouses by making a battery that weighs just over 11kg and is charged by two solar panels. They've also developed solar energy systems for all of the island's other power needs.

So if you thought engineering was only for anti-social males whose thrill wiring has been short-circuited, the Nerd Girls are out to prove you wrong. In the process they are making sure they have a whole load of fun and that they really do make a difference.

# As easy as a bee sees?

If it weren't for the bees we would be in trouble. In the worst case, life on Earth could go the way of Mars. No plants, no animals, no life. Bees are the main way that flowers get pollinated. As the bees sip the nectar they carry pollen from flower to flower, allowing new generations of flowers to grow. But the way a flower looks to our eyes isn't the same way a bee sees it. For example, bee vision works into the ultraviolet part of the spectrum and under the correct lighting in a laboratory the wonderful, normally invisible, patterns that bees can see are revealed. Biologists all over the world have been collecting information about the sorts of patterns that particular flowers display. This display is called a spectral profile, and Samia Faruq, a former computer science undergraduate at Queen Mary, University of London, has done her bit to help these scientists peer into the world of the bees.



Samia Faruq

Her project involved creating a massive online database containing worldwide spectral profile information, so scientists can search this information easily. They can also combine information to help discover new facts using a method called clustering, where the computer pulls together all the data with similar properties.

Samia enjoyed the project: "I met and worked with amazing biologists during the project. It was great to find out what they needed and to be able to create it for them. I got the chance to collaborate and publish material together with them too. To know it will be used in their research is also very rewarding."

Samia is now studying for a PhD looking at colour vision in bees. That should keep her buzzing.



# Plague!

It's a well known story  
...death, panic, the  
collapse of civilisation...  
but this time it really  
happened.

The powers that be, complacent as ever,  
released a highly contagious disease.  
They thought it was ok – it was in a  
contained area with strict quarantine  
rules. But then the plague got out and  
spread to the cities. No-one was safe.  
Worldwide panic followed and society  
crumbled.

It really did happen, but luckily only  
avatars died. It was in the online game  
World of Warcraft. Still, it was a mistake,  
a bad mistake, but a mistake that may  
help us stop a similar thing happening

in the real world. Why? Because  
interdisciplinary computer scientist  
Nina Fefferman heard about it...

---

***A computer game,  
World of Warcraft,  
may help keep us safe  
from the plague.***

---

World of Warcraft has millions of users  
worldwide and the hardcore always need  
new challenges. So the game's organisers  
added a new one in a contained area –  
the plague. Would you be up for it? Could  
you survive the plague area? Trouble is it  
got out and spread very much like a real  
epidemic. It was carried around the  
virtual world by travellers and their pets.  
People went and had a look, believing  
they could avoid the risk themselves.  
They didn't!

The game was reset and it could have  
ended there. Except Nina heard about  
it. She studies the behaviour of people  
during disease outbreaks, creating  
computer models that simulate real  
epidemics. This helps scientists predict  
what might happen in future outbreaks  
and so helps policy-makers plan.

Her team studied what had happened in  
the game and saw that people's behaviour  
was both very realistic and included  
things no one had thought of before. As a  
result of the game outbreak, they are now  
adding such behaviour to the models so  
they can study the consequences.

They hope to run experiments in other  
games in the future. So you never know,  
your avatar might one day save your life...  
and millions of others too.



# *The optical Pony Express*

Suppose you want to send messages as fast as possible. What's the best way to do it? That is what Polina Bayvel, a professor at University College London, has dedicated her research career to: exploring the limits of how fast information can be sent over networks. It's not just messages that it's about nowadays of course, but videos, pictures, money, music, books – anything you can do over the Internet.

Send a text message and it arrives almost instantly. Sending messages hasn't always been that quick, though. The Greeks used runners – in fact the marathon athletic event originally commemorated a messenger who supposedly ran from a battlefield at Marathon to Athens to deliver the message 'we won' before promptly dying. One of the fastest women in the world right now, Paula Radcliffe, at her quickest could deliver a message a marathon distance away in 2 hours 15 minutes and 25 seconds (without dying!)

Horses improved things (and the Greeks in fact normally used horseback messengers, but hey, it was a good story). Unfortunately, even a horse can't keep up the pace for hundreds of miles. The Pony Express pushed horse technology to its limits. They didn't create new breeds of genetically modified fast horses, or anything like that. All it took was to create an organised network of normal ones. They set up pony stations every 10 miles or so right across

North America from Missouri to Sacramento. Why every 10 miles? That's the point a galloping horse starts to give up the ghost. The mail came thundering in to each station and thundered out with barely a break as it was swapped to a new fresh pony.

The Pony Express was swiftly overtaken by the telegraph. Like the switch to horses, this involved a new carrier technology – this time copper wire. Now the messages had to be translated first though, into electrical signals in Morse code. The telegraph was followed by the telephone. With a phone it seems like you just talk and the other person just hears but of course the translation of the message into a different form is still happening. The invention of the telephone was really just the invention of a way to turn sound into an electrical code that could be sent along copper cables and then translated back again.

The Internet took things digital – in some ways that's a step back towards Morse code. Now everything, even sound and images, are turned into a code of ones and zeros instead of dots and dashes. In theory images could of course have been sent using a telegraph tapper in the same way...if you were willing to wait months for the code of the image to be tapped in and then decoded again. Better to just wait for computers that can do it fast to be invented.



In the early Internet, the message carrier was still good old copper wire. Trouble is, when you want to send lots of data, like a whole movie, copper wire and electricity are starting to look like the runners must have done to horse riders: slow, out-of-date technology. The optical fibre is the modern equivalent of the horse. They are actually long thin tubes of glass. Instead of sending pulses of electricity to carry the coded messages, they now go on the back of a pulse of light.

Up to this point it's been mainly men taking the credit, but this is where Polina's work comes in. She is both exploring the limits of what can be done with optical fibres in theory and building ever faster optical networks in practice. How much information can actually be sent down fibres and what is the best way to do it?

Can new optical materials make a difference? How can devices be designed to route information to the right place – such 'routers' are just like mail sorting depots for pulses of light. How can fibre optics best be connected into networks so that they work as efficiently as possible – allowing you and everyone else in your street to be watching different movies at the same time, for example, without the film going all jerky? These are all the kinds of questions that fascinate Polina and she has built up an internationally respected team to help her answer them.

Why are optical fibres such a good way to send messages? Well the obvious answer is that you can't get much faster than light! Actually you can't get ANY faster than light. The speed of light is the fastest anything, including information, can travel according to Einstein's laws. That's not the end of the story though. Remember the worn out marathon runner. It turns out that signals being sent down cables do

something similar. Not actually getting out of breath and dying but they do get weaker the further they travel. That means it gets harder to extract the information at the other end and eventually there is a point where the message is just garbled noise.

What's the solution? It's exactly the one the Pony Express came up with. You add what are called 'repeaters' every so often. They extract the message from the optical fibre and then send it down the next fibre, but now back at full strength again. One of the benefits of fibre optics is that signals can go much further before they need a repeater. That means the message gets to its destination faster because those repeaters take time extracting and resending the message. That, in turn, leaves scope for improvement. The Pony Express made their 'repeaters' faster by giving the rider a horn to alert the stationmaster that they were arriving. He would then have time to get the next horse ready so it could leave the moment the mail was handed over. Researchers like Polina are looking for similar ways to speed up optical repeaters.

*Continued on page 22*



*Continued from page 21*

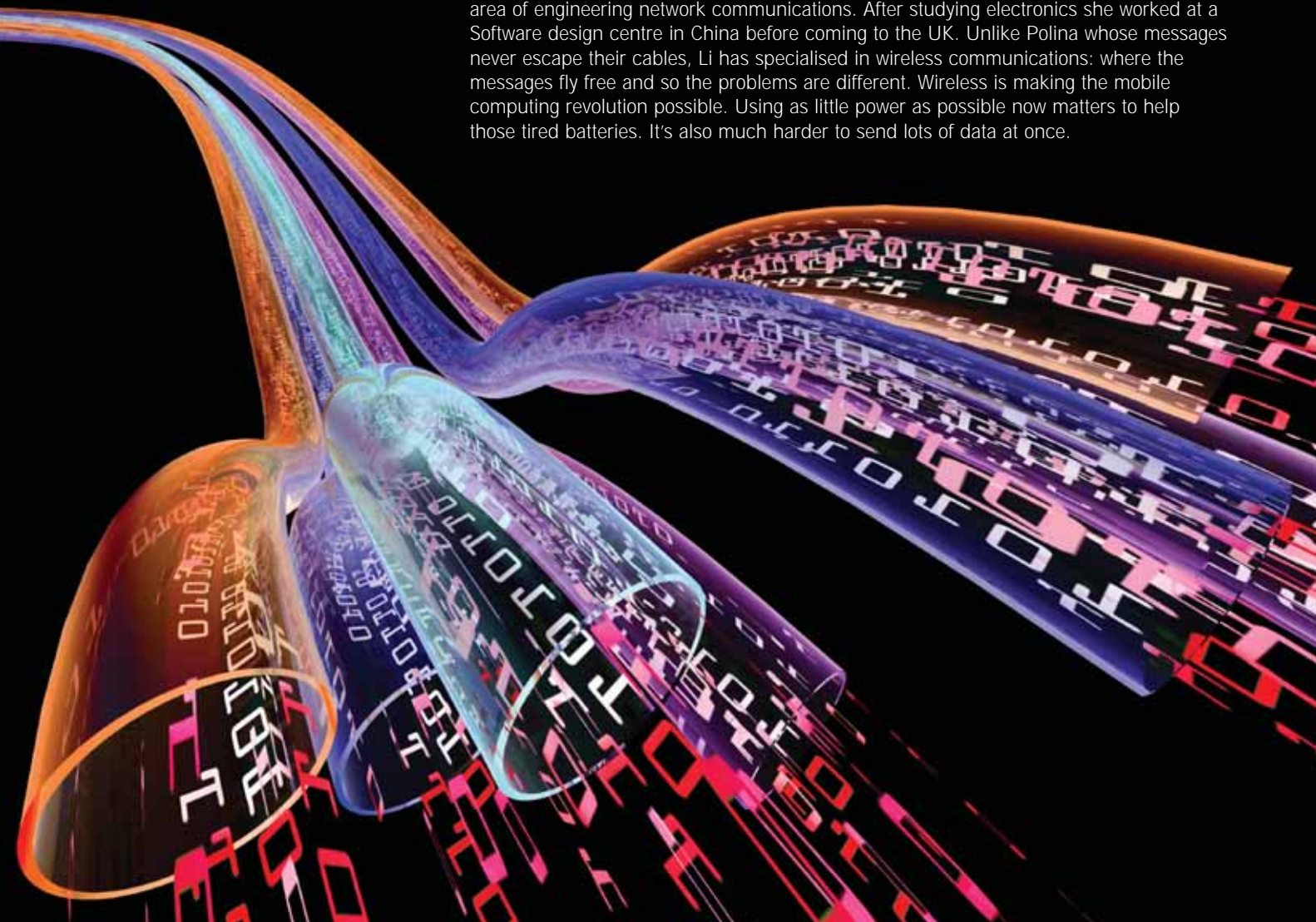
You can do more than play with repeaters to speed things up though. You can also bump up the amount of information you carry in one go. In particular you can send lots of messages at the same time over an optical fibre as long as they use different wavelengths. You can think of this as though one person is using a torch with a blue bulb to send a Morse code message using flashes of blue light (say), while someone else is doing the same thing with a red torch and red light. If two people at the other end are wearing tinted sunglasses then depending on the tint they will each see only the red pulses or only the blue ones and so only get the message meant for them. Each new frequency of light used gives a new message that can be sent at the same time.

The tricky bit is not so much in doing that but in working out which people can use which torch at any particular time so there aren't any clashes, bearing in mind that at any instant messages could be coming from anywhere in the network and trying to go anywhere. If two people try to use the same torch on the same link at the same time it all goes to pot. This is complicated further by the fact that at any time particular links could be very busy, or broken, meaning that different messages may also travel by different routes between the same places, just as you might go a different way to normal when driving if there is a jam. All this, and together with other similar issues, means there are lots of hairy problems to worry about if coming up with the best possible optical network as Polina is aiming to do.

Polina has been highly successful working in this area. She has been made a Fellow of the Royal Society of Engineering for her work and is also a Royal Society Wolfson Research Merit Award holder. It is only given to respected scientists of outstanding achievement and potential. She has also won the prestigious Patterson Medal awarded for distinguished research in applied physics. It's important to remember that modern engineering is a team game, though. As she notes, she has benefited hugely by having inspiring and supporting mentors, as well as superb students and colleagues. It is her ability to work well with other people that allowed her to build a critical mass in her research and so gain all the accolades. All that achieved and she is a mother of two boys to boot. Bringing up children is, of course, a team game too.

## ***Setting the bits free***

Li Zhang a Lecturer at the University of Leeds is another mother of two who works in the area of engineering network communications. After studying electronics she worked at a Software design centre in China before coming to the UK. Unlike Polina whose messages never escape their cables, Li has specialised in wireless communications: where the messages fly free and so the problems are different. Wireless is making the mobile computing revolution possible. Using as little power as possible now matters to help those tired batteries. It's also much harder to send lots of data at once.



# Synthetic speech



Computer-generated voices are encountered more and more frequently in everyday life, not only in automated call centres, but also in satellite navigation systems and home appliances.

Although synthetic speech is getting better, it's still not as easy to understand as human speech, and many people don't like synthetic speech at all. Maria Klara Wolters of Edinburgh University decided to find out why. In particular she wanted to discover what makes synthetic speech difficult for older people to understand, so that the next generation of talking computers will be able to speak more clearly.

She asked a range of people to try out a state-of-the-art speech synthesis system, tested their hearing and asked their thoughts about the voices. She found that older people have more difficulty understanding computer-generated voices, even if they were assessed as having healthy hearing. She also discovered that messages about times and people were well understood, but young and old alike struggled with complicated words, such as the names of medications, when pronounced by a computer.

More surprisingly, she found that the ability of her volunteers to remember speech correctly didn't depend so much on their memory, but on their ability to hear particular frequencies (between 1 and 3 kHz). These frequencies are in the

lower part of the middle range of frequencies that the ear can hear. They contain a large amount of information about the identity of speech sounds. Another result of the experiments was that the processing of sounds by the brain, so called 'central auditory processing' appeared to play a more important role for understanding natural speech, while peripheral auditory processing (processing of sounds in the ear) appeared to be more important for synthetic speech.

As a result of the experiments, Maria drew up a list of design guidelines for the next generation of talking computers, such as: make pauses around important words, slow down, and change to simpler forms of expressions (e.g. "the blue pill" is much easier to understand and remember than a complicated medical name). Such simple changes to the robot voices could make an immense difference to the lives of many older people. They will also make services that use computer-generated voices easier for everyone to use. This kind of inclusive design benefits everybody, as it allows people from all walks of life to use the same technology. Maybe Maria's rules would work for people you know too. Try them out next time grandpa asks you to repeat what you just said!

# ***Saving Bletchley Park***

Why did the allies win the second world war? Was it because of the amazing cunning of our generals, so often able to outwit their German opposite numbers? Or was the key that turned the war in the allies favour due to the efforts of the thousands of civilians, around half of them women, working at Bletchley Park? Bletchley was the top secret British code cracking centre, the birthplace of the computer, the place where they used their computing prowess to crack the German's codes. They were able to speed read the German's messages – even those of Hitler to his commanders.

That was why the Allied generals look so good. They knew what the Germans were intending to do. Bletchley shortened the war by several years, probably saving 22 million lives in the process. Now it needs our help to be saved as it is in danger of irreparable decay due to lack of funding. Software engineer, Sue Black of the University of Westminster has been leading a campaign to save it, very successfully so far as it has gained Heritage Lottery funding. Why not visit Bletchley Park or find out more about the campaign on Sue's blog, [www.savingbletchleypark.org/](http://www.savingbletchleypark.org/)



# ***My first signs***

Alexander Graham Bell was inspired by the deafness of his mother to develop new technologies to help. Lila Harrar, a former computer science student at Queen Mary, University of London, was also inspired by a deaf person to do something to make a difference. Her chance came when she had to think of something to do for her undergraduate project.

Her inspiration came from working with a deaf colleague in a part-time job on the shop floor at Harrods. The colleague often struggled to communicate to customers so Lila decided to do something to encourage hearing as well as deaf people to learn sign language. She developed an interactive tutor program that teaches both deaf and non-deaf users sign language. Her software includes games and quizzes along with the

learning sections...and it really could make a difference as she caught the attention of the company Microbooks. They were so impressed that they decided to commercialise it. As Lila discovered you need both creativity and logical thinking skills to do well at Computer Science...with both plus a bit of business savvy, you could become the country's next great innovator.





# Juggling priorities

Women have been innovating in the area of telecommunications for a long time. A big problem for telecom companies is how to cope when there is too much traffic. If they deal with it badly, no one ends up being able to get through (see Page 31 – how Madonna crashed the Internet). Erna Schneider Hoover, was one of the first people to tackle this problem. Back in the 1960s telephone exchanges were mechanical. An automatic system had replaced the operators who previously had connected calls by plugging wires into a board to link the telephone lines of the caller and the person they wanted to talk to. The mechanical ones were able to do this automatically based on the numbers dialed, but couldn't if too many people were trying to make calls at once. Erna sketched the solution to the problem from the hospital shortly after the birth of one of her three children. It involved a computerized switching system that was able to juggle the things it had to do much better, dropping less urgent tasks like record keeping and charging when things got busy. Telephone switches have been using her ideas ever since.

## **A gendered timeline of technology**

**1945** Grace Murray Hopper and her associates are hard at work on an early computer called Mark I when a moth causes the circuit to malfunction. Hopper (later made an admiral) refers to this as 'debugging' the circuit. She tapes the bug to her logbook. After this, computer malfunctions are referred to as 'bugs'. Her achievements didn't stop there: she develops the first compiler and one of the pioneering programming languages. (See page 12).

**1946** The **Electronic Numerical Integrator and Computer** is the world's first general purpose electronic computer. The main six programmers, all highly skilled mathematicians, were women. They were seen to be more capable programmers because it was considered too repetitive for men and as a result it was labelled 'sub-professional' work. Once more men realised that it was interesting and fun, programming was re-classed as 'professional', the salaries became higher, and men become dominant in the field.

**1949** A **Popular Mechanics** magazine article predicts that the computers of the future might weigh as little as 1.5 tonnes each. That's 11,023 iPods!

**1967** The original series of TV show **Star Trek** includes an episode where mad ruler Harry Mudd runs a planet full of identical female androids who are 'fully functional' at physical pleasure to tend to his whims. But that's not the end of the pleasure bots in this timeline...



***Killer robot?  
Evil scientist?!  
Helpless woman?!?  
(You can be the one to tell Angelina Jolie)***

Lots of people think that computer Science and IT are strictly for men only. That's really bizarre given that right from the start women like Grace Hopper and Ada Lovelace played pivotal roles in the development of computers, and women are still at the leading edge today. To be a successful modern IT pro you have to be a good team player, not to mention great at dealing with clients, which are skills women are generally good at.

'Geeky male computer scientist' is of course just a stereotype, like 'helpless

female in need of rescue by male hunk', 'scientist as mad eccentric in white coat', or 'evil robot wanting to take over the world'.

Where do false stereotypes come from? Films play a part in the way their (usually male, non-scientist) directors decide to represent characters.

Students on a 'gender in computer science' course at Siena College in the US watched lots of films with Computer Science plots from as far back as 1928 to



see how the way women, computers and computer scientists are portrayed has changed over time. Here are their views on some of those films.

# The real pros

So much for fiction ... what about real Women IT Pros? Equalitec ([www.equalitec.org.uk](http://www.equalitec.org.uk)) asked a few about their jobs. Here is what the real professionals think are the best things about what they do...

"Creating the initial ideas, forming the game, making the story... Being part of the creative process and having a hands on approach",  
 – Nana Louise Nielsen, Senior Game Designer, Sumo Digital

"Working with customers to solve their problems. The best feeling in the world is when you leave ... knowing you've just made a huge difference."  
 – Hannah Parker, IT Consultant, IBM

"It changes so often... I am not always sure what the day will be like"  
 – Madleina Scheidegger, Software Engineer, Google.

"I enjoy being able to work from home"  
 – Megan Beynon, Software Engineer, IBM

"I love to see our plans come together with another service going live and the first positive user feedback coming in"  
 – Kerstin Kleese van Dam, Head of Data Management, CCLRC

"...a good experienced team around me focused on delivering results"  
 – Anita King, Senior Project Manager, Metropolitan Police Service

"I get to work with literally every single department in the organisation."  
 – Jemima Rellie, Head of Digital Programme, Tate

Remember stereotypes are fiction, careers are what you make of them and real robots are (usually) nice!

## 1928 Metropolis

In a city of the future the ruling elite live in luxury while the workers live poorly in the underworld. An evil scientist substitutes a robot for a female worker activist. It purposely starts a riot as an excuse so reprisals can be taken. All hell breaks loose until the male hero comes to the rescue...

Computers	X	Evil
Women as IT Pros	X	Helpless
Computer Scientists	X	Evil

"Women are more or less portrayed as helpless ... The computer scientist ... as evil"

## 1956 Forbidden Planet

An all-male crew travel to Altair-4 to discover the fate of the colony there. They discover all that is left is scientist Dr Morbius, his beautiful daughter Altaira and a servent robot called Robby, programmed to be unable to harm humans. But what have Morbius' machines and experiments to do with the colony's fate?

Computers	✓	Helpful & Harmless
Women as IT Pros	X	love interest
Computer Scientists	X	Evil

"Altaira plays a typical woman's role...helpless...unintelligent ...Barbie-like"

## 1971 THX 1138

In an Orwellian future, an android controlled police state where everyone is made to take drugs that suppress emotion. LUH 3417 and THX 1138 stop taking their drugs, fall in love and try to escape...

Computers	X	Evil Police
Women as IT Pros	X	Few
Computer Scientists	X	Heartless

"The computer scientists are depicted as boring, heartless and easily confused"

## 1982 Blade Runner

In the industrial wastelands of a future Los Angeles, large companies have all the power. Robotic 'Replicants' are almost indistinguishable from humans but have incredible strength and no emotions. Deckard (Harrison Ford) must find and destroy a group of Replicants that have developed emotions and so threaten humanity as they rebel against being 'slaves'.

Computers	X	Evil
Women as IT Pros	X	None
Computer Scientists	X	Caused the problem

"A woman plays the minor role of a replicant...but is portrayed as a topless dancer"

## 1995 Hackers

A group of genius teenage hackers become the target of the FBI after they stumble across a high-tech embezzling scheme that is likely to cause a horrific environmental disaster. Dade Murphy and Kate Libby (Angelina Jolie) square off in a battle of the sexes and computer skills.

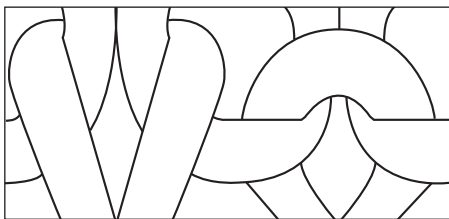
Computers	X	Used illegally
Women as IT Pros	✓	Elite...but illegal
Computer Scientists	X	Criminals

"Angelina plays a hard hitting, elite hacker who is better than everyone in her group except Dane who is her equal"

# Knitters and coders: separated at birth?

People often say that computers are all around us, but you could still escape your phone and iPod and go out to the park, far away from the nearest circuit board if you wanted to. It's a lot more difficult to get away from the clutches of computation itself though. For one thing, you'd have to leave your clothes at home. Queen Mary Electronic Engineer Karen Shoop tells us about the code hidden in knitting, and what might happen when computers learn to read it.

If you're wearing something knitted look closely at it (if it's a sunny day then put this article away till it gets colder). Notice how the two sides don't look the same: some parts look like a raised 'v' and others like a wave pattern. These are made by the two knitting stitches: knit and purl. With knit you stick the needle through and then behind the knitting; with purl you stick the needle in the other direction, starting behind the knitting and then pointing at the knitter [see picture]. Expert knitters know that there's more to knitting than just these two stitches, but we'll stick to knit and purl. As these stitches are combined, the wool is transformed from a series of waves or 'v's into a range of patterns: stretchy stripes (ribs), raised speckles (moss), knots and ropes (cable). It all depends on the number of purls and knits, how they are placed next to each other and how often things are repeated.



knit (k) and purl (p)

Knitters get very proficient at reading knitting patterns, which are just varying combinations of k (knits) and p (purls). So the simplest pattern of all, knitting a square, would look something like:

*'30k (30 knit stitches), finish the line, then repeat this 20 times'.*

A rib would look like:

*'5k, 5p, then repeat this [a certain number of times], then repeat the line [another number of times]'*

To a computer scientist or electronic engineer all this looks rather like computer code or, to be precise, like the way of describing a pattern as a computer program.

## How your jumper is like coding

So look again at your knitted hat/jumper/cardi and follow the pattern, seeing how it changes horizontally and vertically. Just as knitters give instructions for this in their knitting pattern, coders do the same when writing computer programs. Specifically programmers use things called **regular expressions**. They are just a standard way to describe patterns. For example a regular expression might be used to describe what an email address should look like (specifying rules such as that it has one '@' character in the middle of other characters, no full-stops/periods immediately before the @ and so on), what a phone number looks like (digits/numbers, no letters, possibly brackets or hyphens) and now what a knitting pattern looks like (lots of ks and ps). Regular expressions use a special notation to precisely describe what must be included, what might possibly be included, what cannot be, and how many times things should be repeated. If you were going to teach a computer how to read knitting patterns, a regular expression would be just what you need.

## Knitting a regular expression

Let's look at how to write a knitting pattern as a regular expression. Let's take moss or seed stitch as our example. It repeats a "knit one purl one" pattern for one line. The next line then repeats a "purl one knit one" pattern, so that every knit stitch has a purl beneath it and vice versa. These two lines are repeated for as long as is necessary. How might we write that both concisely and precisely so there is no room for doubt?

In knitting notation (assuming an even number of stitches) it looks like:

*Row 1: \*k1, p1; rep from \**

*Row 2: \*p1, k1; rep from \**

or

*Row 1: (K1, P1) rep to end*

*Row 2: (P1, K1) rep to end*

*Repeat these 2 rows for length desired.*

All this is fine ... if it's being read by a human, but to write experimental knitting software the knitting notation we have to use a notation a computer can easily follow: regular expressions fit the bill. Computers do not understand the words we used in our explanation above: words like 'row', 'repeat', 'rep', 'to', 'from', 'end', 'length' and 'desired', for example. We could either write a program that makes sense of what it all means for the computer, or we could just write knitting patterns for computers in a language they can already do something with: regular expressions. If we wanted to convert from human knitting patterns to regular expressions we would then write a program called a **compiler** (see Page 12) to do the translation.

In a regular expression, to give a series of actions, we just name them. So  $kp$  is the regular expression for one knit stitch followed immediately by one purl. The knitting pattern would then say repeat or rep. In a regular expression we group actions that need to be repeated inside curved brackets, resulting in  $(kp)$ . To say how many times we need to repeat, curly brackets are used, so  $kp$  repeated 10 times looks like this:  $(kp)\{10\}$ .

Since the word 'row' is not a standard coding word we then use a special character, written,  $\backslash n$ , to indicate that a new line (=row) has to start. The full regular expression for the row is then  $(kp)\{10\}\backslash n$ . Since the first line was made of repetitions of  $kp$  the following line must be made of repetitions of  $pk$ , or  $(pk)\{10\}\backslash n$

These two lines have to be repeated a certain number of times themselves, say 20, so they are in turn wrapped up in yet more brackets, producing:  $((kp)\{10\}\backslash n(pk)\{10\}\backslash n)\{20\}$ .

If we want to provide a more general pattern, not fixing the number of  $kp$  in a row or the number of rows, the 10 and 20 can be replaced with what are called variables -  $x$  and  $y$ . They can each stand for any number, so the final regular expression is:

$((kp)\{x\}\backslash n(pk)\{x\}\backslash n)\{y\}$

How would you describe a rib as a regular expression (remember, that's the pattern that looks like stretchy stripes)? The regular expression would be  $((kp)\{x\}\backslash n)\{y\}$ .

Regular expressions end up saying exactly the same thing as the standard knitting patterns, but more precisely so that they cannot be misunderstood. Describing knitting patterns in computer code is only the start, though. We can use this to write code that makes new patterns, to find established ones or to alter patterns, like you'd need to do if you were using thicker wool, for example. An undergraduate student at Queen Mary, Hailun Li, who likes knitting, used her knowledge to write an experimental knitting application that lets users enter their own combination of  $ps$  and  $ks$  and find out what their pattern looks like. She took her hobby and saw how it related to computing.

**Look at your woolly jumper again... it's really made out of computation!**



# Smart translation

Computers don't speak English, or Urdu or Cantonese for that matter. They have their own special languages that human programmers have to learn if they want to create new applications.

Even those programming languages aren't the language computers really speak. They only understand 1s and 0s. The programmers have to employ translators to convert what they say into Computerese (binary): just as if I wanted to speak with someone from Poland, I'd need a Polish translator. Computer translators aren't called translators though. They are called 'compilers', and just as it might be a Pole who translated for me into Polish, compilers are special programs that can take text written in a programming language and convert it into binary.

The development of good compilers has been one of the most important advancements from the early years of computing and Fran Allen, one of the star researchers of computer giant, IBM was awarded the Turing Prize for her contribution. That is the computer science equivalent of a Nobel Prize. Not bad given she only joined IBM to clear her student debts from university.

Fran was a pioneer with her groundbreaking work on 'optimising compilers'. Translating isn't just about taking a word at a time and substituting each for the word in the new language. You get gibberish that way. The same goes for computer languages.

Things written in programming languages are not just any old text. They are instructions. You actually translate chunks of instructions together in one go. You also add a lot of detail to the program in the translation, filling in every step.

Suppose a Japanese tourist used an interpreter to ask me for directions of how to get to Sheffield from Leeds. I might explain it as "Follow the M1 South from Junction 43 to Junction 33". If the Japanese translator explained it as a compiler would they might actually say (in Japanese): Take the M1 South from Junction 43 as far as Junction 42, then follow the M1 South from Junction 42 as far as Junction 41, then follow ... from Junction 34 as far as Junction 33. Computers actually need all the minute detail to follow the instructions.

The most important thing about computer instructions (i.e., programs) is usually how fast following them leads to the job getting done. Imagine I was on the information desk at Heathrow Airport and the tourist wanted to get to Sheffield. I've never done that journey. I do know how to get from Heathrow to Leeds as I've done it a lot. I've also gone from Leeds to Sheffield a lot, so I know that journey too. So the easiest way for me to give instructions for getting from London to Sheffield, without much thought and be sure it gets the tourist there might be to say:

Go from Heathrow to Leeds:

1. Take the M4 West to Junction 4B
2. Take the M25 clockwise to Junction 21

3. Take the M1 North to Leeds at Junction 43

Then go from Leeds to Sheffield

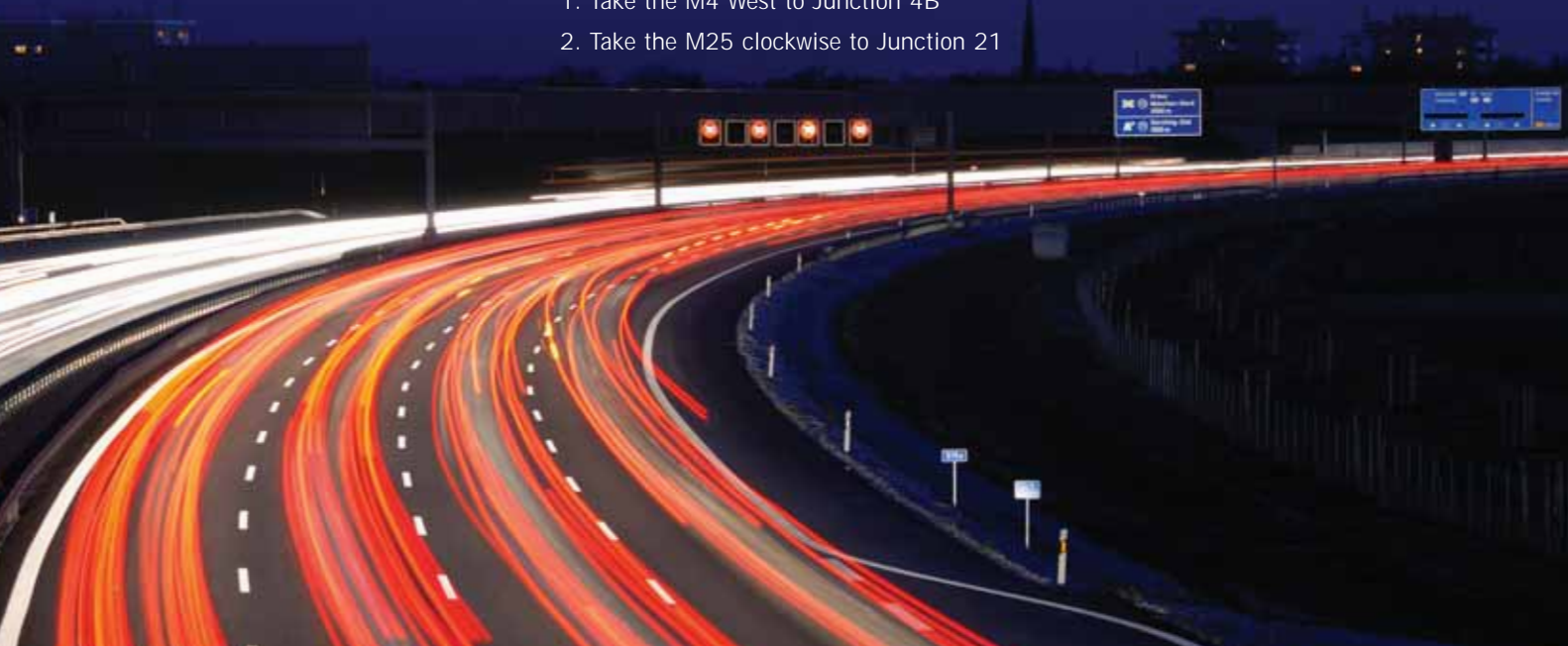
4. Take the M1 South to Sheffield at Junction 33

That is easy to write and made up of instructions I've written before perhaps. Programmers reuse instructions like this a lot – it both saves their time and reduces the chances of introducing mistakes into the instructions. That isn't the optimum way to do the journey of course. You pass the turnoff for Sheffield on the way up. An optimising compiler is an intelligent compiler. It looks for inefficiency and actually converts it into a shorter and faster set of instructions. The Japanese translator, if acting like an optimizing compiler, would actually remove the redundant instructions and simplify it to:

1. Take the M4 West to Junction 4B
2. Take the M25 clockwise to Junction 21
3. Take the M1 North to Sheffield Junction 33

Much faster! Much more intelligent! Happier tourists!

Next time you take the speed of your computer for granted, remember it is not just that fast because the hardware is quick, but because, thanks to people like Fran Allen, the compilers don't just do what the programmers tell them to do. They are far smarter than that.



# ***Rebel with a cause***

Florence Nightingale, the most famous female Victorian after Queen Victoria, is known for her commitment to nursing especially in the Crimean War. She rebelled against convention to become a nurse at a time when nursing was seen as a lowly job, not suitable for 'ladies'.

She broke convention in another less well-known, but much more significant way too. She was a mathematician – the first woman to be elected a member of the Royal Statistical Society. She also pioneered the use of pictures to present the statistical data that she collected about causes of war deaths and issues of sanitation and health.

Soldiers were dying in vast numbers in the field hospital she worked in... not directly from their original wounds but from the poor conditions. But how do you persuade people of something that (at least then) is so unintuitive? Even she originally got the cause of the deaths wrong, thinking they were due to poor nutrition, rather than the hospital conditions as her statistics later showed. Politicians, the people with power to take action, then, and probably now, were incapable of understanding statistical reports full of numbers. She needed a way to present the information so that the facts would jump out to anyone. Only then could she turn her numbers into life-saving action. Her solution was to use pictures, often presenting her statistics as books of pie charts and circular histograms.

Whilst she didn't invent them, Florence Nightingale certainly was responsible for demonstrating how effective they could be in promoting change, and so subsequently popularising their use. She undoubtedly saved more lives with her statistics than from her solitary rounds at night by lamplight.

Data visualisation is now an important area of computer science. As computers allow us to collect and store ever more data, it becomes harder and harder to make any sense of it all – to pick out the important nuggets of information that matter. Raw numbers are little use if you can't turn them into knowledge, or better still wisdom. The right kind of picture for the right kind of data can do just that as Florence Nightingale showed.

'The Lady of the Lamp': more than just a nurse, but a remarkable statistician and pioneer of a field of computer science... a lady who made a difference by rebelling with a cause.

## ***How Madonna crashed the Internet***

The Internet started off as typical boys toys. Designed for the military it was purpose built to keep going whatever: even in the event of nuclear war. Shame that one woman was able to bring it to its knees... pop star Madonna. When she took to the stage at Brixton Academy in 2001 for a rare appearance she made Internet history and caused more than a little Internet misery. Her concert performance was webcast; that is it was broadcast real time over the Internet. A record-breaking audience of 9 million tuned in, and that's where the trouble started... Go to the cs4fn website [www.cs4fn.org](http://www.cs4fn.org) to find out more.



# If you go down to the woods today

**What kind of emotion should computers evoke? Calm? Frustrating? Professor Yvonne Rogers tells us about her vision for the future.**

No one argues that computers should be frustrating to use, but Yvonne Rogers has a different idea of what the new vision could be. Not calm. Anything but calm in fact (apart from frustrating of course). Not calm, but engaging and exciting!

---

**Not calm,  
but engaging  
and exciting**

---

Her vision of a tranquil wood is not relaxing but provocative and playful. To prove the point her team turned some real woods in Sussex into an 'ambient wood'. The ambient wood was an enhanced wood. When you entered it you took probes with you, that you could point and poke with. They allowed you to take readings of different kinds in easy ways. Time hopping 'periscopes' placed around the woods allowed you to see those patches of woodland at other times of the year. There was also a special woodland den where you could then see the bigger picture of the woods as all your readings were pulled together using computer visualisations.



Not only is the ambient wood technology visible and in your face but it makes the invisible side of the wood visible in a way that provokes questions about the wildlife. You notice more. You see more. You think more. A walk in the woods is no longer a passive experience but an active, playful one. Woods are the exciting places of

childhood stories again but now there are even more things to explore.

The idea behind the ambient wood is to revisit the original idea of computers but in a new context. Computers started as tools, and tools don't disappear, they extend our abilities. Tools originally extended our physical abilities – a hammer allows us to hit things harder, a pulley to lift heavier things. They make us more effective and allow us to do things a mere human couldn't do alone. Computer technology can do a similar thing but for the human intellect...if we design them well.

---

***“The most important thing the participants gained was a sense of wonderment at finding out all sorts of things and making connections through discovering aspects of the physical woodland (e.g., squirrel's droppings, blackberries, thistles)”***  
– Yvonne Rogers

---

Many people dream about a future in which technology invisibly watches the world and removes the obstacles in the way before you even notice them. It's a little like the way servants to the aristocracy were expected to always have everything just right but at the same time were not to be noticed by those they served. The way this is achieved is to have technology constantly monitoring, understanding what is going on and how it might affect us and then calmly fixing things. The problem is it needs really 'smart' technology – a high level of artificial intelligence – to achieve, and that so far has proved more difficult than anyone imagined. Our behaviour and

desires are full of subtlety and much harder to read than was imagined. Even a super-intellect would probably keep getting it wrong.

There are also ethical problems. If we do ever achieve the dream of total calm we might not like it. It is very easy to be gung ho with technology and not realize the consequences. Calm computing needs monitors – the computer measuring everything it can so it has as much information as possible to make decisions from.

A classic example of how this can lead to people rejecting technology intended to help is in a project to make a “smart” residential home for the elderly. The idea was that by wiring up the house to track the residents and monitor them the nurses would be able to provide much better care, and relatives be able to see how things were going. The place was filled with monitors. For example, sensors in the beds measured resident's weight while they slept. Each night the occupants weight could invisibly be taken and the nurses alerted of worrying weight loss over time. The smart beds could also detect tossing and turning so someone having bad nights could be helped. A smart house could use similar technology to help you or I have a good night's sleep and help us diet.

The problem was the beds could tell other things too: things that the occupants preferred to keep to themselves. Nocturnal visitors also showed up in the records. That's the problem if technology looks after us every second of the day, the records may give away to others far more than we are happy with.

Yvonne's vision is different. It is not that the computers try to second-guess everything but instead extend our abilities. It is quite easy for new technology to lead to our being poorer intellectually than we were. Calculators are a good example. Yes we can do more





complex sums quickly now, but at the same time without a calculator many people can't do the sums at all. Our abilities have both improved and been damaged at the same time. Yvonne's probes allow you to see the woods in a new way, but to use the information however you wish. Crucially the probes encourage imagination too.

The alternative to the smart house (or calculator) that pampers allowing your brain to stay in neutral, or the residential home that monitors you for the sake of

the nurses and your relatives, is one where the sensors are working for you. Where you are the one the bed reports to helping you to then make decisions about

---

#### ***What next?***

***"I'd like to see kids discover new ways of probing their bodies to find out what makes them tick!"  
— Yvonne Rogers***

---

your health, or where the monitors you wear are part of a game that you play because it's fun.

So if Yvonne has her way, you won't be heading for a soporific future while the computer deals with real life for you. Instead it will be a future where the computers are sparking your imagination, challenging you to think, filling you with delight...and where the woods come alive again just as they do in the storybooks.



Images supplied by Yvonne Rogers

# *Byzantine birthdays*

You may not think of computers as argumentative, but some of them do actually bicker quite a lot, and for good reason. Many hi-tech systems we depend on rely on the right computers getting their way, and the problems involved are surprisingly tricky to get right. Barbara Liskov's contributions to this fiendishly difficult problem of making sure the right computers win their arguments helped her scoop the world's top computing prize in 2009.

The ACM Turing Award, which Barbara won, is the computing equivalent of a Nobel Prize. She was awarded it for more than 40 years' work. Her early work on programming languages has been used in every significant new language that has been invented since. More recently she has moved on to problems about 'distributed systems': computer systems involving lots of separate computers that have to work together. That's where the arguing comes in.

Barbara has been working on an area of distributed computing called 'Byzantine fault tolerance'. It's all about providing a good service despite the fact that just about anything can go wrong to the computers or the network connecting them. It's so important because those computers could be doing anything from running an e-commerce site over the Internet to keeping an airliner in the air.

## ***Happy birthday***

Here's a thought experiment to show you how tricky distributed computing problems can be. Alice is a cellist in an orchestra and it turns out that the birthday of the conductor, Cassie, is on the day they are playing a concert. Jo, the conductor's partner, has asked Alice to arrange for the orchestra to play Happy Birthday mid-concert as a surprise. The rest of the orchestra were enthusiastic. The only trouble is they didn't have the chance to agree which break to do it in. In fact, no one's sure they're definitely doing it at all, and now they are now all sitting in their places ready to start.

For it to work they all have to start precisely together or it will just sound horrible. If anyone starts on their own they will just look and sound silly. Can it be done, or should they all just forget the whole thing?





Alice decides to go ahead. She can tell the others it's on by whispering to those next to her and telling them to pass the message on. As long as her message gets to enough people across the different sections it will be ok. Won't it?

Actually no.

The problem is: how will she know enough people did get the message? It has to be passed when people aren't playing, so some could presumably not get it in time. How will she know? If the whispers didn't get through and she starts to play, she will be the embarrassed one.

### ***Are you in?***

That seems an easy thing to solve though – when each person gets the message they just send one back saying, "I'm in".

If she gets enough back, she knows it's on, doesn't she? Ahh! There the problem lies. She knows, but no one else can be sure she knows. If any are in doubt they won't play and it will still go horribly wrong. How does she let everyone know that enough people are willing to go for it?

Alice is in the same situation she was at the start! She doesn't know it will happen for sure and neither does anyone else.

She can start whispering messages again saying that enough people have agreed but that won't help in the end either. How does she know all the new messages get through?

### ***Change the problem***

A computer scientist might have a solution for Alice – change the problem.

Following this advice, she starts by whispering to everyone that she will stand up and conduct at an appointed time. Are they in? Now all she needs to be sure of is that when she stands up, enough people have agreed to play so that she won't look silly. The others send back their message saying 'I'm in', but no one else needs to know in advance whether the song is definitely going ahead. If she doesn't stand up they won't play. If she does, they go ahead.

### ***Delivering a good service***

Byzantine fault tolerance is about designing this kind of system: one that involves lots of 'agents' (people or computers) that have to come to an agreement about what they know and will do. The aim is for them to work together to deliver a service. That service might be for

an orchestra to play Happy Birthday, but is more likely to be something like taking airline bookings over the Internet, or even deciding on action to take to keep the airliner they are flying in the air. The separate computers have to agree as a group even when some could stop working, make mistakes due to software bugs or even behave maliciously due to a virus at any point. Can a way be engineered that allows the system as a whole to still behave correctly and deliver that service? This is the problem Barbara Liskov has been working on recently with Miguel Castro at MIT.

Of course they weren't thinking of birthdays and orchestras. They were interested in providing the service of looking after documents so they can be accessed anytime, anywhere. A simple computer does this with a file system. It keeps track of the names of your documents and where it has stored them in its memory. When you open a document it uses the records it has kept to go and fetch it from wherever it was put. With this kind of file system, though, if something goes wrong with your machine you could lose your documents forever.

*Continued on page 36*

Continued from page 35

### **Spread it around**

A way to guard against this is to create a file system that distributes copies of each file to different computers around the Internet. When you make changes, those changes are also sent to all the other computers with copies. Then if the copy on one machine is corrupted, perhaps by a hacker or just by a disk crash, the file system as a whole can still give you the correct document. That is where the computers need to start arguing. When you ask for your document back how do all the computers with (possibly different) copies decide which is the correct, uncorrupted version? That sounds easy, but as the orchestra example shows, as soon as you create a situation where the different agents (whether human or computer) are distributed, and worse you can't trust anyone or anything for sure, there are lots of subtle ways it can all go horribly wrong.

The way Barbara and Miguel's solution to this problem works is similar to what Alice was doing. One computer acts as what is called the 'primary' (Alice played this role). It is where the request from the client (Jo) goes. The primary sends out a request to

all the backup machines for the document, just like Alice's whispered messages. All the backups reply with their copy of the document. As soon as more than some predetermined number come back with the same document, then that is the good copy.

### **Not so simple**

Of course the detail of Barbara and Miguel's method is a lot trickier than that. They've had to figure out how to cope if something goes wrong with the primary (Alice herself) to ensure that the client still gets their document. Their version also works without any synchronisation to make things happen in lockstep (suppose Alice is at the back so can't stand up and conduct to keep things in time). There are lots of other details in Barbara and Miguel's version too. Messages are time-stamped, for example, so that the recipients can tell if a message is new or just a copy of an old one.

### **Practically perfect**

The particularly special thing about Barbara and Miguel's way of providing fault tolerance, though, is that it doesn't take an excessively long time. Various people had come up with solutions before,

### **General knowledge**

It's called Byzantine Fault Tolerance after some imaginary generals in the ancient days of the Eastern Roman Empire, whose capital was Byzantium. The original problem was about how two generals could know to attack a city at the same time.

but they were so slow no one could really use them. The new method is so fast it's almost as if you weren't bothering about fault tolerance at all. Better still, the fact that it doesn't need synchronisation – no conducting – means it also works when the replicated services are on the Internet where the computers act independently, and there is no way to keep them in lockstep.

Barbara's work might never actually help with an orchestral surprise like Alice's, However, because of it future computers will be in a better position to shrug off their own kind turning rogue due to hackers, cosmic rays or even just plain old breakdowns. Not a bad reason to have a byzantine argument.





# ***Lucky history***

Not everyone sets themselves goals when young and then spends their life achieving them. Sometimes it works best to just go with the flow. The direction a career takes often depends a lot on luck rather than planning. Take Fiona Polack, a senior lecturer at York, for example. She didn't follow an obvious path to get where she is at all, though computer scientists often don't. She now specialises in engineering complex systems: an area that pulls together aspects of computer science, engineering, biology and other sciences. She started off taking geography at university though, then going on to do a PhD in history at Cambridge (not the obvious start). How on earth did she get into engineering? Well, she took a career break to have children – one boy and one girl. On returning to her career she took an MSc in information processing having got the computer bug because as she has pointed out, she happened to do her PhD “in the only 1980s history group using computers”.

## ***A gendered timeline of technology***

**1972** Karen Spärck Jones publishes a paper describing a new way to pick out the most important documents when doing searches. Twenty years later, once the web is up and running, the idea comes of age. It's now used by most search engines to rank their results. (See page 46)

**1972** Ira Levin's book 'The Stepford Wives' is published. A group of suburban husbands kill their successful wives and create look-alike robots to serve as docile housewives. It's made into a film in 1975. Sounds like those men were feeling a bit threatened.

**1979** The US Department of Defense introduces a new programming language called Ada after Ada Lovelace. (See page 9)

**1982** The film *Blade Runner* is released. Both men and women are robots but oddly there are no male robots modelled as 'basic pleasure units'. Can't you guys think of anything else?

# ***Cold hard complexity: learning to talk in nature's language***

A gentoo penguin slumps belly-first on a nest at Damoy, on the Antarctic Peninsula. Nearby some lichen grows across a rock, and schools of tiny, shrimp-like krill float through the Southern Ocean. Every one of these organisms is a part of life in the Antarctic, and scientists study each of them. But what happens to one species affects all the others too. To help make sure that they all survive, scientists have to understand how penguins, plants, krill and everything else in the Antarctic interact with one another. They need to figure out the rules of the ecosystem.

## ***Working together***

When you're trying to understand a system that includes everything from plants to penguins, things get a bit complicated.

Fortunately, ecology has a new tool to help, called complexity theory. Anje-Margriet Neutel is a Biosphere Complexity Analyst for the British Antarctic Survey. It's her job to take a big puzzle like the Antarctic ecosystem, and work out where each plant and animal fits in. She explains that 'complexity is sort of a new brand of science'. Lots of science is about isolating something – say, a particular chemical – from its surroundings so you can learn about it, but when you isolate all the parts of a system you miss how they work together. What complexity tries to do is build a model that can show all the important interactions in an ecosystem at the same time.

## ***Energy hunt***

So for a system as big as a continent full of species, where do you start? Anje's got a sensible answer: you start with what you can measure. Energy's a good candidate. After all, every organism needs energy to stay alive, and staying alive is pretty much the first thing any plant or animal needs to do. So if you can track energy and watch it move through the ecosystem, you'll learn a lot about how things work. You'll find out what comes into the system, what goes out and what gets recycled.

## ***Playing with models***

Once you've got an idea of how everything fits together you've got what scientists call a model. Not a model you put together with glue, though – a mathematical simulation. The really clever thing you can do with models is start to mess around with them. As an example Anje says 'What would happen if you took one group of organisms and put in twice as much of them?' If you had a system with, say, twice as many penguins, the krill would have to be worried because more penguins are going to want to eat them. If they all run out what happens to the penguins? Or the seals that like eating krill too? It gets complicated pretty quickly, and those complicated reactions are just what scientists want to predict.

## ***The language of nature***

Figuring out how an ecosystem works is all about rules and structure. Ecosystems are huge complicated things, but they're not random – whether they work or not depends on having the right organisms doing jobs in the right places, and on having the right connections between all the different parts. It's like a computer program that way. Weirdly, it's also a bit like

language. In fact, Anje's background is in studying linguistics, not ecology. Think of an ecosystem like a sentence – there are thousands of words in the English language but in order to make a sentence you have to put them together in the right way. If you don't have the right grammar your sentence just won't make sense, and if an ecosystem doesn't have the right structure it'll collapse. Anje says that's what she wants to discover in the ecosystems she studies: 'I'm interested in the grammar of it, in the grammar of nature.'

### ***Surviving Antarctica***

Since models can help you predict how an ecosystem reacts to strange conditions, Anje's work could help Antarctica survive climate change. 'The first thing is to understand how the models work, how the models behave, and then translate that back to the biology that it's based on,' she explains. 'Then say OK, this means

we expect there may be vulnerable areas or vulnerable climate regions where you can expect something to happen if you take the model seriously.' If scientists like Anje can figure out how Antarctica's ecosystems are set up to work, they'll get clues about which areas of the continent are most at risk and what they can do to protect them.

Surviving on a continent where the temperature hardly ever gets above freezing is tough, and climate change is probably going to make it even tougher. If we can figure out how Antarctic ecosystems work, though, we'll know what the essential elements for survival are, and we'll have clues about how to make things better. Extracting the secret grammar of survival isn't going to be a simple job, but that's no surprise to the people working on it. After all, they're not called complexity scientists for nothing.

## ***A wild way to escape diseases***

Protecting wildlife might save humans as well as animals. In 2008 zoologists found that conserving animal habitats rather than building on them could prevent people from catching diseases. Lots of the most dangerous human diseases began by infecting animals, so figuring out what factors help viruses jump species could help prevent epidemics in the future. Kate Jones from the Institute of Zoology in London and a team of researchers used computer modelling to help do just that. They found that places where people have muscled in on a diverse animal habitat are also the most likely for diseases to make the big leap into humans. That means that programs designed to conserve wild habitats might have a great side benefit for humans too – by staying away it means that we'll catch fewer exotic animal lurgies.



# Under pressure

**What have jellyfish got to do with helping people recover from dance or football injuries?**

**Bushra Akhtar, a computer scientist and biologist at Queen Mary, University of London is working on it!**

You are made up of bits, little biological bits, called cells. When your cells work the way they should it's great and you're healthy, but in some medical conditions your cells undergo

a change that stops them working the way they should. An example is with cells you use every day walking to school, dancing, playing football or riding a bike. These are the cells in your cartilage, and they have a tough, high-pressure job to do.

Cartilage cells live in your joints: the joints of your arms, your legs, everywhere. Every step you take, your cartilage squashes and so do the cartilage cells inside the tissue. The cells detect these 'mechanical signals' and respond by producing all the things that keep the cartilage tissue healthy and strong. If this process goes wrong due to disease or if you stop squashing your cartilage (like weightless astronauts in space) then your cartilage breaks down.

The study of the way cells change shape when they are pushed, prodded or squashed, together with the biological consequences, is called 'mechanobiology'. It's a fascinating subject that involves understanding how all the clever engineering and physics inside a cell work, letting it keep its proper shape and perform its biological function. Examining the cells' 'mechanotransduction' (how

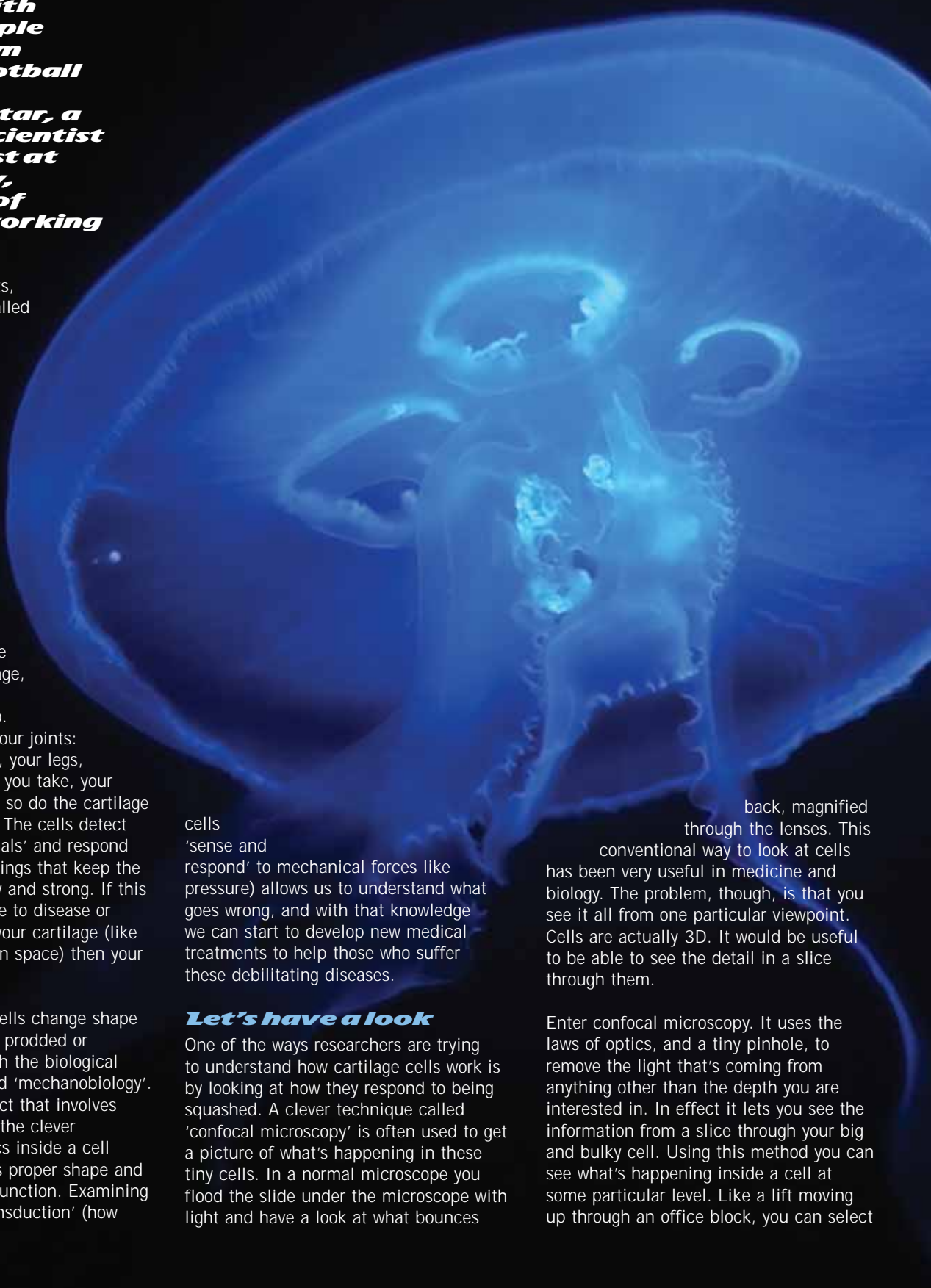
cells 'sense and respond' to mechanical forces like pressure) allows us to understand what goes wrong, and with that knowledge we can start to develop new medical treatments to help those who suffer these debilitating diseases.

## **Let's have a look**

One of the ways researchers are trying to understand how cartilage cells work is by looking at how they respond to being squashed. A clever technique called 'confocal microscopy' is often used to get a picture of what's happening in these tiny cells. In a normal microscope you flood the slide under the microscope with light and have a look at what bounces

back, magnified through the lenses. This conventional way to look at cells has been very useful in medicine and biology. The problem, though, is that you see it all from one particular viewpoint. Cells are actually 3D. It would be useful to be able to see the detail in a slice through them.

Enter confocal microscopy. It uses the laws of optics, and a tiny pinhole, to remove the light that's coming from anything other than the depth you are interested in. In effect it lets you see the information from a slice through your big and bulky cell. Using this method you can see what's happening inside a cell at some particular level. Like a lift moving up through an office block, you can select





the level in the cell to look at the information from. What's more you can put all those slices together to make a 3D image that you can view from any angle.

### ***It's all about labels and jellyfish***

Clever as it is confocal microscopy can't see the invisible. If you are interested in what gives a cell its mechanical properties, you have a problem.

The cytoskeleton (the 'skeleton of the cell') is what gives the cell its shape and physical strength like your skeleton does for you. Unfortunately it can't be seen using normal microscopy.

We need another clever idea. That's where labelling comes in. You can add chemicals to the cells that stick to the parts you're interested in, like the cytoskeleton. These labels can be created by a bit of genetic manipulation. Jellyfish, those floating blobs that can spoil a trip to the seaside, have given science a really useful tool. Some species of jellyfish 'glow', just like a white t-shirt does at a disco. You can take the jellyfish genes that make them glow, (called 'Green Fluorescence Protein' or 'GFP'), and add this genetic instruction into other cells so that parts of them glow in the dark when a light is shone at them. You can see where the jellyfish marker is

attached and with confocal microscopy take slices through this new luminous label, showing up the parts of the cell you're interested in.

### ***Actin apart, what's the bigger picture?***

Actin is a part of the cell cytoskeleton that helps give it its shape, and it can be labelled with the GFP jellyfish gene. Being able to see how the shape of the actin in a cell changes under mechanical stresses is a first step to being able to understand how cartilage cells work (or don't work) properly.

### ***Cells going down the tube***

You can get the images, but what do they tell you? In comes the computer science to help measure how the cell is doing and so understand the way a cartilage cell changes. Enter the work of computer scientist and biologist Bushra Akhtar. She wants to help find out the answer. She is working towards developing better ways to produce improved images of cells subjected to mechanical forces, and to create computer software to let us automatically measure these changes. She is exploring the use of a method called 'Digital Image Correlation'. It's an image processing method that looks for parts of an image that are the same; we say that same-looking parts are correlated. One use of the technique is to automatically spot suspicious behaviour, such as a passenger leaving a bag on a busy station platform.

By looking at slices produced by confocal microscopy her techniques can compute how the labelled cell parts change over time. They can extract precise numbers for how the cell parts move and deform when they are subjected to a mechanical load. Bushra sucks cells into a thin glass tube, called a pipette, and calculates how much different parts of the cell structure change as they are pulled in. This gives useful information on how cells respond to pressures and may in future help us understand mechanotransduction better. That will hopefully lead to new medical treatments: all thanks to a jellyfish, a very small hole in a microscope and some novel computer software.

**It could just help shape the future of cells!**

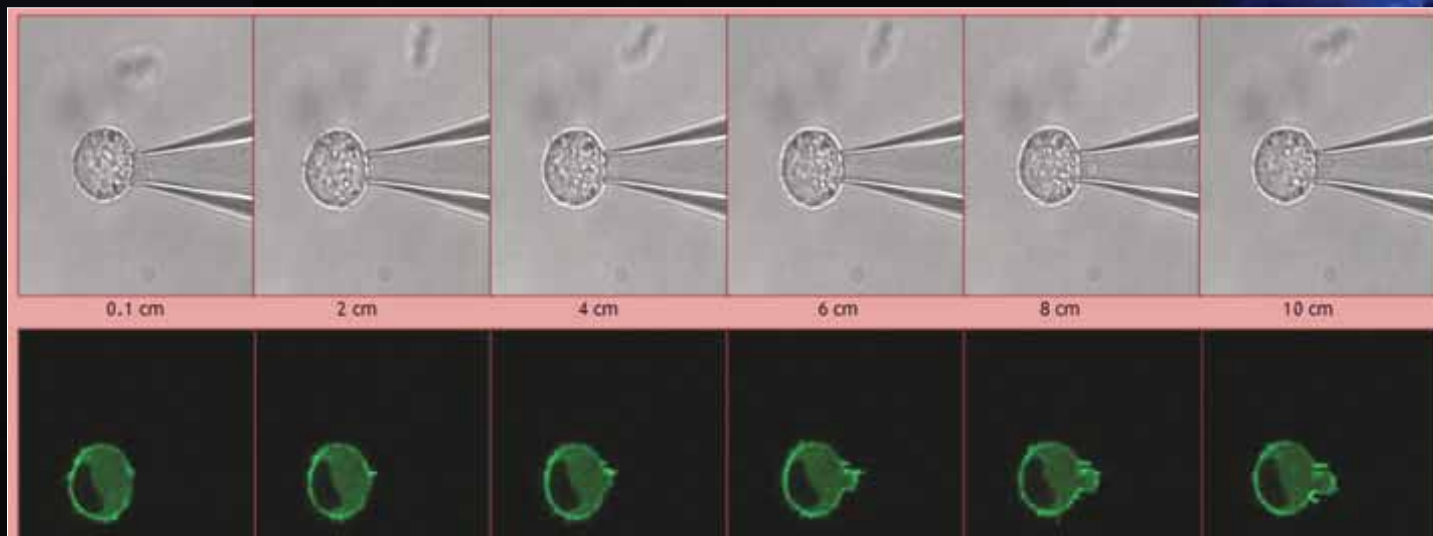


Image: Bushra Akhtar and Martin Knight

# Making sense of squishiness

Look out the window at the human-made world. It's full of hard, geometric shapes – our buildings, the roads, our cars. They are made of solid things like tarmac, brick and metal that are designed to be rigid and stay that way. The natural world is nothing like that though. Things bend, stretch and squish in response to the forces around them. That provides a whole bunch of fascinating problems for computer scientists like Lourdes Agapito to solve.

Computer scientists interested in creating three-dimensional models of the world have so far mainly concentrated on modelling the hard things. Why? Because they are easier! You can see the results in computer-animated films like *Toy Story*, and the 3D worlds like the ones in computer games. Even the soft things tend to be rigid.

Lourdes works in this general area creating 3D computer models, but she wants to solve the problems of creating them automatically just from the flat images in videos and is specifically interested in things that deform – the squishy things.

Look out the window and watch the world go by. As you watch a woman walk past you have no problem knowing that you are looking at the same person as you were a second ago – even if she becomes partially hidden as she walks behind the post box and turns to post a letter. The sun goes behind a cloud and the scene is suddenly darker. It starts to rain and she opens an umbrella. You can still recognise her as the same object. Your brain is pulling some amazing tricks to make this seem so mundane. Essentially it is creating a model of the world – identifying all the three-dimensional objects that you see and tracking them over time. If we can do it, why can't a computer?

Unlike hard surfaces, deformable ones don't look the same from one still to the next. You don't have to just worry about changes in lighting, them being partially hidden, and that they appear different from a different angle. The object itself will be a different shape from one still to the next. That makes it far harder to work out which bits of one image are actually the same as

the ones in the next. Lourdes has taken on a seriously hard problem.

Existing vision systems that create 3D objects have made things easier for themselves by using existing models. If a computer already has a model of a cube to compare what it sees with, then spotting a cube in the image stream is much easier than working it out from scratch. That doesn't really generalise to deformable objects though because they vary too much. Another approach, used by the film industry, is to put highly visible markers on objects so that those markers can be tracked. That doesn't help if you just want to point a camera out the window at whatever passes by though.

Lourdes' aim is to be able to point a camera at a deformable object and have a computer vision system be able to create a 3D model simply by analysing the images. No markers, no existing models of what might be there, not even previous films to train it with, just the video itself. So far her team have created a system that can do this in some situations such as when a person changes their facial expression. Their next goal is to be able to make their system work for a whole person as they are filmed doing arbitrary things. It's the technical challenge that inspires Lourdes the most, though once the problems of deformable objects are solved there are applications of course. One immediately obvious area is in operating theatres. Keyhole surgery is now very common. It involves a surgeon operating remotely, seeing what they are doing by looking at flat video images from a fibre optic probe inside the body of the person being operated on. The image is flat but the inside of the person that the surgeon is



trying to make cuts in is three-dimensional. It would be far less error prone if what the surgeon was looking at was an accurate 3D model of the video feed rather than just a flat picture. Of course the inside of your body is made of exactly the kind of squishy deformable surfaces that Lourdes is interested in. Get the computer science right and technologies like this will save lives.

Lourdes may or may not be the person who turns her team's solutions into the applications that in the future save lives in operating theatres, spot suspicious behaviour in CCTV footage or allow filmmakers to quickly animate the actions of actors. Whoever does create the applications, we still need people like Lourdes who are just excited about solving the fundamental problems in the first place.

### **Home time**

At the same time as tackling seriously hard if squishy computer science problems, Lourdes is also a mother of three. A major reason she can fit it all in, as she points out, is that she has a very supportive partner who shares in the childcare. Without him it would be impossible to balance all the work involved in leading a top European research team. It's also important to get away from work sometimes. Running regularly helps Lourdes cope with the pressures and she recently completed her first half marathon.



# Arabesque art

You don't need a paintbrush to create winning art. It's possible to create images by writing a program. Raghd Rostom demonstrated how beautiful and intricate patterns can be created in this way. Her programmed art based on an Arabesque style led to her winning BrainAcademy 2007, a competition about computer science and creativity.

Raghd used a drawing system called GeomLab to create her winning images. It was developed by Mike Spivey, a computer scientist at Oxford University, as a fun way for people to learn to program in a style known as functional programming.

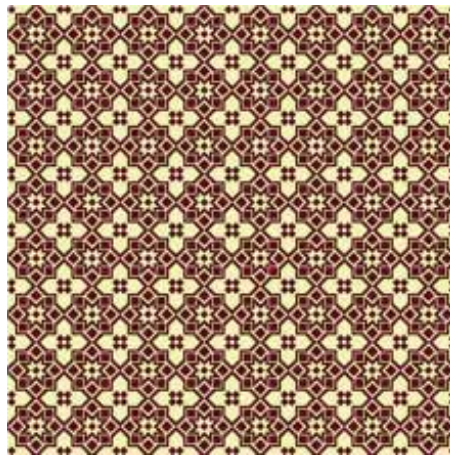
## Tile by numbers

The basic method Raghd used was to create tiles that she could then repeat. To create a tile involves specifying lines and filled areas by giving a series of pairs of numbers that give the coordinates. For example, the sequence of numbers: [0,0, 4,0, 4,4, 0,4, 0,0] would draw the outline of a square with corners at positions (0,0), (4,0), (4,4) and (0,4).

## Mixing light

Colours are indicated by giving three numbers. They stand for the different amounts of red, green and blue to be mixed. So if you want red you write: `rgb(1,0,0)`. It has red (1) but no blue (0) or green (0) – it is red. Yellow can be made by mixing red and green with no blue: `rgb(1,1,0)`. Notice that this system is not the familiar one of the way paint colours mix, but the way light colours mix. Light works differently. Mix all your paint together and you get brown. When you mix all the light colours together as in `rgb(1,1,1)` you get white light, which is why sunlight looks white – it's a mixture of all the colours in the rainbow.

Raghd used this method of colour mixing to create cream and brown colours: cream is `rgb(1,1,0.75)` and brown, `rgb(0.7,0.5,0.3)`. She then created the basic tile (shown above) that forms the foundation of her picture.



## How to program a mosaic

Whole pictures could be created as one big tile in this way, but for images that include repetition like Arabesque tilings, programming gives you a better way. You can bind up a basic tile as a new command – a function. You think up a name for it and link the name to the series of commands. Then whenever you want that pattern to appear, rather than give the full sequence of tiling commands, you just give your new command. The result is that that whole tile appears. Raghd created a tile called `mosaic` using a 'define' command:

```
define mosaic =  
_tile(36,36,0,0, [[0,0, 0,3, 18,12, ...
```

If you wanted then to create a row of 3 of these you would just need to repeat the command, `mosaic` 3 times using the `$` command to say "put them side by side":

```
mosaic $ mosaic $ mosaic
```

Raghd used a more flexible method than this called recursion to do the same thing. This involves writing a new command that says what to do with some arbitrary number of tiles, `n`.

If `n` is just 1 then we just put down our single tile, no problem:

```
row (1) = mosaic
```

This says that the new command `row(1)` means the same as `mosaic`.

What if `n` is more than 1? Then we want to place a single tile next to a row that is one tile shorter. A row of three is just a tile placed next to a row of 2. We can write this as follows:

```
row(n) = row(n-1) $ mosaic when n>1
```

This says a row of `n` is the same as a row of `(n-1)` placed next to a single tile when `n` is more than 1. Putting these together we get a little functional program:

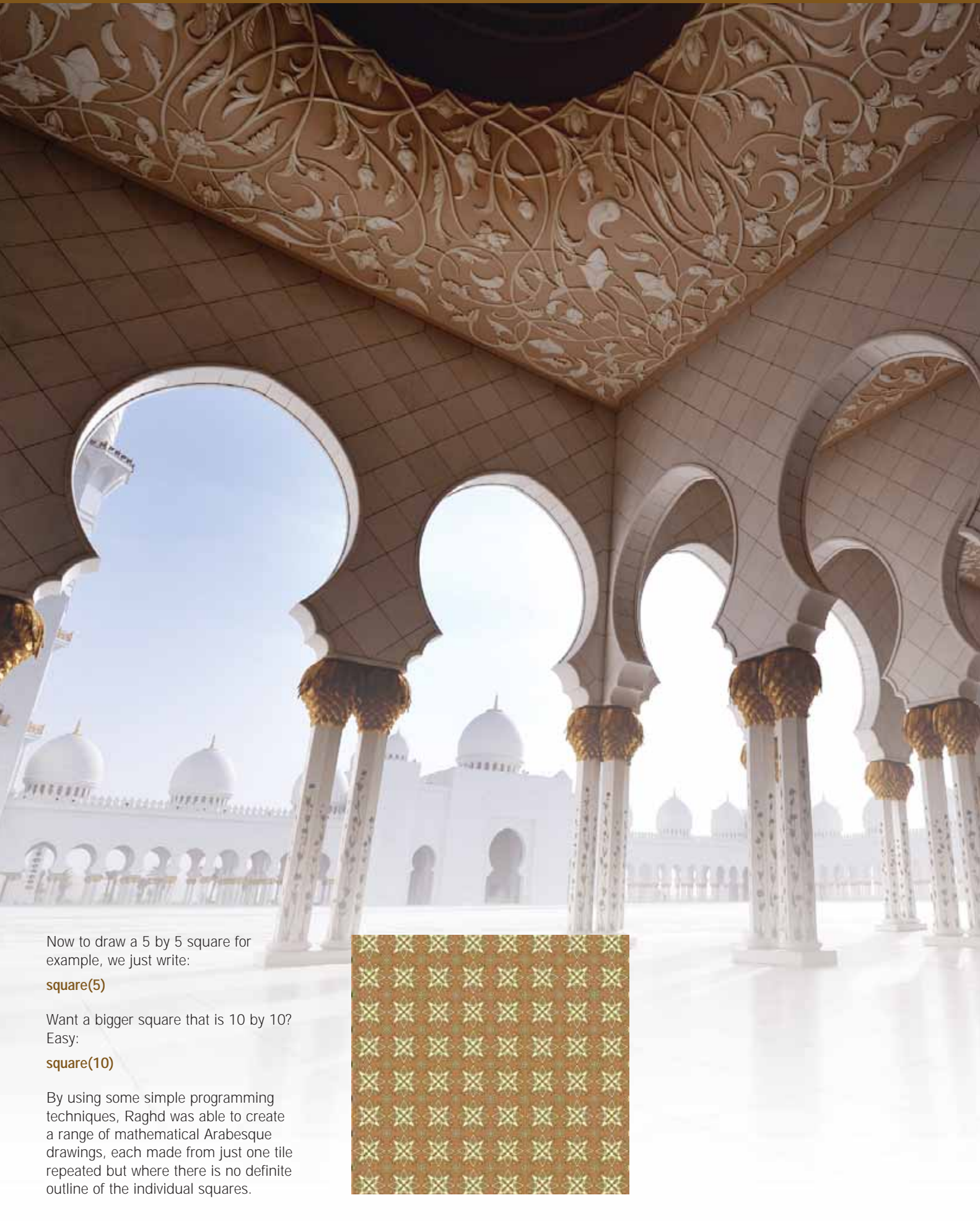
```
define  
row(n) = row(n-1) $ mosaic when n>1  
| row(1) = mosaic
```

Now we can just write `mosaic(3)` to get a row of three, but better still if we want a row of 5 we just write `mosaic(5)` instead. No extra programming required.

We can then go a step further and make increasingly large squares from our rows in a similar way. Given a small square we can make a bigger square by adding a new row on the top, then adding a new column down the side:

```
define  
square(n) = row(n) &  
square(n-1) $  
rot(row(n-1))) when n>1  
| square(1) = mosaic
```

A square of size 1 is just our `mosaic` tile. A square of size `n` is made from a row of size `n` placed on top of a square of size one less next to a rotated row of that same size. Note here how Raghd used the command called `rot`. It just rotates our row through 90 degrees before drawing it.



Now to draw a 5 by 5 square for example, we just write:

**square(5)**

Want a bigger square that is 10 by 10? Easy:

**square(10)**

By using some simple programming techniques, Raghd was able to create a range of mathematical Arabesque drawings, each one tiled from just one tile repeated but where there is no definite outline of the individual squares.



# *Playing the weighting game*

Imagine having a reality TV show where yet again Simon Cowell is looking for talent. This time it's talent with a difference though, not stars to entertain us but ones with the raw ability to help find webpages. Yes, this time the budding stars are all words. Word Idol is here!

The format is simple. Each week Simon's aim is to find talented words to create a new group: a group with star quality, a group with meaning. Like any talent competition, there are thousands of entries. Every word in every webpage out there wants to take part. They all have to be judged, but what do the specialist judges look for?

OK, we're getting carried away. Simon Cowell may not be interested but there is big money in the idea. It's a talent show that is happening all the time. The aim is to judge the words in each new webpage as it appears so that search engines can find it if ever someone goes looking. The real star of this show isn't Simon Cowell

but a Cambridge professor, Karen Spärk Jones. She came up with the way to judge words.

Karen worked out that to do this kind of judging a computer needs a thesaurus: a book of words. It just lists groups of words that mean the same thing. A computer, Karen realised, could use one to understand what words mean.

The fact that there are so many ways to say the same thing in human languages, makes it really hard for a computer to understand what we write. That is where a thesaurus comes in. If you ask a computer to search for web pages about whales, for example, it helps to know that, a page that talks about orcas is about whales too.

Worse still, most words have more than one meaning, a fact that keeps crossword lovers in business.

Take the following example:

**"Leona is the new big star of the music business."**

The word 'star' here obviously means a celebrity, but how do you know? It could also mean a sun or a shape. The fact that it's with the word 'music' helps you to work out which meaning is right even if you have no idea who or what Leona is. As Karen realised, a computer can also work out the intended meanings of words by the other words used with them. A thesaurus tells it what the critical groupings are, but



what Karen wanted was a way a computer could work the thesaurus out for itself and now she had a way.

Her early approach was to write a program that takes lots and lots of documents and make lists of the words that keep appearing close together. If 'music' appears with 'star' lots then that is a new meaning. After building up a big collection of such lists of linked words, the program can then use it to decide which pages are talking about the same thing and so which ones to suggest when a search is done.

So Karen had found the first way to judge whether a word has the right 'talent' to go in a group. The more often words appear

together the higher the score or 'weighting' they should be given. Simple!

The only trouble is it doesn't really work.

That is where Karen's big insight came. She realised that if two words appear together in a lot of different documents then, surprisingly perhaps, putting them together in a group isn't actually that useful for finding documents! Do a search and they will just tell you that lots of web pages match. What you really want is to be told of the few web pages that contain the meaning you are looking for, not lots and lots that don't.

The important word groupings are actually those that do appear together lots, but

only in a small number of web pages. That suggests they give a very focused meaning. Word groups like that help you narrow down the search. So Karen now had a better way to judge word talent. Give high marks for pairs that do appear together but in as few web pages as possible.

That idea was the big breakthrough and led to what is now called *idf weighting*. It is the way to judge words, and is so good that it's now used by pretty much every search engine out there.

Playing the idf weighting game may not make great TV but thanks to Karen it really does make for great web.

# Inspiring Wendy Hall

What inspires researchers to dedicate their lives to study one area? In the case of computer scientist Dame Wendy Hall it was a TV programme, *Hyperland*, starring former Dr Who Tom Baker and writer Douglas Adams of *Hitchhiker's Guide to the Galaxy* fame, that inspired her to become one of the most influential researchers of her area.

A pioneer and visionary in the area of web science, many of Dame Wendy's ideas have started to appear in the next generation web: the 'great web that is yet to come' (as Douglas Adams might put it), otherwise known as the *semantic web*. She has stacked up a whole bunch of accolades for her work. She is a Professor at the University of Southampton, a former president of the British Computer Society and now the first non-US president of the most influential body in computer science, the Association for Computing Machinery. She is also a Fellow of the Royal Academy of Engineering and last year she topped it all and gaining her most impressive sounding title for sure by being made a Dame Commander of the British Empire.

So how did that TV programme set her going?

Douglas Adams and Tom Baker acted out a vision of the future, a vision of how TV was going to change. At the time the web didn't exist and TV was just something you sat in front of and passively watched. The future they imagined was interactive TV. TV that was personal. TV that did more than just entertain but served all your information needs.

In the programme Douglas Adams was watching TV, vegetating in front of it...and then Tom Baker appeared on Douglas's screen. He started asking him questions...and then he stepped out of the TV screen. He introduced himself as a software agent, someone who had all the information ever put into digital format at his fingertips. More than that he was Douglas's personal agent. He

would use that information to answer any questions Douglas had. Not just to bring back documents (Google-style) that had something to do with the question and leave you to work out what to do with it all, but actually answer the question. He was an agent that was servant and friend, an agent whose character could even be changed to fit his master's mood.

Wendy was inspired...so inspired that she decided she was going to make that improbable vision a reality. Reality hasn't quite caught up yet, but she is getting there.

Most people who think about it at all believe that Tim Berners-Lee invented the idea of the web and of hypertext, the links that connect web pages together. He was the one that kick-started it into being a global reality, making it happen, but actually lots of people had been working in research labs round the world on the same ideas for years before, Wendy included, with her *Microcosm* hypermedia system. Tim's version of hypermedia – interactive information – was a simple version, one simple enough to get the idea off the ground. Its time is coming to an end now though.

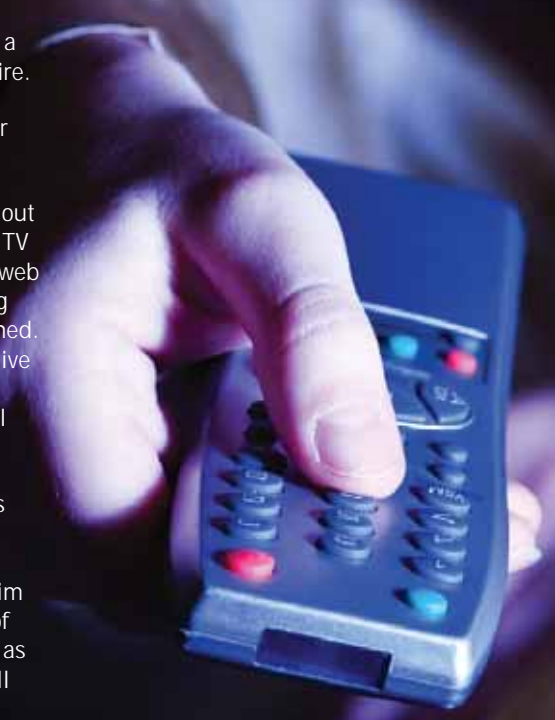
What is coming next? The semantic web, based on the ideas of Wendy and others, and it will be much more powerful.

It is a version of the web much closer to that TV programme, a version where the web's data is not just linked to other data but where words, images, pictures, videos are all tagged with meaning: tags that the software agents of the future can use to understand.

The structure is now there for it to happen. What is needed is for people to start to use it, to write their web pages that way, to actually make it everyday reality. Then the web programmers will be able to start innovating with new ideas, new applications that use it, and the web scientists like Wendy will be able to study it: to work out what works for people, what doesn't and why.

Then maybe it's your turn to be inspired and drive the next leap forward.

*This article is inspired by a keynote talk Wendy Hall gave in Madrid, 2008.*





# ***Sisters are doin' it for themselves***

Back in 1999 at Carnegie Mellon University in the US, they thought there ought to be a group for women computer scientists to get together socially and help each other. Since then they've grown into a professional organization working both on and off campus to show how exciting computer science is and also that it thrives on diversity. They use 'speed dating' to pair up newcomers with 'Big Sisters' at the start

of the year. That way everyone has someone to talk to. Now, not only do they help each other, they give talks, do research, practice professional skills, swap tips and just make friends. They keep the support up even after they've left uni and got jobs. Visit their website at [women.cs.cmu.edu](http://women.cs.cmu.edu) to see what they're up to and maybe even get ideas to form your own group!



## ***Saving the Iberian Lynx***

Manchester University run an annual competition that challenges UK students aged 7-19 to create an animated film using their computer. In 2009 the competition coincided with a special issue of cs4fn on 'Programming to save the world'. As a consequence we sponsored a prize for Best Environmental Awareness Animation.

It was won by Alison MacPherson from Kinlochberrie High School in Scotland for her animated short on the plight of the Iberian Lynx. Watch it at [www.cs.manchester.ac.uk/Animation09/](http://www.cs.manchester.ac.uk/Animation09/) along with all the other winning entries. Why not give it a go and enter a short yourself next year?

## ***A gendered timeline of technology***

**1984** Technology anthropologist **Lucy Suchman** draws on social sciences research to overturn the current computer science thinking on how best to design interactive gadgets that are easy to use. She goes on to win the Benjamin Franklin Medal, one of the oldest and most prestigious science awards in the world.

**1985** In the film **Weird Science**, two teenage supergeeks hack into the government's mainframe and instead of using their knowledge and skills to do something really cool...they create the perfect woman. Yawn. Not again.

**1995** **Angelina Jolie** stars as the hacker Acid Burn in the film **Hackers**, proving once and for all that women can play the part of the technologically competent in films.

**2004** A new version of **The Stepford Wives** is released starring Nicole Kidman. It flops at the box office and is panned by reviewers. Finally! Lets hope they don't attempt to remake this movie again.

**2005** The president of **Harvard University**, Lawrence Summers, says that women have less "innate" or "natural" ability than men in science. This ridiculous remark causes uproar and Summers leaves his position in the wake of a no-confidence vote from Harvard faculty.

# ***DON'T press that button!***

Doctors and nurses are there to save lives and most of the time that is what they do. Their job is tough, though, and if they make mistakes they can kill or seriously injure the people they are trying to save. Thousands of people do, unfortunately, suffer from mistakes every year. It's only a small number of those being treated but for the individuals concerned a mistake can be devastating. Many involve medical gadgets like the ones that release drugs through tubes to keep you alive or out of pain. At the end of the tubes there is a computer constantly controlling how much drug actually goes into your body.

Should we blame the doctors and nurses who make the mistakes? That is what often happens, but maybe it's not actually their fault. Perhaps, for example, those gadgets could be designed differently, in a way that makes mistakes less likely in the first place. Maybe by just looking for someone

to blame we miss the chance to prevent similar problems in the future.

A new £6 million project called CHI+MED run by Ann Blandford, a professor at University College London, is going to tackle this problem. It aims to make a

difference – a BIG difference. If it succeeds it will completely change the way medical devices are designed, bought and used so those mistakes are largely a thing of the past.





Ann is the director of the UCL Interaction Centre. It combines computer science and psychology research. She has a rather varied background, having originally studied mathematics and then worked in industry as a computer engineer before doing a PhD in artificial intelligence and then switching to cognitive psychology research. She combined that with her interest in computer science to become a world expert in interaction design. Building this highly successful career didn't prevent her from raising a family too: she has two, now grown up, daughters. Oh, and if you aren't impressed yet, she's also a rock climber in her spare time!

She is coordinating a team on CHI+MED that also includes researchers from Swansea and two other London universities (City and Queen Mary) as well as hospitals in both cities, so she's going to be busy for the next 6 years. As she pointed out, though, the extra work is nothing compared to running a whole research centre!

So what can be done about all those mistakes doctors and nurses make? Well it's obviously crucial that medical gadgets are easy to use. The key is to understand

why people make mistakes and then to design the gadgets so they both help people avoid making errors and help recover if things do go wrong.

The first thing to accept is that everyone makes mistakes. Have you ever got the wrong answer because you keyed the wrong things into a calculator? Now imagine those numbers you were typing in were setting how much painkiller a patient got. You are doing it lots of times every day with slightly different calculators and in different situations. You are being interrupted all the time. If you are a nurse and make a similar mistake to the ones you probably make all the time with calculators, you could very easily give someone an overdose. You must NEVER make the mistake. Try it from now on with calculators. From now on, don't make a single mistake using a calculator ... for the next 40 years or so. Are you up to the task?

Is it really that simple to make mistakes that kill though? Take the following example, based on a real hospital incident. A nurse was used to using a particular syringe pump (one of the gadgets that just pumps a drug into the person it's tubed up

to). On this day she was using a different pump to give a patient a dose of a drug. The second one was from the same manufacturer and looked very, very similar, except there was a hidden difference in what certain identical-looking buttons did. With the first pump if you pressed the units button enough times it would cycle around the unit values: 8,9,0. She assumed pressing the same button on the second device did the same thing. It didn't. Instead, it cycled into the tens: 8,9,10. Neither of the two gadgets alert the nurses using them about what happens when the units button is pressed more than nine times: the keys beeps in exactly the same way. The result here was a big overdose of drug was given to the patient: out by a factor of 10. That can be enough to kill.

How is CHI+MED going to tackle such problems? Well an incident like this will trigger a cascade of different kinds of research. The team might find out about it from reading hospital or research reports, or from the team's own researchers who will be in hospitals talking to doctors and nurses about their jobs, watching operations, and so on.

*Continued on page 52*

*Continued from page 51*

A first step will be to create realistic simulations of both pumps that can be used in subsequent parts of the research. The team creating the simulations will investigate what other features are different and what are similar in this family of pumps, using mathematical tools to automate doing this. In the process methods and tools to analyse the designs will be developed with device designers. The aim will be to make it easier for them to do similar evaluations of their designs. That will help ensure the gadgets of the future don't have important inconsistencies in the first place.

As new inconsistencies come to light the team, working with the hospitals, will investigate more thoroughly what impact they could have in practice. Given the way the devices are actually used, could the differences lead to patients being harmed? That is all part of thoroughly understanding the situation that the gadgets are to be deployed in, something that is very important for any new IT system.

This kind of mistake made by the nurse, where someone follows familiar steps in the wrong situation is called a 'capture error', because your behaviour has been 'captured' by a habit. It is a well-known kind of mistake that people make regularly. You may have done it in telling someone your home phone number instead of some other one you intended for example. The way our brains are wired makes us liable to do this kind of thing. The CHI+MED team working in the lab will study the causes of these capture errors and why they occur – what is going on in our heads to make us so likely to do it? Do interruptions, which are a fact of hospital life, make them more likely? Are they more likely if someone is trying to do two things at once, as nurses and doctors have to do? What kind of thing helps us avoid making this kind of mistake? Would different beeps at critical points be enough, or maybe having different looking buttons on the different devices? These experiments will involve people using the simulations of the devices doing real tasks, but in a lab.

The logicians involved in the project will meanwhile be creating mathematical models of the devices, of the human behaviour observed in the experiments and of the situations seen in the hospitals. They will run 'virtual experiments' where mathematical techniques are used to see what happens when the mathematical version of nurses use the mathematical

version of the model in mathematical hospitals. The difference here is that the maths can be used to explore all possible things that might happen when people behave like the models, not just those that do happen on a particular day. This can throw up yet new problems to study.

All of this work will be fed back to a team whose brief is to come up with new designs that avoid the problems discovered. The new designs will be investigated in all the same ways to ensure they really do fix things and don't just introduce new problems.

Of course the focus of all this activity won't be just on those two pumps from the original incident. The team in the hospitals will look for similar issues with other devices and that may lead to yet new

activity. They will also talk to the people in the hospitals responsible for buying new devices and find out why they make the decisions they do, so that in future they can more easily tell which devices may lead to problems.

The approach for CHI+MED that Ann envisages will thus improve safety by a scientific approach to understanding and designing out errors. Modern science and engineering aren't about single subjects but about teams of people bringing different expertise. People like Ann who are experts in lots of subjects are invaluable for seeing the big picture and to be able to come up with projects like CHI+MED in the first place. If she has her way avoidable medical mistakes from poor interaction design will be a thing of the past.



# How a computer scientist organised her wedding

In the weeks preceding their wedding, brides the world over face the same tricky problem: figuring out where to seat their guests for the meal. Most brides spend days looking at pieces of paper, penciling in guests' names at tables, then rubbing them out when that plan doesn't quite work. Computer scientist Karen Petrie, from the University of Dundee, had another solution for her wedding. She wrote a computer program to create her table plan. She tells us more.

I work in an area of computing called Constraint Programming. Problems often consist of choices, and making the best choice can be extremely difficult. Constraint programming is the branch of artificial intelligence where computers help us to make these choices.

A constraint programming problem consists of a series of choices, options for each choice and restrictions, or 'constraints' on those choices. A solution just allocates options to choices so that all of the constraints are satisfied. For table planning, for example, the problem was to place guests (options) at tables (choices), subject to constraints like "married couples should be placed together". Other constraints include that people of similar age should be placed at the same table, people must be seated with at least one other person they know, and people who don't get on should be separated.

Unfortunately my program came up with 195 valid table plans for my 150 guests! I know that most people would be happy that their friends and family were so friendly that they could be seated in so many positions with no possible arguments, but for me it was a headache. Which table plan was the best one to choose? I ended up reading through them all and doing some tweaks to come up with what I thought was the ideal solution. This led me to think that there must be a better way.

The experience inspired me to develop a system that will summarise the solutions of a constraint problem in an easy-to-understand and compact way. The new

system will tell users both what's the same about different solutions and, more importantly, point out all the differences. For table plans this means that users will be able to see which tables contain the same guests whatever the plan, and which guests can be moved between tables. They will then be able to choose the best table plan. Alternatively they can add more constraints on those guests that can be moved between tables, then find a new ideal plan.

However, that's not the end of the problem. We were lucky in planning our wedding in that there was a possible table plan. What if our guests had so many conflicting personalities that there weren't any plans that fulfilled all the constraints? Just tell the user that there are no possible solutions? That would hardly help an already stressed bride.

I'm therefore also working on a system to overcome the problem of having too many awkward guests. The way it works is to calculate the best possible plans: the ones where the most, if not all, constraints are met. The system then asks the user to decide which of the

constraints should not be fulfilled. With table plans that means it presents a small number of plans, telling the bride which constraints aren't met in each. She can then choose the table plan that best suits her needs.

I hope this system for dealing with cases where a problem does not have a single neat solution, in a user-friendly manner, will also work for other similar problems. That includes problems like rostering nurses, timetabling classes at schools and scheduling buses and trains. This would make constraint programming a useful technology in many aspects of our daily life.

The good news is that I successfully married Chris Jefferson, who is also a researcher in constraint programming, on the 5th of September, 2008. There were no complaints from guests about where they were sitting and no arguments. The guests did not realise their seats had been allocated by a computer program; although, it was obvious that they were at the wedding of computing enthusiasts by the cake.



Image: Karen Petrie

# Proof without words

Graphic news images often help sway public opinion. Images of famine in Africa led to LiveAid, a massive relief effort in 1985. Images from war zones of civilians can be disturbing enough that war leaders lose or gain political support as a result (depending on who did the bad stuff). Images can have far more power than words to argue a case, and to persuade.

Mathematical proofs are just arguments intended to persuade too. They aim to leave no element of doubt that some fact is true, not for emotional reasons but by logic. Mathematicians use mathematical notation – special symbols used in precise ways – to represent things in their proofs. That's just a way of making sure the arguments are precise, with no room for doubt. Sometimes that can make them seem arcane and difficult to follow, though that's only until you've learnt the mathematical language being used.

Mathematical proofs don't have to use words and symbols though. In fact people have been presenting proofs as pictures at least since the Ancient Greeks, and just as with news images a diagrammatic proof can be much more persuasive. Sometimes just by looking at a diagram the truth of a fact can become obvious.

For example, here is a mathematical 'fact' we might want to prove:

*"The square of any number is equal to the sum of consecutive odd numbers."*

That may sound a bit hairy. To get even hairier (if you aren't a mathematician), we can write this precisely in mathematical notation as:

$$n^2 = 1 + 3 + 5 + \dots + (2n - 1)$$

Don't worry about the notation though. Just look at the pictures. They show what we mean by the fact and should persuade you it's true.

The square of a number can be drawn as a picture of dots in a square. In other words one way to work out  $10^2$ , say, is to create a square of dots with sides of length 10 and then count the dots. That's why it's called 'squaring'!

One way to draw the dots to make up a square is as follows. First draw one dot in the corner, then draw three dots in an L-shape round it, then draw 5 dots in an L-shape round that...and keep going. Add a new L-shape (including the first dot) 10 times, say, once for each dot along the side, and you get a square of size 10 with 100 dots altogether. Notice that at every step you still have a square, though. Also notice that each L-shape is 2 dots bigger than the one before. That's because you can make it by adding one dot on the end of each arm of the last L-shape.

That means the number of dots in a square can be calculated by adding a sequence of odd numbers, one for each L-shape added:  $1 + 3 + 5 + \dots$ . As you add

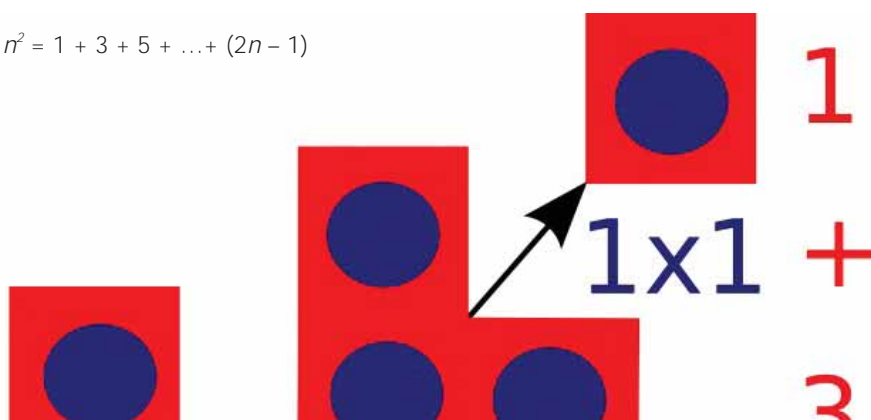
L-shapes you work up through squares of all sizes, so all squares can be made by adding odd numbers in this way.

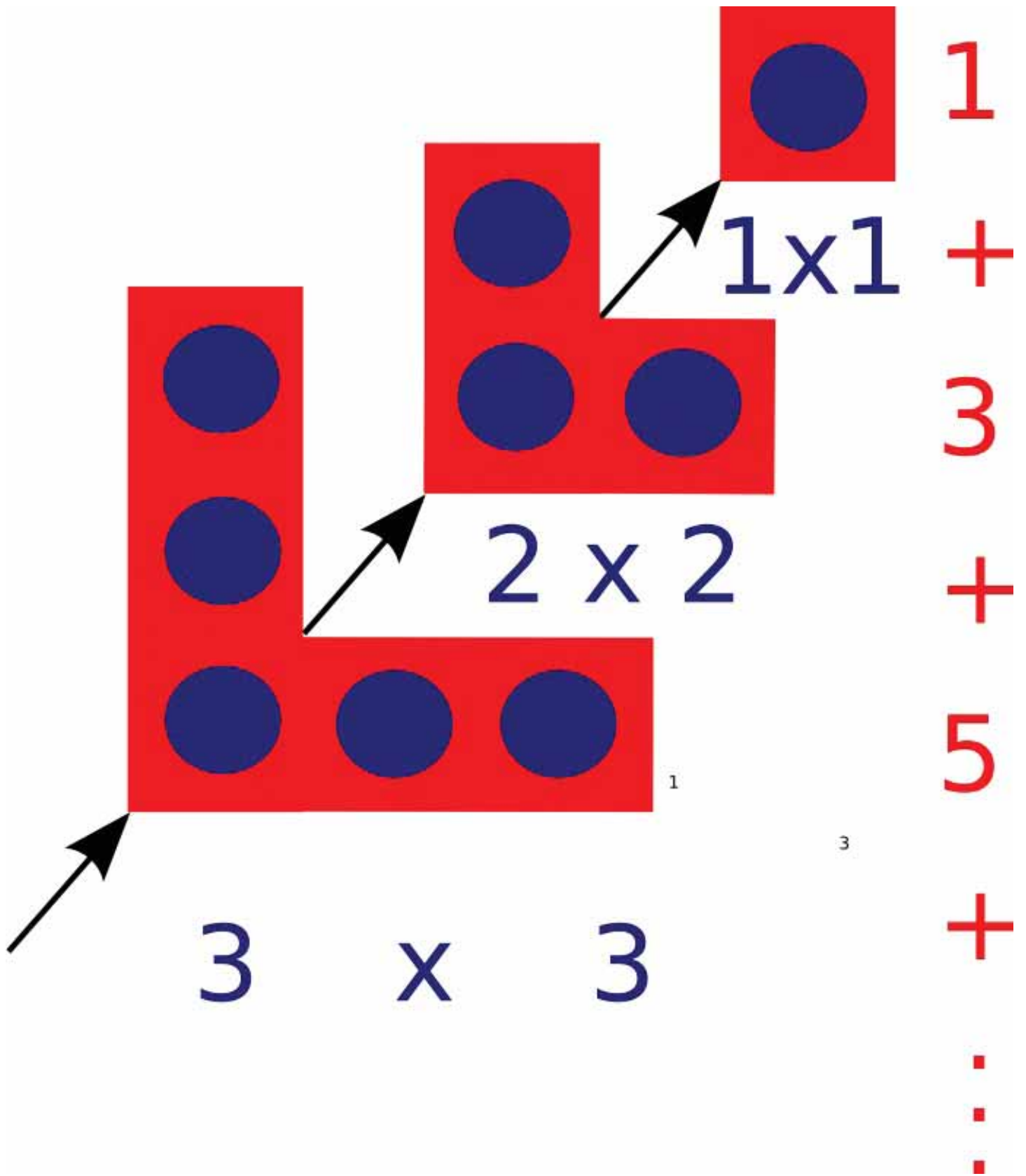
We've just explained it in words, but actually it's all in the picture, so it's possible to see without needing the words at all.

At least it may be possible for a person to see perhaps, but what about a computer? Could a computer 'see' a proof from a diagram? Computers are now very good at helping humans do logical proof using mathematical notation – after all they work themselves by pushing 'symbols' about and following rules blindly, which is all logical proof is. Seeing a proof in a diagram is different altogether though... or is it?

Mateja Jamnik, of the University of Cambridge has been tackling this problem. In fact her system, DIAMOND, can already check diagrammatic proofs created by a person to see if they really do convince. With DIAMOND you could, for example, take a series of L shape pictures like ours above and build them up step-by-step giving squares of different sizes. The system can then pull out the structure of this step-by-step proof and from that automatically obtain the equation that it proves.

DIAMOND needs a person to develop a diagrammatic proof for it to check. In the future, if Mateja has her way, the computers will be devising new diagrammatic proofs themselves that then convince we humans.





# *Designing an innovative radio*

Gulmina Rextina, Jayashree Sathyanarayanan and Lim LiShiang formed a team of computer science master's students. They took part in a role-play to design a new easy to use mobile digital radio. It was part of a role-play team project on an interactive systems design course at Queen Mary, University of London. Each team had three months to design a prototype and evaluate it to ensure that it really was easy-to-use. They then had to give a final presentation of their design. A final vote decided the winners of the Dragon's Den-style competition.

Whereas most other teams designed fairly conventional-looking radios that just aimed to be easy to use, Gulmina, Jayashree and Lim came up with an innovative headset radio that was also designed for partially sighted people to use.

*Here they tell us about the experience.*

## **How did you end up as a team?**

We had worked as part of a larger group before and really enjoyed it. We gelled then so knew we wanted to form a team this time. We all have different strengths so we could all contribute equally: one of us focusing on graphic design aspects, one coding and one the html for the prototype, for example. All three of us really enjoyed it. We wouldn't have worked so well together as a group if we hadn't.

When we ran into problems we would send out 'help' emails and then we would all work on the problems together. We've worked in groups before where some of the team didn't pull their weight, and just expected us to sort out the problems. When that happens it is really hard to get the work done and it ends up being a rush at the end. Because we all played our part this time we weren't in a panic. In fact we finished way before time, so we spent the last few days just sorting out details that made it all look more polished rather than rushing to get the core parts done.

## **Where did the idea come from?**

It was Lim that first suggested the headset idea. We didn't originally think we could fit all the controls on – we assumed we would need some other control, like a device on a wristband with a screen of some kind and more controls. It seemed like it was too

complicated to do it all with audio and it still be easy to use. Then we joined Tony Stockman [a Queen Mary Lecturer who is blind] in his group to do our final year projects. He was telling us that with no sight he found iPods hard to use – he had to rely on listening for the clicks. That was when we decided to do it with audio alone – though we still thought we might need an optional visual component that people could use if they wished.

## **How did the evaluation phase help?**

Doing evaluation with potential users was really helpful. We had fallen in love with our design from the start, but we weren't sure others would. It was important we didn't let our feelings get in the way if we had to change it because of the users' comments or other things we found from the evaluation.

One of the things that came out of the evaluation was the importance of putting buttons that were related together on the same side of the headset. We had originally been focusing too much on it just looking good, but doing the evaluation made us think about how it would really be used and that made us realise we needed to move those buttons together.





We found that IT people who acted as volunteers for user testing came up with a lot of good suggestions for new features. That is a way designers work – adapting ideas from other designs to fit the new context. We had to balance new features with the need for making it easy to use. That meant that some of the suggestions had to be rejected. A lot of the other groups' radios ordered the stations by most frequently used first. We stuck with the idea of just having favourite buttons though, as actually it keeps the design simple. We found that the younger people it is designed for wanted to be in control. Also when everything is done with audio, as our design, you needed to be clear where things were.

With some things, being 'usable' just depends on what you are used to. One user suggested having 'press and hold' to store something in a favourite. That would be hard to use on our headset design though. It would also be very hard for a first time user to work out if they hadn't come across it before. Instead we went for a separate button and made the design

store it in the next free space in its memory. The device then tells you where it has stored it. Our design always tells you what it is doing, which we thought was an important principle.

#### **Were you happy with the way the non-visual interface came out?**

It was a surprise really that we didn't need a visual interface at all. We were sceptical at the start, but then it started to make sense to us, though we still weren't sure other people would accept the idea. When you think about it, the normal visual interfaces are bombarding you with information constantly, but you don't really care about that information most of the time. Our design gives you information just when you want it. In most mobile situations, like when you are walking down the street, you can't really look at the information anyway, so most of the time it is useless.

We made a physical model as well as the computer prototype and that turned out to be really important as people could then understand the design. Understanding the

physical shape of it, including the way you can feel the different shapes of the buttons, and the way you wear it is important to understand how the design works. So is seeing how it also stands on the table when not being worn, so can then be used just as a normal radio. It is important to see that to take a view on whether it works as a design or not.

It's not just a gadget. It is intended as a piece of fashion too – which is why we included in the design the different colour choices so they could be personalised.

We kept our design secret from the other groups because we knew it was special. Most of the other groups were just doing designs of radios similar to those in the shops in looks. We didn't want any industrial espionage!

#### **What was the hardest part to get right?**

One of the hardest parts was getting the audio of the music in the prototype to pause when we wanted to insert some information (such as when the information

*Continued on page 58*





Gulmina Rextina, Jayashree Sathyanarayanan and Lim LiShiang

*Continued from page 57*

button was pressed) but then continuing the music from where it left off. The course lecturers told us that that was one of the features they were impressed with. In the early stages of the design they were sceptical about whether having audio information when the person was listening to music would work. They were impressed that we solved the technical problems to actually make it work in the prototype, but even more impressed that our prototype proved that the idea was really natural in practice.

#### **Did the competition role-play help?**

The competition part of the coursework really motivated us. We set ourselves the aim of winning the competition from the start. After the first class we decided we were going to win, though it was still a surprise at the end when we actually did. It was the class that decided rather than the lecturers. The whole class were able to vote and could vote for as many groups as they liked. At the end everyone voted for us. That was amazing.

We are really happy we took the course. It has been a really good experience and we have learnt things that will be really useful for real work. Our families are also very proud that we came up with the best design and won the competition.

You can evaluate the prototype of Gulmina, Jayashree and Lim's design yourselves at [www.cs4fn.org](http://www.cs4fn.org).



# ***Out of this world***

Evelyn Boyd Granville, who in 1949 became the second ever African American woman to earn a Maths PhD, worked on the computer programs for both the Mercury project, the first US mission to put people into space and the Apollo project that ultimately put Neil Armstrong on the Moon.



***Put in the effort,  
have some fun, and  
see how far YOU can go.***



## ***A gendered timeline of technology***

**2006 Fran Allen** is the first woman to win the Turing Award, which is considered the Nobel Prize of computer science, for work dating back to the 1950s. Allen says that she hopes that her award gives more "opportunities for women in science, computing and engineering". (See page 30)

**2006 Torchwood's technical expert** (it is the organisation protecting the Earth from alien invasion in the BBC's cult TV series) is not only a woman but also a quiet, highly intelligent computer genius. Fiction catches up with reality at last.

**2008 Barbara Liskov** wins the Turing Award for her work in the design of programming languages and object-oriented programming. This happens 40 years after she becomes the first woman in the US to be awarded a PhD in computer science. (See page 34)

**2009 Wendy Hall** is made a Dame Commander of the British Empire for her pioneering work on hypermedia and web science. (See page 47)

*See inside for more of  
our gendered timeline.*

This special issue was made possible due to support from the Westfield Trust

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

The University of Dundee and the Grenoble Institute of Technology contributed to this issue. For a full list of our university partners see [www.cs4fn.org](http://www.cs4fn.org)



**[www.cs4fn.org](http://www.cs4fn.org)**