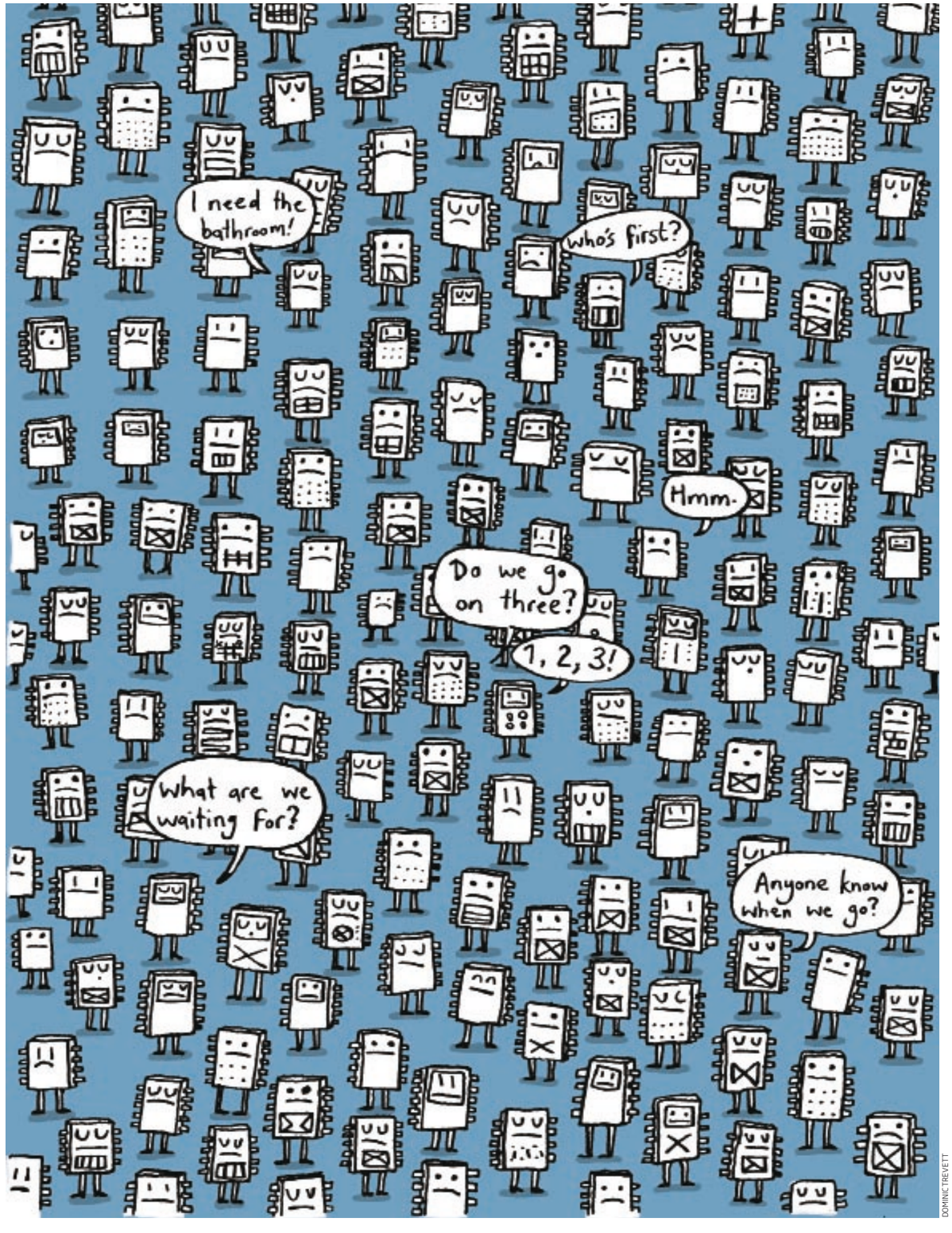


DOMINIC TREVETT

Parallel sparking

Multitasking is a breeze for the human brain, but a big problem for computers. The solution may trigger the second digital revolution, says Douglas Heaven

PETER BENTLEY is an optimist. In an office overlooking the rooftops of University College London, he is programming a computer that is incapable of running any existing software. Currently, he is struggling even to make it do basic arithmetic. But Bentley thinks his machine – or ones very much like it – could tear down 70 years of received wisdom and change the future of computing.

Bentley's machine is inspired by biological networks like the brain, in which billions of neurons are each wired up to thousands of others. "Every time you catch a ball, you're doing the equivalent of solving hugely complicated equations in parallel," he says. The use of many small nodes operating simultaneously seems to be an optimal way of doing computation – more robust, more efficient and lightning quick. If it's good enough for nature, why not a computer?

A fully realised artificial brain may still be a speck on the horizon, but we are already a few steps towards it. As computer chips grow in power, the basic way they are put together is being rethought. Instead of making faster processors, computer manufacturers are now putting more processors on a single chip. It's not quite an artificial brain, but it uses the same principle: that many hands make light work.

You may already own a "multicore" PC or smartphone with a handful of processors working in parallel, but the ultimate ambition is to yoke together tens of thousands, or even millions, of them. That could give our computers an unprecedented capacity for learning and decision-making: self-driving cars equipped with such chips, for instance, could keep a constant lookout for danger and react with quicker-than-human reflexes when something goes wrong. In the longer term, such computer resources could be used to simulate events, such as weather systems or military conflicts, allowing us to predict the future in minute detail.

To reach these goals, we will need to build entirely new programming tools that make the most of the technology. If they don't, software will stagnate and the redesign of microchips will have been for nothing. That's unless mavericks like Bentley depart from the foundations of computing with radical designs that bear little relation to anything that has gone before.

Almost all computers today rely on the principles that John von Neumann first scribbled down during a train journey in 1945. They are built from electrical "switches" known as transistors, which process data by controlling the flow of electrical current according to basic logical operations. Initially, transistors were a little over a centimetre wide, but they soon started shrinking according to a trend first noted in by 1965 Gordon Moore – a co-founder of chip manufacturer Intel. Moore's law states that the number of transistors that can be squeezed into a given size of circuit doubles every two years or so. Such exponential growth continued to drive the computing industry into the 21st century. Today, we have transistors that are just 22 nanometres wide – 4000 of them side by side would equal the width of a human hair.

Death of Moore's law

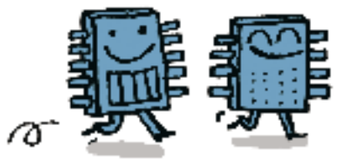
Crucially, more transistors meant more complex and faster chips, with processors that executed instructions quicker and quicker. So, for some time, Moore's law also served as a predictor of increasing processor speed. The number of instructions that a processor could handle in a second rocketed from around 200,000 in the first home computers of the 1980s to 100 billion in modern laptops.

But Moore's law has entered its dying days. Although transistors continue to shrink, the ›

Power trip

The computer processor has come a long way since the invention of the transistor. Moore's law has it that the number of transistors on a chip doubles every two years or so, due to constant miniaturisation, but problems with overheating now mean designers have to look for other ways to boost capability

1947 Transistor invented



1971 Intel 4004 chip packs in 2300 transistors to achieve 92,000 instructions per second

Mid-1970s First consumer computers, including Apple I and IBM 5100

1998 Following rapid increases in processor speeds in the 1990s, the first Apple iMac manages 525 million instructions per second

2000 Moore's law enters its twilight phase

2001 IBM releases POWER4, the first commercially available dual-core processor. It is capable of 11 billion instructions per second

2010/2011 Dual-core chips start to make their way into smartphones and tablets

2011 Multicore Intel chips with over 2 trillion transistors handle more than 100 billion instructions per second. Harnessing the cloud pulls together ~30,000 such chips into a parallel-processing monster (at a cost of $1300 an hour)

new chips are becoming so complex that they are running up against a fundamental limit: their ability to dissipate heat. "These chips have heat sinks and fans and all kinds of things," says Bentley. "But they still get so hot they melt themselves off their circuit boards."

The industry's response was simple. It doubled down. Instead of using twice as many transistors to build faster processors, it put two or more processors – or cores – on a single chip. The individual processors were not faster, and so did not create more heat, but they could run in parallel, executing multiple streams of instructions at once. Think of it in terms of a paint-by-numbers picture. With 10 friends, all wielding different brushes, it's going to take a lot less time than doing it by yourself. Until very recently, such parallelism was found only in supercomputers, one-offs built and maintained by experts to serve niche scientific purposes, but you will now find many mainstream computers with as many

"New hardware could fix many parallel computing bugs, but run up against the limits of programmers"

as eight cores. Even some smartphones make use of two or more processors for things like panorama photo apps, which stitch together images to build a panoramic view.

But if parallel thinking solved the problem of overheating, the resulting CPUs became a Pandora's box for software developers. Herb Sutter at Microsoft in Silicon Valley, California, was one of the first to see the danger. In 2004 he wrote an article, called *The Free Lunch Is Over*, that was a wake-up call for developers hoping to ride the wave of constantly improving hardware.

The problem is that only certain types of software can be easily pulled apart into parallel jobs. Image and video processing work well, since each core can be given a small set of pixels to work on. But most programs quickly run into bottlenecks as they share data. To extend the painting-by-numbers metaphor, it is as if one friend needs the green pen while another is still using it to colour in her part of the picture. Besides causing extended delays, that can create major crashes if both chips make a grab for the same piece of data at the same time.

Worse still, it is nearly impossible to debug these problems because they are so unpredictable – the problem might reveal itself only once in every ten operations. Such an undetected bug almost derailed the 1997 Mars Pathfinder mission shortly after the rover landed, for example. "The problem didn't arise in preflight testing because it only occurred in certain heavy-load usage scenarios which were not anticipated," says Paul Kelly at Imperial College London.

There may be ways around these problems. Kelly, for instance, is looking at ways to manage the data used to model the complex dynamics of fluids, such as ocean currents or air flowing over racing cars. These simulations create particular problems because they involve interdependent data – knowing the air flow in one volume of space is normally essential to work out the movements in the next. Kelly is working on tricks that will enable programmers to identify the "joints" in the data structure, revealing the best places to chunk the information without cutting too many ties.

Smart solutions like this will be essential as the plans for parallel processing become ever more ambitious. Multicore processing turned out to be only the first of four quite distinct types of parallelism. The second innovation is vector parallelism, where short sections of parallel code – typically quick-fire arithmetic operations – can be run on a single core. Some chips now have parallel cores each running parallel code; parallelism squared. Then there are chips that combine different types of core, which Sutter calls heterogeneous parallelism. Most cores are general-purpose, designed to run any kind of program. But some, such as graphics processing units (GPUs), are tuned for a specific type of computation. In theory, chips incorporating different cores treat programmers to a pick-and-mix selection of computing resources, letting them direct each part of their software to run on the type of core that best suits it.

Cloud power

But the ultimate source of parallelism could be the cloud. Sutter thinks that in a few years it will be possible to write code that runs on a phone but scales to an indefinite number of cores when it has an internet connection, effectively yoking together multiple machines into a single but vastly parallel chip. "Each might have multiple heterogeneous cores with vector units," says Sutter. "And I can have 10,000 of them." In a few years programmers could be working with an eye-watering million-fold parallelism.

Each revolution will be the source of yet more headaches for programmers. For a start, each type of parallelism has to be programmed in a different way. Coding for the cloud also brings many extra problems, like time lag and cores disappearing in the middle of a computation when a connection drops. Programmers will also have to unify their procedures to make sure that the same kinds of code will work no matter what parallelism is being used – be it multicore or the cloud.

As the problems mount, some people are beginning to believe the time is ripe for us to rethink the very foundations of computing. "If you were going to start from scratch and create a parallel computer, you wouldn't do it like this," says Bentley. He compares it to trying to make a train by gluing together lots of automobiles. It could work, he says, "but I'd rather have a train that was a train".

Those concerns put Bentley in good company. By the end of his life, von Neumann seemed to have turned from his groundbreaking work on the original computer to investigating parallel processing inspired by the brain.

Bentley's designs are also inspired by the dynamics of the brain and other biological systems. One of the central ideas is to decouple the tight communication between cores. In conventional parallel computers, the cores receive instructions in rapid succession, which can create hold-ups as they wait for data. Bentley's cores operate in a free-for-all, hooking up as and when needed, just as different brain regions communicate on an ad hoc basis. It sounds as if it shouldn't work, but because the cores do not share a common pool of data or instructions, there is less opportunity for a traffic jam, and many of the problems of mainstream multicore parallelism go away.

Bentley and his colleague Christos Sakellariou have already built a prototype, which they presented at the International Conference on Evolvable Systems in Singapore in April. It uses a special chip called a field-programmable gate array, with customisable logic circuits that can emulate the flexibility of a natural system.

Out of necessity, the system was cobbled together from what was available. Although it is not yet able to run advanced software, in simple speed tests it already outperforms a high-end consumer computer. "And we've just got some piddling little prototype," he says. "We haven't got the resources of Intel."

Bentley is not alone in turning to the principles of the brain. Steve Furber at the University of Manchester, UK, first made his name as a designer of the BBC Micro computer and later developed the ARM chip, versions of which are now found in over 90 per cent of smartphones. He is now better known for building an experimental parallel computer called SpiNNaker, in which the cores are highly interconnected, like neurons. And, like neural circuits, these flexible networks incorporate a certain amount of redundancy, meaning that the cores can busy themselves with other jobs instead of waiting for a message that has been delayed or lost.

A third system, by computing guru Jaron Lanier, who has advised Linden Labs on building Second Life and helped Microsoft develop the Kinect depth-sensing camera, takes a similar approach. The individual modules in his parallel system have a "virtual sense organ" that lets them read each other's processing and form internal models of their neighbours. It's a bit like the way ants in a colony perform their own task while keeping track of each other's movements. This should allow the computer to pre-empt processing bottlenecks and so avoid them.

Self-driving traffic

The researchers have big ambitions for their systems. Lanier, for instance, thinks it might aid the adoption of self-driving cars, since networks of automated vehicles will need to manage themselves safely and securely, while avoiding traffic jams. "This car coordination problem will be a massive real-time parallel computation," says Lanier. But the challenge will not be describing the mathematics of such a system. "It will be the logistical part, making sure all the parts talk to each other all the time," he says. "Complexity can just explode for that kind of stuff. That's where I think a different kind of hardware design would really help; one that's easier to understand."

Lanier also thinks that parallel computation will one day allow the real-time monitoring of the carbon footprint of a whole society, using millions of sensors. That could vastly improve the accuracy of climate predictions.

Despite their enthusiasm that these types of hardware will fix many of the bugs plaguing other types of parallel computing, some researchers fear that such computers will run up against the cognitive limits of their programmers. That is because, although different brain regions may act in parallel in the background, our communication has always been sequential, from language to mathematics and finally modern computer programs. Programmers often talk about "legacy systems" that hold up progress, and Bentley thinks our sequential communication might be the ultimate example. "It's a legacy system that is now thousands of years old."

It could be that overcoming that habit requires a certain style of thinking found in just a few people. Bentley points to certain mathematicians who can visualise problems in such a way that they seem to use different parts of their brain to do different bits of calculation simultaneously.

"You find these amazing people who do higher-level maths," says Bentley, "the Einsteins of the world, who can translate it and create their own notations to write it down." Interestingly, von Neumann may have been of this ilk. One of his colleagues observed that von Neumann seemed to have a brain of "a species superior to that of man".

Clearly, the invention of the computer required a very special mind. We will need many more of them as we take von Neumann's device through the next revolution. ■

Douglas Heaven is a writer based in London, UK