CLOSING IN ON THE PERFECT CODE

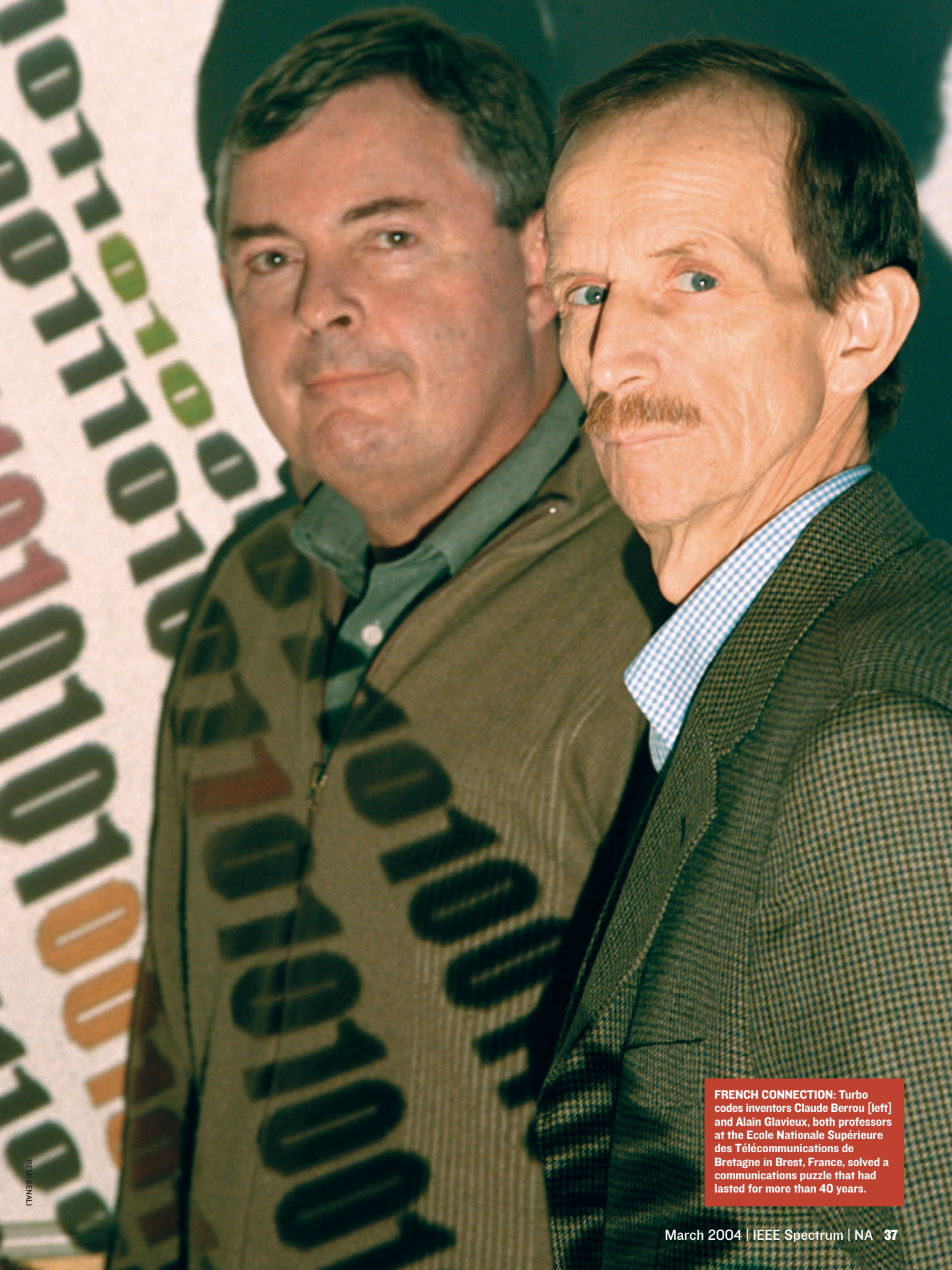
TURBO CODES, which let engineers pump far more error-free data through a channel, will be the key to the next generation of multimedia cellphones By Erico Guizzo

It's not often in the rarefied world of technological research that an esoteric paper is greeted with scoffing. It's even rarer that the paper proves in the end to be truly revolutionary.

It happened a decade ago at the 1993 IEEE International Conference on Communications in Geneva, Switzerland. Two French electrical engineers, Claude Berrou and Alain Glavieux, made a flabbergasting claim: they had invented a digital coding scheme that could provide virtually error-free communications at data rates and transmitting-power efficiencies well beyond what most experts thought possible.

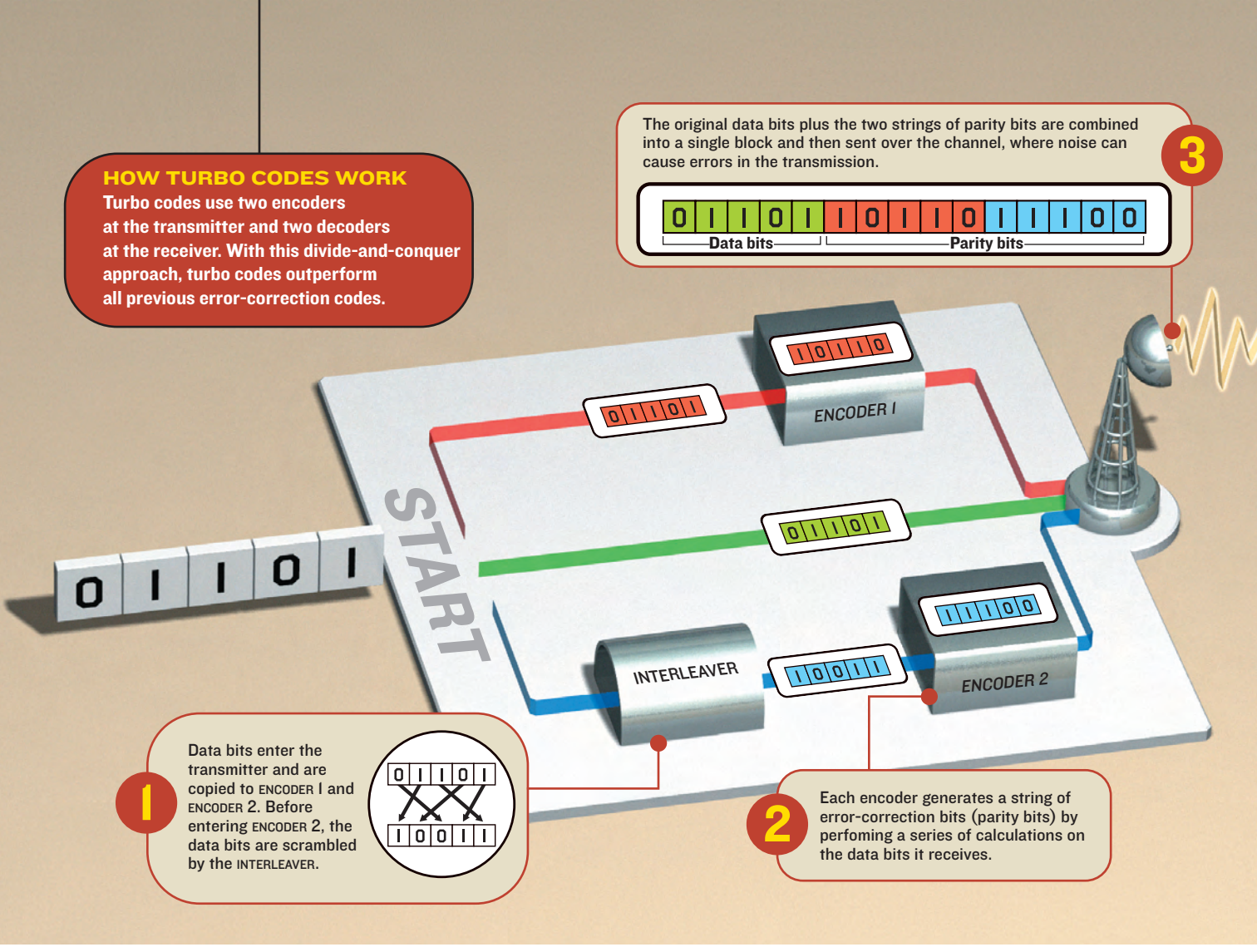
The scheme, the authors claimed, could double data throughput for a given transmitting power or, alternatively, achieve a specified communications data rate with half the transmitting energy—a tremendous gain that would be worth a fortune to communications companies.

Few veteran communications engineers believed the results. The Frenchmen, both professors in the electronics department at the Ecole Nationale Supérieure des Télécommunications de Bretagne in Brest, France, were then unknown in the information-theory community. They must have gone astray in their calculations,



FRENCH CONNECTION: Turbo codes inventors Claude Berrou [left] and Alain Glavieux, both professors at the Ecole Nationale Supérieure des Télécommunications de Bretagne in Brest, France, solved a communications puzzle that had lasted for more than 40 years.

REMI BENALI



some reasoned. The claims were so preposterous that many experts didn't even bother to read the paper.

Unbelievable as it seemed, it soon proved true, as other researchers began to replicate the results. Coding experts then realized the significance of that work. Berrou and Glavieux were right, and their error-correction coding scheme, which has since been dubbed turbo codes, has revolutionized error-correction coding. Chances are fairly good that the next cellphone you buy will have them built in.

From a niche technology first applied mainly in satellite links and in at least one deep-space communications system, turbo codes are about to go mainstream. As they are incorporated into the next-generation mobile telephone system, millions of people will soon have them literally in their hands. This coding scheme will let cellphones and other portable devices handle multimedia data such as video and graphics-rich imagery over the noisy channels typical of cellular communications. And researchers are studying the use of turbo codes for digital audio and video broadcasting, as well as for increasing data speeds in enhanced versions of Wi-Fi networks.

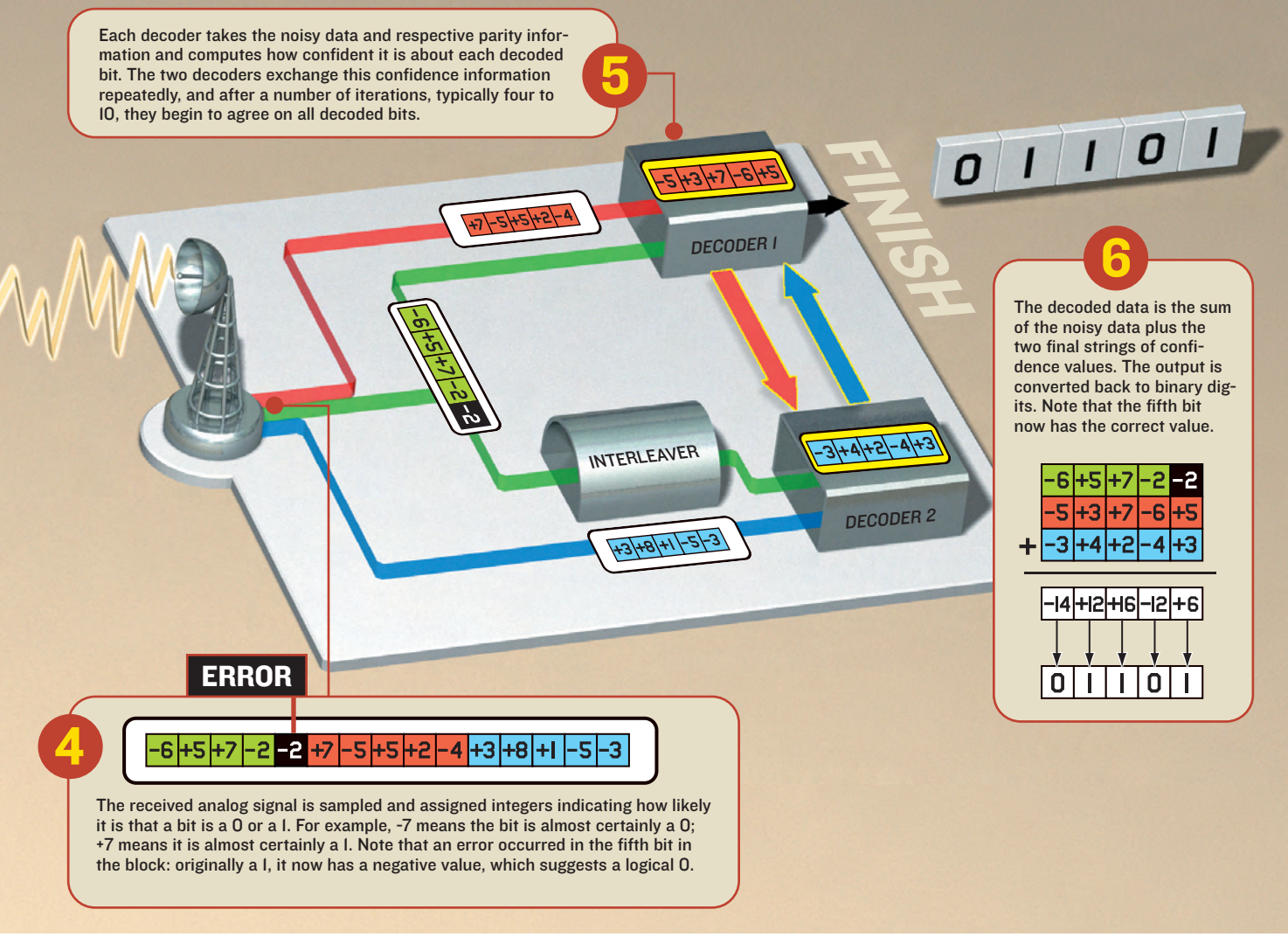
With possibilities like these, turbo codes have jumped to the forefront of communications research, with hundreds of groups working on them in companies and universities all over the world. The list includes telecommunications giants like France Télécom and NTT DoCoMo; high-tech heavyweights like Sony, NEC, Lucent, Samsung, Ericsson, Nokia, Motorola, and Qualcomm; hardware and chip manufacturers like Broadcom, Conexant, Comtech AHA, and STMicroelectronics; and start-ups like Turboconcept and iCoding.

Turbo codes do a simple but incredible thing: they let engineers design systems that come extremely close to the so-called channel capacity—the absolute maximum capacity, in bits per second, of a communications channel for a given power level at the transmitter. This threshold for reliable communications was discovered by the famed Claude Shannon, the brilliant electrical engineer and mathematician who worked at Bell Telephone Laboratories in Murray Hill, N.J., and is renowned as the father of information theory [see sidebar, "Shannon: Cracking the Channel"].

In a landmark 1948 paper, Shannon, who died in 2001, showed that with the right error-correction codes, data could be transmitted at speeds up to the channel capacity, virtually free from errors, and with surprisingly low transmitting power. Before Shannon's work, engineers thought that to reduce communications errors, it was necessary to increase transmission power or to send the same message repeatedly—much as when, in a crowded pub, you have to shout for a beer several times.

Shannon basically showed it wasn't necessary to waste so much energy and time if you had the right coding schemes. After his discovery, the field of coding theory thrived, and researchers developed fairly good codes. But still, before turbo codes, even the best codes usually required more than twice the transmitting power that Shannon's law said was necessary to reach a certain level of reliability—a huge waste of energy. The gap between the practical and the ideal, measured in decibels—a ratio between the signal level and the noise level on a logarithmic scale—was about

JOHN MACNEILL



3.5 dB. To chip away at it, engineers needed more elaborate codes.

That was the goal that persisted for more than four decades, until Berrou and Glavieux made their discovery in the early 1990s. When they introduced turbo codes in 1993, they showed it was possible to get within an astonishing 0.5 dB of the Shannon limit, for a bit-error rate of one in 100 000. Today, turbo codes are still chipping away at even that small gap.

The solution to overcoming the noise that plagued all communications channels, according to Shannon's seminal paper, was to divide the data into strings of bits and add to each string a set of extra bits—called parity bits—that would help identify and correct errors at the receiving end. The resulting group of bits—the data bits plus the parity bits—is called a codeword, and typically it represents a block of characters, a few image pixels, a sample of voice, or some other piece of data.

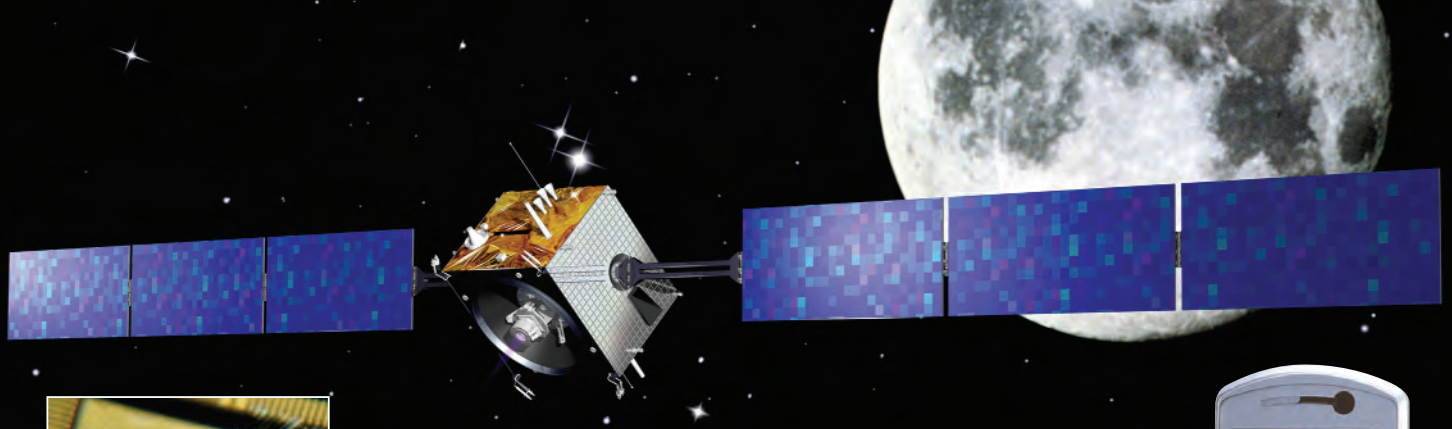
Shannon showed that with the right collection of codewords—with the right code, in other words—it was possible to attain the channel capacity. But then, which code could do it? "Shannon left unanswered the question of inventing codes," says David Forney, a professor of electrical engineering at the Cambridge-based Massachusetts Institute of Technology (MIT) and an IEEE Fellow. Shannon proved mathematically that coding was the means to reach capacity, but he didn't show exactly how to construct these capacity-approaching codes. His work, nevertheless, contained valuable clues.

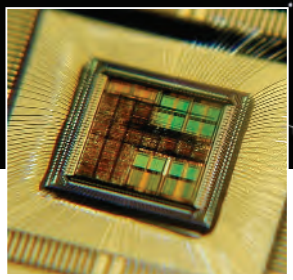
Shannon thought of codewords as points in space. For example, the codeword 011 can be considered a point in a three-dimensional space with coordinates $x = 0$, $y = 1$, and $z = 1$. Codewords with more than three bits are points in hyperspace. Noise can alter a codeword's bits, and therefore its coordinates, displacing the point in space. If two points are close to each other and one is affected by noise, this point might fall exactly onto the other, resulting in decoding error. Therefore, the larger the differences in codewords—the farther apart they are—the more difficult it is for noise to cause errors.

To achieve capacity, Shannon demonstrated that you should randomly choose infinitely long codewords. In other words, going back to his spatial analogy, if you could make the codewords both random and as long as you wanted, you could put the points arbitrarily far from each other in space. There would be essentially no possibility of one point erroneously falling on another. Unfortunately, such long, random codes are not practical: first, because there is an astronomical number of codewords; second, because this code would be extremely slow to use as you transmitted many, many bits for just one codeword. Still, the random nature of a good code would turn out to be critical for turbo codes.

Coding experts put aside Shannon's ideal random codes, as they concentrated on developing practical codes that could be implemented in real systems. They soon began to develop good codes by cleverly choosing parity bits that constrained codewords to certain values, making these codewords unlikely to be confused with other ones.

For example, suppose we have an eight-bit codeword (seven data bits plus one parity bit). Suppose we further insist that all the codewords have an even number of 1s, making that extra par-





OUT OF THE LAB: Europe's SMART-1 [top] is the first probe in space using turbo codes. In Japan, 3-G cellphones [right] use turbo codes to transmit pictures, video, and e-mail. Bell Labs' turbo decoder chip [left] promises to boost 3-G data rates from the current 384 kb/s to 20 Mb/s in upcoming networks.

ity bit a 1 if necessary to fulfill that requirement. Now, if any of the eight bits is altered by noise, including the parity bit itself, the receiver knows there was an error, because the parity count won't check—there would be an odd number of 1s.

This basic scheme can detect an error, but it can't correct it—you don't know which bit was flipped. To correct errors, you need more parity bits. Coding experts have come up with numerous and ever more sophisticated ways of generating parity bits. Block codes, Hamming codes, Reed-Solomon codes, and convolutional codes are widely used and achieve very low error rates.

Nevertheless, a computational-complexity problem hounded coding specialists and plagued all these codes. The complexity problem emerges as you figure the cost of a code in terms of the amount of computation required to decode your data. The closer you get to Shannon's limit, the more complicated this process becomes, because you need more parity bits and the codewords get longer and longer.

For codewords with just 3 bits, for instance, you have a total of only 2^3, or 8, codewords. To approach capacity, however, you might need codewords with, say, 1000 bits, and therefore your decoder would need to search through an unimaginably large collection of 2^{1000}—approximately 10^{301}—codewords. For comparison, the estimated number of atoms in the visible universe is about 10^{80}.

The upshot was that if you set about exploiting the best existing codes as your strategy for achieving arbitrarily reliable communications at Shannon's limit, you would be doomed to failure. "The computational complexity is just astronomical," says IEEE Fellow R. Michael Tanner, a professor of electrical and computer engineering and provost at the University of Illinois at Chicago. "These codes don't have the capability to do it." How could researchers get past this barrier? It was hopeless, some actually concluded in the late 1970s.

Turbo codes solved the complexity problem by splitting it into more manageable components. Instead of a single encoder at the transmitter and a single decoder at the receiver, turbo codes use two encoders at one end and two decoders at the other [see illustration, "How Turbo Codes Work"].

Researchers had realized in the late 1960s that passing data through two encoders in series could improve the error-resistance capability of a transmission—for such a combination of encoders, the whole is more than the sum of the parts. Turbo codes employ two encoders working synergistically—not in series, but in parallel.

The turbo process starts with three copies of the data block to be transmitted. The first copy goes into one of the encoders, where a convolutional code takes the data bits and computes parity bits from them. The second copy goes to the second encoder, which contains an identical convolutional code. This second encoder gets not the original string of bits but rather a string with the bits in another order, scrambled by a system called an interleaver. This encoder then reads these scrambled data bits and computes parity bits from them. Finally, the transmitter takes the third copy of the original data and sends it, along with the two strings of parity bits, over the channel.

That rearranging of the bits in the interleaver is the key step in the whole process. Basically, this permutation brings more diversity to the codewords; in the spatial analogy, it pushes the points farther apart in space. "The role of the permutation is to introduce some random behavior in the code," says Berrou. In other words, the interleaver adds a random character to the transmitted information, much as Shannon's random codes would do.

But then turbo codes, like any other code with a huge number of codewords, would also hit the wall of computational complexity. In fact, turbo codes usually work with codewords having around a thousand bits, a fairly unwieldy number. Hopeless? Yes, if you had a single decoder at the receiver. But turbo codes use two component decoders that work together to bypass the complexity problem.

The role of each decoder is to get the data, which might have been corrupted by noise along the channel, and decide which is the more likely value, 0 or 1, for each individual bit. In a sense, deciding about the value of each bit is as if you had to guess whether it's raining or not outside. Suppose you can't look out a window and you don't hear any sounds; in this case, you basically have no clue, and you can simply flip a coin and make your guess. But what if you check the forecast and it calls for rain? Also, what if you suddenly hear thunder? These events affect your guess. Now you can do better than merely flipping a coin; you'll probably say there's a good chance that it is raining and you will take your umbrella with you.

Each turbo decoder also counts on "clues" that help it guess whether a received bit is a 0 or a 1. First, it inspects the analog signal level of the received bits. While many decoding schemes transform the received signal into either a 0 or a 1—therefore throwing away valuable information, because the analog signal has fluctuations that can tell us more about each bit—a turbo decoder transforms the signal into integers that measure how confident we can be that

LEFT TO RIGHT: BELL LABS, ESA, DOCOMO

a bit is a 0 or a 1. In addition, the decoder looks at its parity bits, which tell it whether the received data seems intact or has errors.

The result of this analysis is essentially an informed guess for each bit. "What turbo codes do internally is to come up with bit decisions along with reliabilities that the bit decisions are correct," says David Garrett, a researcher in the wireless research laboratory at Bell Labs, part of Lucent Technologies, Murray Hill, N.J. These bit reliabilities are expressed as numbers, called log-likelihood ratios, that can vary, for instance, between -7 and +7. A ratio of +7 means the decoder is almost completely sure the bit is a 1; a -5 means the decoder thinks the bit is a 0 but is not totally convinced. (Real systems usually have larger intervals, like -127 to +127.)

Even though the signal level and parity checks are helpful clues, they are not enough. A single decoder still can't always make correct decisions on the transmitted bits and often will come up with a wrong string of bits—the decoder is lost in a universe of codewords, and the codeword it chooses as the decoded data is not always the right one. That's why a decoder alone can't do the job.

But it turns out that the reliability information of one decoder is useful to the other and vice versa, because the two strings of parity bits refer to the very same data; it's just that the bits are arranged in a different order. So the two decoders are trying to solve the same problem but looking at it from different perspectives.

The two decoders, then, can exchange reliability information in an iterative way to improve their own decoding. All they have to do, before swapping reliability strings, is arrange the strings' content in the order each decoder needs. So a bit that was strongly detected as a 1 in one decoder, for example, influences the other decoder's opinion on the corresponding bit.

In the rain analogy, imagine you see a colleague going outside carrying an umbrella. It's a valuable additional piece of information that would affect your guess. In the case of the turbo decoders, now each decoder not only has its own "opinion," it also has an "external opinion" to help it come up with a decision about each bit. "It's as if a genie had given you that information," says Gerhard Kramer, a researcher in the mathematical sciences research center at Bell Labs. This genie whispers in your ear how confident you should be about a bit's being a 1 or a 0, he says, and that helps you decode that bit.

At the heart of turbo coding is this iterative process, in which each component decoder takes advantage of the work of the other at a previous decoding step. After a certain number of iterations, typically four to 10, both decoders begin to agree on all bits. That means the decoders are not lost anymore in a universe of codewords; they have overcome the complexity barrier.

"It's a divide-and-conquer solution," says Robert J. McEliece, a professor of electrical engineering at the California Institute of Technology, in Pasadena, and an IEEE Fellow. "It broke the problem into two smaller pieces, solved the pieces, and then put the pieces back together."

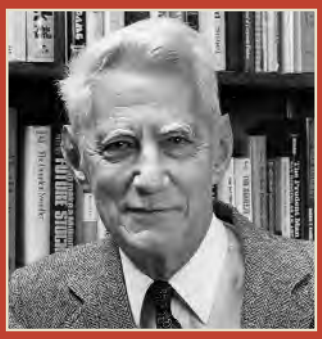
Another way of thinking about the turbo decoding process is in terms of crossword puzzles, Berrou says. Imagine that Alice solved a crossword and wanted to send the solution to Bob. Over a noiseless channel, it would be enough to send the array with the words. But over a noisy channel, the letters in the array are messed up by noise. When Bob receives the crossword, many words don't make sense. To help Bob correct the errors, Alice can send him the clues for the horizontal and vertical words. This is redundant information, since the crossword is already solved, but it nevertheless helps Bob, because, as with parity bits, it imposes constraints on the words that can be put into the array. It's a problem with two dimensions: solving the rows helps to solve the columns and vice versa, like one decoder helping the other in the turbo-decoding scheme.

Flash back 11 years as an amused 42-year-old Berrou wanders the corridors of the convention center in Geneva, peeking over the shoulders of other attendees and seeing many of them trying to understand his paper. At the presentation, young Ph.D. students and a scattering of coding veterans pack the auditorium, with people standing by the door. When Berrou and Glavieux finish, many surround them to request more explanations or simply to shake their hands.

Still, convincing the skeptics that the work had no giant overlooked error took time. "Because the foundation of digital communications relied on potent mathematical considerations," Berrou recollected later, "error-correcting codes were believed to belong solely to the world of mathematics."

What led Berrou and Glavieux to their important breakthrough was not some esoteric theorem but the struggle to solve real-world problems in telecommunications. In the late 1980s, when they began to work on coding schemes, they were surprised that an important concept in electronics—feedback—was not used in digital receivers.

SHANNON: CRACKING THE CHANNEL



BELL LABS

In 1948, Claude Shannon, then a young engineer working at Bell Telephone Laboratories in Murray Hill, N.J., published a landmark paper titled "A Mathematical Theory of Communication."

In that paper, Shannon defined what the once fuzzy concept of "information" meant for communications engineers and proposed a precise way to quantify it: in his theory, the fundamental unit of information is the bit.

Shannon showed that every communications channel has a maximum rate for reliable data transmission, which he called the channel capacity, measured in bits per second. He demonstrated that by using certain coding schemes, you could transmit data up to the channel's full capacity, virtually free of errors—an astonishing result that surprised engineers at the time.

"I can't think of anybody who could ever have guessed that such a theory existed," says Robert Fano, an emeritus professor of computer science at the Massachusetts Institute of Technology, in Cambridge, and a pioneer in the information theory field. "It's just an intellectual jump; it's very profound."

The channel capacity became an essential benchmark for communications engineers, a measure of what a system can and cannot do, expressed in many cases by the famous formula, $C = W \log_2 (1 + P/N)$. In the formula, C is the capacity in bits per second, W is the bandwidth in hertz, P is the transmitter power in watts, and N is the noise power, also in watts.

From space probes to cellphones and CD players, Shannon's ideas are invisibly embedded in the digital technologies that make our lives more interesting and comfortable.

A tinkerer, juggling enthusiast, and exceptional chess player, Shannon was also famous for riding the halls of Bell Labs on a unicycle. He died on 24 February 2001, at age 84, after a long battle with Alzheimer's disease. *—E.G.*

In amplifiers, a sample of the output signal is routinely fed back to the input to ensure stable performance. Berrou and Glavieux wondered, why shouldn't it work for coding as well?

They ran the first experiments with their novel coding scheme in 1991 using computer simulations, and when the results came out, they were stunned. "Every day I asked myself about the possible errors in the program," says Berrou.

The first thing Berrou and Glavieux did after confirming that their results were correct was to patent the invention in France, Europe, and the United States. At the time, France Télécom was the major sponsor of their work, so the French company took possession of the turbo code patents. The inventors and their institution, however, share part of the licensing profits. (Turbo codes were not patented in Asia, where they can therefore be used for free.)

It was France Télécom that asked Berrou to come up with a commercial name for the invention. He found the name when one day, watching a car race on TV, he noticed that the newly invented code used the output of the decoders to improve the decoding process, much as a turbocharger uses its exhaust to force air into the engine and boost combustion. Voilà: "turbo codes"!

Turbo codes are already in use in Japan, where they have been incorporated into the standards for third-generation mobile phone systems, known officially as the Universal Mobile Telecommunications System (UMTS). Turbo codes are used for pictures, video, and mail transmissions, says Hirohito Suda, director of the Radio Signal Processing Laboratory at NTT DoCoMo, in Yokosuka, Japan. For voice transmission, however, convolutional codes are used, because their decoding delays are smaller than those of turbo codes.

In fact, the decoding delay—the time it takes to decode the data—is a major drawback to turbo codes. The several iterations required by turbo decoding make the delay unacceptable for real-time voice communications and other applications that require instant data processing, like hard disk storage and optical transmission.

For systems that can tolerate decoding delays, like deep-space communications, turbo codes have become an attractive option. In fact, last September, the European Space Agency, based in Paris, France, launched SMART-1, the first probe to go into space with data transmission powered by turbo codes. ESA will also use the codes on other missions, such as Rosetta, scheduled for launch early this year to rendezvous with a comet. The National Aeronautics and Space Administration, in Washington, D.C., is also planning missions that will depend on turbo codes to boost reliable communications. "The first missions that will be using these codes will be Mars Reconnaissance Orbiter and Messenger," says Fabrizio Pollara, deputy manager of the communications systems and research section at NASA's Jet Propulsion Laboratory in Pasadena, Calif.

Digital audio broadcasting, which provides CD-quality radio programs, and satellite links, such as the new Global Area Network of Inmarsat Ltd., in London, are both also about to incorporate turbo codes into their systems.

And beyond error correction, turbo codes—or the so-called turbo principle—are also helping engineers solve a number of communications problems. "The turbo-coding idea sparked lots of other ideas," says Lajos Hanzo, a professor in the School of Electronics and Computer Science at the University of Southampton, United Kingdom, and an IEEE Fellow. One example is in trying to mitigate the effects of multipath propagation—that is, signal distortion that occurs when you receive multiple replicas of a signal that bounced off different surfaces. Turbo codes may eventually help portable devices solve this major limitation of mobile telephony.

Finally, another major impact of turbo codes has been to make researchers realize that other capacity-approaching codes existed. In fact, an alternative that has been given a new lease on life is low-density parity check (LDPC) codes, invented in the early 1960s by Robert Gallager at MIT but largely forgotten since then. "In the 1960s and 1970s, there was a very good reason why nobody paid any attention to LDPC codes," says MIT's Forney. "They were clearly far too complicated for the technology of the time."

Like turbo codes, LDPC attains capacity by means of an iterative decoding process, but these codes are considerably different from turbo codes. Now researchers have implemented LDPC codes so that they actually outperform turbo codes and get even closer to the Shannon limit. Indeed, they might prove a serious competitor to turbo codes, especially for next-generation wireless network standards, like IEEE 802.11 and IEEE 802.16. "LDPC codes are using many of the same general ideas [as turbo codes]," says Caltech's McEliece. "But in certain ways, they are even easier to analyze and easier to implement." Another advantage, perhaps the biggest of all, is that the LDPC patents have expired, so companies can use them without having to pay for intellectual-property rights.

Turbo codes put an end to a search that lasted for more than 40 years. "It's remarkable, because there's this revolution, and nowadays if you can't get close to Shannon capacity, what's wrong with you?" says the University of Illinois's Tanner. "Anybody can get close to the Shannon capacity, but let's talk about how much faster your code goes...and if you are 0.1 dB from Shannon or 0.001 dB."

It was the insight and naiveté typical of outsiders that helped Berrou and Glavieux realize what the coding theory community was missing. "Turbo codes are the result of an empirical, painstaking construction of a global coding/decoding scheme, using existing bricks that had never been put together in this way before," they wrote a few years ago.

Berrou says their work is proof that it is not always necessary to know about theoretical limits to be able to reach them. "To recall a famous joke, at least in France," he says, "the simpleton didn't know the task was impossible, so he did it." ■

Beyond error correction, turbo codes are helping MOBILE DEVICES ACHIEVE BETTER RECEPTION

TO PROBE FURTHER

The 2004 International Conference on Communications, to be held in Paris on 20–24 June, will include several sessions on turbo codes. See http://www.icc2004.org/.

"What a Wonderful Turbo World," an electronic book by Adrian Barbulescu, contains a detailed analysis of turbo codes and source code in C for simulations. See http://people.myoffice.net.au/~abarbulescu/.

For a discussion of implementation issues and a presentation of a real-life prototype, see *Turbo Codes: Desirable and Designable*, by A. Giulietti, B. Bougard, and L. Van der Perre (Kluwer Academic, Dordrecht, the Netherlands, 2004).