process extension, and how is the megakaryocyte submembrane cytoskeleton disrupted to allow proplatelet protrusion? Does iron deficiency or inflammation change platelet size and number by affecting proplatelet formation? And because proplatelets issue only from large polyploid megakaryocytes, what are the mechanisms that uncouple megakaryocyte DNA replication and cell division, leading to polyploidy? The elegant intravital look at platelet formation provided by Junt and colleagues has only increased our appetite to more fully understand thrombopoiesis, and in so doing, potentially intervene in the process for therapeutic benefit of patients who lack an adequate homeostatic defense.

References
1. T. Junt et al., Science 317, 1767 (2007).
2. D. Metcalf et al., Proc. Natl. Acad. Sci. U.S.A. 72, 1744 (1975).
3. W. Vainchenker et al., Blood 54, 940 1979.
4. D. I. Martin et al., Nature 344, 444 (1990).
5. P. H. Romeo et al., Nature 344, 447 (1990).
6. F. J. de Sauvage et al., Nature 369, 533 (1994).
7. K. Kaushansky et al., Nature 369, 568 (1994).
8. D. Zucker-Franklin, S. Petursson, J Cell Biol. 99, 390 (1984).
9. R. P. Becker, P. P. De Bruyn, Am. J. Anat. 145, 183 (1976).
10. M. Tavassoli, M. Aoki, Blood Cells 15, 3 (1989).
11. E. S. Choi et al., Blood 85, 402 (1995).
12. H. Schulze et al., Blood 107, 3868 (2006).
13. J. E. Italiano Jr., P. Lecine, R. A. Shivdasani, J. H. Hartwig, J. Cell Biol. 147, 1299 (1999).
14. R. M. Leven, F. Tablin, Exp. Hematol. 20, 1316 (1992).
15. M. K. Larson, S. P. Watson, Blood 108, 1509 (2006).
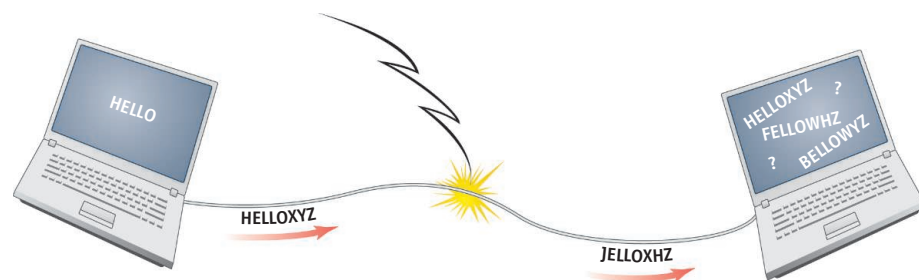
10.1126/science.1148946

COMPUTER SCIENCE

Coding and Computing Join Forces

Researchers who develop theories of computation are using their insights to create improved error correction codes.

Bernard Chazelle

Unlike vinyl recordings of yore, today's CDs and DVDs are impervious to the feral assaults of even the most determined toddler. For this triumph of civilization over savagery, we owe thanks to coding theory, one of the crown jewels of the digital era. Since Claude Shannon's pioneering work in the mid-20th century, error-correcting codes are found in all manner of communication devices—so much so, in fact, that by the 1990s coding theorists began to wonder if their brightest days had not passed. However, recent developments highlight a remarkable confluence of coding and computing, which may herald the shape of things to come.

Imagine Bob communicating with Alice the 21st-century way, text messaging. Complying with her request, Bob texted Alice his age: 48. But one digit was garbled, and 28 is what she got. It might have been wise of Bob to add at least one redundant symbol so she could spot the corruption of any single digit, sparing him much grief. For example, Bob could take the sum of the digits modulo 10 [i.e., the remainder of $(4 + 8) = 12$ divided by 10, which equals 2]. Adding the "2" gives the new message "482" and Alice could detect the error by doing the same calculation. In fact, by tagging on more symbols to his message, Bob could have enabled his friend to recover it in the presence of one or more garbled digits.

If Alice is to have any chance of restoring Bob's message if corrupted by e errors, Bob would need to inject at least $2e$ redundant symbols. Courtesy of so-called Reed-Solomon codes, this is sufficient. On the downside, it requires complex computations. Yet the added

The author is in the Department of Computer Science, Princeton University, Princeton, NJ 08540, USA. E-mail: chazelle@cs.princeton.edu

Safe transit. An example of list decoding. Additional information ("xyz") is appended to a message ("hello") sent through a noisy channel, enabling the receiver to check the message's accuracy. The garbled message is compared against a list of all possible encoded messages bearing close similarity to the one received ("jelloxhz").

complication has not stopped these codes from becoming the world's most popular.

If the message Alice receives differs from the one Bob sent in more than e places, the decoding may not be unique. Back at the dawn of coding theory, Elias observed (1) that such ambiguity is unlikely and saw in this an opportunity. Why not let Alice set a parameter $E > e$ of her choice and reconstruct a list of all the messages Bob might have written that could have produced the received message in the presence of up to E errors? If the transmission noise stays within this bound, obviously Alice's list will include his message and it will just be a matter of picking it out from the crowd. To do that, she might choose the message whose encoding matches the received message most closely; or her preference might go to the message she deems most likely to be Bob's.

The success of "list decoding," as Elias's suggestion is called, hinges on the shortness of the list (as a function of E) and the ease of collecting it. In two breakthrough papers, Sudan (2) and Guruswami and Sudan (3) showed how to list-decode Reed-Solomon codes efficiently. And here the word "efficiently" is everything, for decoding is trivial when time is not an issue. Sudan and Guruswami's key

insight was to trade single-variable polynomials, the natural habitat of Reed-Solomon, for the two-variable kind. This then led to a series of improvements by Parvaresh and Vardy (4) and Guruswami and Rudra (5).

What were Sudan and Guruswami, two theoretical computer scientists, doing on the stomping grounds of coding theorists? List decoding can function as a tool for changing one computational problem into another. Suppose that I compose a message that enumerates the solutions of a given problem for all possible inputs of a certain length. With list decoding, my message can be recovered from its encoded version even if 99% of its information has been destroyed. Until the work of these researchers (2–5), a message more than 50% garbled could not be recovered.

Thinking now of the encoded message as another problem's "solution sheet," we conclude that even an algorithm so bad that it solves the new "problem" erroneously 99% of the time can be used to recover the message correctly, and hence solve the original problem, all of the time. Conversely, a problem known to be hard on just a few inputs can be transformed into one that is hard on nearly all of them. Strange as

it may seem, cryptographers crave such problems because hard problems can be used to create difficult-to-break encryption.

In some cases, Alice may need to recover, say, the 217th bit of the message Bob intended without having to read all of the message she received. For this we turn to "locally decodable" codes and, specifically, to a remarkable recent result of Yekhanin (6): Bob's message can be encoded so that, should a small enough fraction of its symbols die in transit, Alice would still be able, with high probability, to recover the original bit anywhere in the message she chooses. The surprise: She can do it by picking at random a mere three bits of the received message and combining them the right way.

The randomness of the three single-bit lookups makes locally decodable codes ideally suited for private information retrieval. The concept was introduced by Chor *et al*. (7) to allow users of a database to make queries without divulging what they are. Yekhanin's scheme would keep the anonymity of a query by breaking it down into three subqueries and passing on each one to a separate copy of the database. Individually, each subquery would look random and therefore unrelated to the parent query.

Computing theorists have been borrowing from coding theory for decades. Recently they have begun to return the favor. This symbiotic relationship, it is safe to predict, is far from having run its course. The quest for a practical solution to private information retrieval is still wide open. How to turn the beautiful mathematics of local decoding into working privacy tools is one of the main challenges ahead.

References
1. P. Elias, *List Decoding for Noisy Channels*, Research Laboratory of Electronics Technical Report 335 (Massachusetts Institute of Technology, Cambridge, MA, 1957).
2. M. Sudan, *J. Complexity* **13**, 180 (1997).
3. V. Guruswami, M. Sudan, *IEEE Trans. Inform. Theory* **45**, 1757 (1999).
4. F. Parvaresh, A. Vardy, in *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science* (IEEE Computer Society, Los Alamitos, CA, 2005), pp. 285–294.
5. V. Guruswami, A. Rudra, in *Proceedings of the 38th Annual ACM Symposium on Theory of Computing* (Association of Computing Machinery, New York, 2006), pp. 1–10.
6. S. Yekhanin, in *Proceedings of the 39th Annual ACM Symposium on Theory of Computing* (Association of Computing Machinery, New York, 2007), pp. 266–274.
7. B. Chor, O. Goldreich, E. Kushilevitz, M. Sudan, *J. Assoc. Comp. Machinery* **45**, 965 (1998).

ARCHAEOLOGY

Easter Island Revisited

New information about Easter Island is helping to identify the cause of the massive deforestation that occurred prior to European arrival, but unanswered questions remain.

Jared Diamond

Easter Island is famous for many reasons: its remoteness, its total deforestation, its hundreds of giant stone statues, their destruction by the carvers' own descendants, the violent transformation of its Polynesian society, and as a metaphor for our world today (1). New information about these themes has recently been flooding in from on-going projects.

When first seen by Europeans in the 1700s, Easter was almost unique among tropical Pacific islands in lacking trees over 3 m tall. But pollen in swamp cores revealed the former existence of a giant palm similar to the world's largest living palm, the Chilean wine palm (see the first figure). By identifying 78,000 bits of burned wood from raidocarbon-dated ovens and middens, Orliac and Orliac (2) recognized more than 20 other tree and woody plant species exterminated during human settlement. The palm was mostly gone by A.D. 1450, and the other large trees by A.D. 1650, after which the islanders had to burn grasses and sedges instead of wood for fuel.

The end of the forest brought other huge
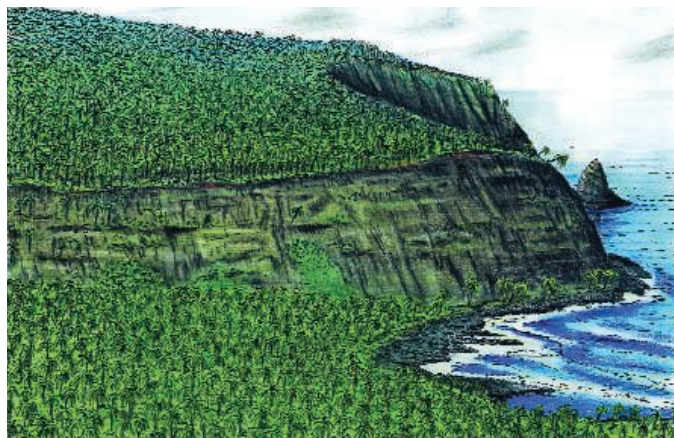
Before human arrival. In this artist's view, the Poike Peninsula is covered with a forest dominated by a giant palm tree, now extinct.

losses for islanders. The palm had yielded food (more than 400 liters of sap per tree per year, plus nuts and palm hearts) and material for baskets, sails, thatching, and mats (3). Other vanished tree species had yielded edible fruits, fiber for rope, bark cloth, and wood for canoes, levers, and carvings (2).

Deforestation also forced changes in horticultural practices (3–6). Easter's early farmers planted crops between the palms, which provided fertilizer, shade, and protection for the soil. In that first phase, erosion was negligible, and horticulture was sustainable (3). Around A.D. 1280, the islanders began felling the palms, removing the trunks (presumably for

timber), and burning the debris, as shown by a radiocarbon-dated charcoal layer, burned roots and palm nuts, and burned palm stumps chopped off near the ground, but no large pieces of trunk wood at these sites (3). The loss of the palm canopy exposed soil to heating, drying, wind, and rain. The resulting sheet erosion proceeded uphill at 3 m/year, removing fertile topsoil and burying down-slope settlements and gardens (3). Palm burning and sheet erosion have been studied especially at Poike but also appear elsewhere until A.D. 1520.

Faced with lower crop yields as a result of deforestation, the islanders responded around A.D. 1400 by occupying the formerly little-used uplands, and by introducing the remarkable labor-intensive gardening method of "stone mulching" on a vast scale (3–6). That meant covering half of the island with more than a billion stones averaging 2 kg in weight (6). In experiments, stone mulching decreases soil water evaporation, protects against wind and rain erosion, and reduces daily temperature fluctuations (7). Pulverized stones may also raise soil fertility by slowly releasing nutrients (8). That function would have been valuable, because nutrient levels (especially phosphorus) often limit tropical

The author is in the Geography Department, University of California at Los Angeles, Los Angeles, CA 90095–1524, USA. E-mail: jdiamond@geog.ucla.edu