A Neuromorph's Prospectus

Kwabena Boahen | Stanford University

As transistors shrink to nanoscale dimensions, trapped electrons are making it difficult for digital computers to work. In contrast, the brain works fine with single-lane nanoscale devices that are intermittently blocked. Conjecturing that error-tolerance can be achieved by combining analog dendritic computation with digital axonal communication, neuromorphic engineers have created Neurogrid, the first neuromorphic system with billions of synaptic connections.

Nanotransistors present challenges for digital computers because of electrons captured by dangling bonds at the silicon–silicon-dioxide interface. Depending on how large a *thermal fluctuation* (see the "Primer" sidebar; all words in italics throughout this article are defined in this sidebar) is required to break the bond formed, the "trap" is a source of stochasticity (short-lived) or of heterogeneity (long-lived). Both types of variability have severe consequences if the transistor has only a few "lanes" of electron traffic. Each lane is about 5-nm wide—the distance at which a trapped electron's electric field brings an approaching electron to a halt—the exact width depends on *permittivity*. Thus, the number of lanes drops below six when the transistor's width shrinks below 30 nanometers (left side of Figure 1). With so few lanes, a smattering of "accidents" or "potholes" brings traffic to a complete halt, shutting off the current intermittently (short-lived traps) or permanently (long-lived traps), respectively.

To avoid an unacceptable increase in dead chips or computer crashes, the industry is in the process of switching from planar to 3D transistors (right side of Figure 1). This newfangled transistor's "double-decker freeway," so to speak, made it possible to shrink its footprint—allowing more to fit on a wafer—while increasing, rather than decreasing, the number of lanes. Hence, the likelihood that traffic halts, while not eliminated, is kept negligible. Consequently, the fraction of chips that die stays small and computer crashes occur at manageable rates. Going 3D, however, increased the fabrication process's

 1521-9615/17/$33.00 © 2017 IEEE Copublished by the IEEE CS and the AIP

Primer

Crossbar: connects each of N axon wires to each of N dendrite wires using an $N \times N$ array of synapse circuits. If connectivity is sparse, synaptic recourses are wasted—for instance, the wasted fraction is $1 - 1/\sqrt{N}$ when each axon only connects to $\sqrt{N}$ dendrites. IBM's TrueNorth chip uses a crossbar.[1]

Gaussian noise: its amplitude at any point in time is described by the Gaussian distribution, also known as the normal distribution. It is said to be white if amplitudes at any pair of times aren't correlated and said to be additive if it's simply added to the signal.

Memory chip: stores digital information in a crossbar arrangement of memory cells, each of which typically hold 1 bit. This circuitry is now being fabricated on multiple planes—a step toward truly 3D circuit construction—relieving the pressure to shrink transistor footprint.

Mesh: connects each node to its nearest neighbors (north, south, east, and west in the 2D version). Copying isn't possible in this routing network—it leads to deadlock—a situation analogous to gridlock in downtown Manhattan. TrueNorth uses a mesh.[1]

Permittivity: a measure of how much a medium resists the formation of an electric field. Using the permittivity of bulk silicon to calculate the electric field and equating the work done against it to the mobile electron's thermal energy (at 25°C) yielded the 5-nm lane width. Accounting for the adjacent oxide's lower permittivity increases it to 9.5 nm,[2] whereas accounting for charge-screening decreases it to 2 nm.[3]

Pipelining: an implementation whereby multiple operations overlap in their execution, thereby increasing throughput. It's directly analogous to how a car assembly line works.

Poisson distribution: describes the probability of a certain number of discrete events happening in a fixed interval of time, or space, assuming these events happen independently. If the mean number is μ, the probability of one or more events happening is $1 - e^{-\mu}$.

Scaling principles: a derivation from first principles of how a MOS transistor's switching energy (and time) scales as its dimensions shrink.[4] Accordingly, shrinking dimensions by a factor of S = 180 nm/28 nm while dropping voltages by a factor of U = 1.8 V/1 V reduces switching energy by a factor of $SU^2 = 20.8$.

Thermal fluctuation: a deviation of a microscopic object's energy from the population's mean energy ($kT = 4.1 \times 10^{-21}$ J at 25°C). The mean as well as the fluctuations increase in energy as temperature increases, such that a fluctuation of size E occurs with probability proportional to $\exp(-E/kT)$.[5]

Voltage: the amount of energy per unit charge. Thus, if E joules of energy is expended trafficking m electrons, the corresponding voltage is $E/(mq)$, where q is the electronic charge. This voltage increases as m drops with decreasing transistor size, reaching 0.25 V for 44 electrons and 0.5 V for 11 electrons when $E = 220kT$, at T = 25°C.

References

1. P. Merolla et al., "A Million Spiking-Neuron Integrated Circuit with a Scalable Communication Network and Interface," *Science*, vol. 345, no. 6197, 2014, pp. 668–673.
2. E. Simoen et al., "Explaining the Amplitude of RTS Noise in Submicrometer MOSFETs," *IEEE Trans. Electronic Devices*, vol. 39, no. 2, 1992, pp. 422–429.
3. A. Asenov et al., "RTS Amplitudes in Decananometer MOSFETs: A 3-D Simulation Study," *IEEE Trans. Electronic Devices*, vol. 50, no. 3, 2003, pp. 839–845.
4. J.M. Rabaey, A. Chandrakasan, and B. Nikolic, *Digital Integrated Circuits,* Prentice Hall, 2003.
5. C.A. Mead and L. Conway, *Introduction to VLSI Systems*, Addison Wesley, 1980.

complexity. As a consequence, in 2014, the number of transistors a dollar bought dropped for the very first time (Figure 2).[1] This cost trend is for information-processing chips, not for *memory chips*.

The challenge posed by single-lane nanotransistors—the ultimate in scaling—will be extremely difficult to surmount with an all-digital approach to computing. A single-lane transistor is envisaged to have a cylindrical channel with the gate all around it. Given the channel's surface area and the density of traps, which is measured to be 5.4×10^{-4} nm^{-2},[2] we can calculate the probability of such a transistor having one or more traps, in which case it will switch completely off intermittently or even indefinitely. With a diameter of 5 nm and a length of 5 nm, the surface area is 78.5 nm^2, and hence each single-lane transistor will have 0.042 traps, on average. As the distribution is *Poisson*,[3] 4.2 percent of these nanotransistors will have one or more traps. Stochastic switching or outright failure in this fraction of transistors will be catastrophic for our present all-digital computing paradigm.

Analog Computation cum Digital Communication

The challenge nanotransistors present can be addressed by combining analog computation with digital communication like the brain does (Figure 3). Digital computers and brains both use digital signals to communicate, but do so differently. Because a computer's digital computation can't tolerate errors, a high cost is paid in energy to squelch stochasticity and heterogeneity in transistors.[4] In contrast, by

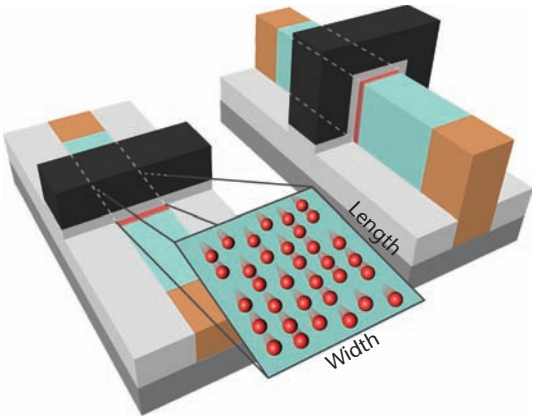

Figure 1. Planar and 3D transistors. In a planar transistor (left), electrons (red) travel along one side of a piece of silicon, from source (brown) to drain (brown). Their flow is controlled by voltage applied to the gate (black), which is insulated from the channel by a thin layer of silicon-dioxide (medium gray). In a 3D transistor (right), electrons travel along three sides of a "fin" protruding from the surface.

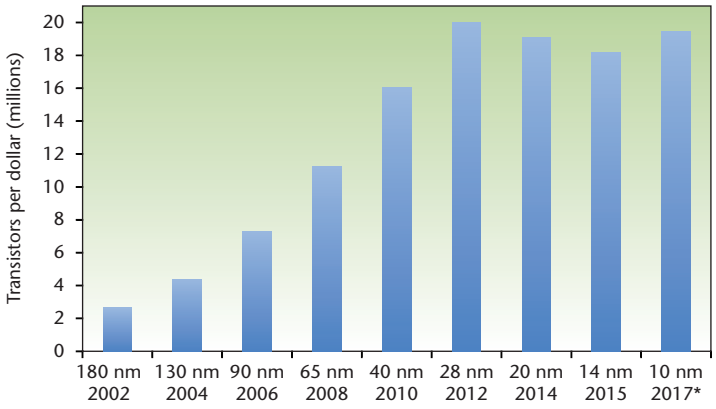

Figure 2. Transistors per dollar. Manufacturing advances continue to shrink transistors' dimensions over time (gate length given). Since 2014, however, a dollar no longer buys more transistors every year—or two—halting a half-century trend dubbed Moore's law. (Source: the Linley Group, used with permission.)

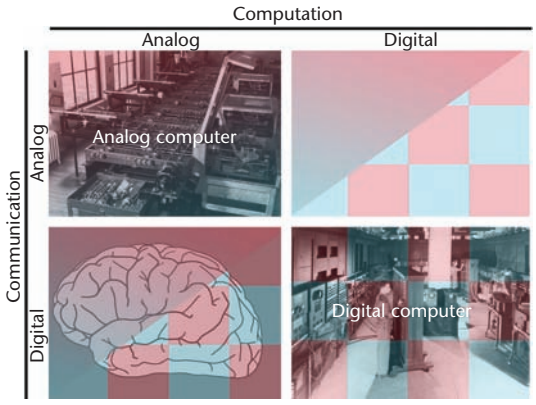

Figure 3. Signal choices for communication and computation. Communication moves operands and results around while computation combines operands to produce results. Analog computers (upper left) use analog signals to compute as well as to communicate, whereas digital computers (lower right) use digital signals to compute as well as to communicate. In contrast, the brain (lower left) uses analog signals to compute and digital signals to communicate. Using analog signals to communicate and digital signals to compute (upper right) hasn't been explored. (Digital computer source: US Army, public domain.)

using analog computation instead, which degrades gracefully, the brain tolerates errors in its digital communication. This tolerance enables it to activate just 20 stochastic, single-lane, nanoscale devices per elementary operation (Figure 4), which is just 20 fJ/op. Thus, 20 W suffices to convey spikes to and graded potentials from each of the brain's 10^{15} synapses once per second. The tradeoffs between analog and digital signal choices in communication and computation can be quantified using information theory and thermodynamics.

For communication, using many error-prone, low-energy, channels—as brains do—is more energy-efficient than using a few pristine, high-energy, channels—as computers do—because the information conveyed decreases only logarithmically as signal energy decreases. In the 1940s, Claude Shannon came up with a quantitative measure for information and derived the capacity of a communication channel in bits of information per second.[5] Each signal the channel conveys carries a certain number (b) of information bits. This number grows logarithmically ($b = \frac{1}{2}\log_2(1 + E/kT)$) with the ratio of signal energy (E) to noise energy (kT, for thermal noise). This expression was derived for a channel with additive, white *Gaussian noise*. The number of signals conveyed per second grows linearly with bandwidth (B). Their product gives the channel capacity ($C = Bb$). Notice that, for $E \gg kT$, the number of bits a signal carries drops by only one when its energy decreases a little more than fourfold—for example, it drops from two to one when signal energy decreases from $15kT$ to $3kT$. Hence, energy efficiency ($b/(E + kT)$) doubles, increasing from ⅛ to ¼ bits per kT. Note, however, that two of these low-energy channels are needed to match the high-energy channel's capacity, taking up more space. The complete space-energy-bandwidth tradeoff has been analyzed for

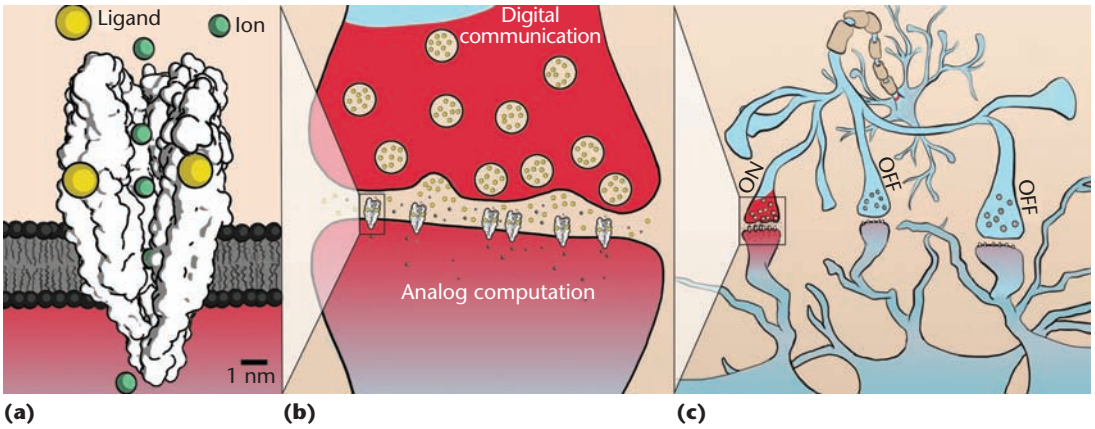

Figure 4. Elementary brain operations. (a) Buffeted by thermal energy, the ion channel opens and closes continually, passing a current resembling a random telegraph signal that switches between 0 and 5 pA. A brief pulse of voltage or ligand may cause the open probability to rise from close to zero to 0.2, decaying exponentially thereafter (at the population level) with a time constant of about 10 ms. Thus, the ion channel conducts an average of 1 pA across 0.1 V and consumes 1 fJ. (b) Vesicles loaded with ligand fuse with the membrane of an axon's terminal, a process successfully triggered about a third of the time by 0.1V deflections in its membrane potential that last about 1 ms. These spikes subserve digital communication. The ligand diffuses across a synaptic cleft and binds to ion channels in a dendrite's juxtaposed membrane, causing a graded deflection in its membrane potential. These graded potentials subserve analog computation. (c) In communicating a spike from its cell body throughout its axonal arbor to thousands of synapses and graded potentials from these synapses along dendritic branches to their cell bodies, only 20 ion channels are opened per synapse. As these ion-channels open briefly (10 ms) and spikes arrive infrequently (1 spike/s/synapse), only 100 of the 10^4 synapses a neuron receives are active at any time, passing 2 nA in total.

neuromorphic systems.[6] Instructively, communication energy efficiency is maximized by trading most of our signal-to-noise ratio for energy savings—avoiding fine analog coding (Figure 5). A similar argument with respect to noise energy, which increases as the channel's bandwidth increases, reaches the same conclusion.[7]

Errors are highly probable when communication is maximally energy efficient, the regime in which the brain operates. An axon conveys all-or-nothing voltage spikes that fail to trigger ligand release at its terminals two-thirds of the time[8] and carry little more than 1 bit of information.[9] In contrast, a digital computer's binary *voltage* signals are corrupted no more than once every 10 days or so, even though each of its billion circuits performs a billion operations per second. To achieve such a low error rate ($p_{err} = 10^{-24}$), it must operate with signal energy E greater than $220kT$; the number of electrons trafficked determines the voltage this amount of energy corresponds to. The probability that a thermal fluctuation causes an error is given by $\exp(-\frac{1}{4}E/kT)$ for binary coding with a threshold of ½, which computers use, as an error occurs when the fluctuation exceeds a quarter of the signal's energy, E (see Figure 5, error probability).[4] Consequently, the computer's communication energy efficiency is 60 times

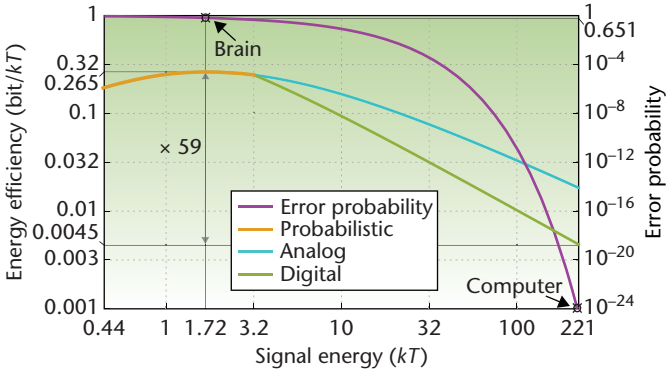

Figure 5. Energy efficiency of communication. A signal with energy E conveys $b = \frac{1}{2}\log_2(1 + E/kT)$ bits of information with an energy efficiency of $b/(E + kT)$ bits per joule. For $E > 3kT$, it conveys more than 1 bit (analog); binary coding caps this to 1 bit (digital). For $E < 3kT$, it conveys less than 1 bit (probabilistic) as noise frequently foils the signal (error probability). The brain operates in this regime.

lower than the maximum possible $(1/(\ln(4)e) = 0.265$ bits per kT), which is achieved when $E = (e - 1)kT = 1.72kT$ and $p_{err} = 0.651$.

For computation, using analog signals—as brains do—is more energy efficient at low precision, whereas using digital signals—as computers do—is more energy efficient at high precision, because

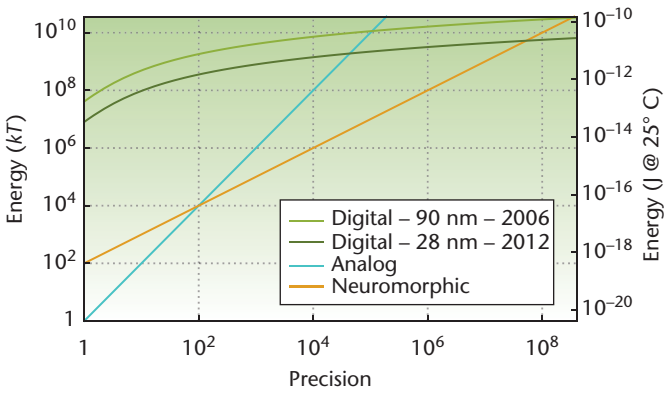

Figure 6. Energy cost of computation. Starting at the thermal limit, two decades above, and seven decades above, respectively, energy increases quadratically, linearly, and logarithmically with precision (signal amplitude/noise amplitude) for analog, neuromorphic, and digital computation. This scaling of power per unit bandwidth to low-pass filter a signal predicts that analog is the most energy-efficient at low precision (below 10^2), neuromorphic at intermediate precision (between 10^2 and 5×10^7), and digital at high precision (above 5×10^7). The neuromorphic curve assumes neuron count, n, scales linearly with precision, r, starting with $n = 50$ at $r = 1$, and that a silicon neuron consumes twice as much energy as an ideal analog low-pass filter.

energy scales differently with precision in the analog and digital cases.[10,11] Consider low-pass filtering a temporal signal in an analog fashion or in a digital fashion, that is, implementing the dynamic transformation $\tau dx/dt + x(t) = c(t)$. An ideal analog first-order, low-pass filter's precision, defined as the ratio of (root-mean squared) signal amplitude to noise amplitude, is given by $r = \sqrt{(E_{ana}/kT)}$, where E_{ana} is the energy consumed to cycle its output voltage from the minimum value to the maximum value and back. This relationship arises simply because the energy consumed to generate a voltage signal scales quadratically with its amplitude. Hence, $E_{ana} = r^2kT$ joules per cycle (Figure 6, analog). In contrast, a digital implementation requires a multiply-accumulate unit, which conducts a number of boolean operations every time step that grows quadratically with the number of data bits, b. For $b = 16$, it consumes $E_{mac} = 0.17$ and 0.87 pJ per multiply-accumulate, respectively, for units fabricated in 28 and 90 nm processes operating at 0.25 and 0.19 V (their minimum-energy points).[12] Thus, if a cycle is divided into m time steps, it consumes $E_{dig} = mb^2E_1$ joules per cycle, where $E_1 = E_{mac}/16^2$. Substituting $b = \log_2(r + 1)$ yields $E_{dig} = m(\log_2(r + 1))^2E_1$ (Figure 6, digital). For $m = 50$ and $r = 255$ (that is, $b = 8$), it consumes 2.1 pJ per cycle (28-nm process). For comparison, the ideal analog implementation consumes 0.26 fJ per cycle—8,000 times less. In this case, comparing an actual digital implementation with an ideal analog implementation is justified because, in practice, analog circuit designers come to within a factor of two or three of the ideal—the so-called noise efficiency factor.[13]

Analog computation consumes much less energy at low precision because it uses much fewer primitives than digital computation. In our low-pass filter example, it uses just two physical primitives: a capacitor and a transconductance amplifier built with as few as five transistors. In contrast, digital uses 56 logical primitives—three-input, two-output Boolean functions—each built with 30 transistors. Nevertheless, analog's energy consumption, which scales quadratically with precision, eventually overtakes digital's, which scales logarithmically. The crossover point has migrated to the left over the years—as the digital curve moved down with miniaturization—favoring digital over analog computation for more and more applications (see Figure 6). However, this migration has stalled due to digital computation's inability to work with transistors with a few electron-traffic lanes—widening a transistor's channel increases the energy required to switch it on.

Here's how we neuromorphs plan to combine analog computation with digital communication to address the nanotransistor challenge. We have a five-point plan: first, implement computation with subthreshold analog circuits[14,15] to consume close to the theoretical minimum energy. Second, implement communication with asynchronous digital circuits[16,17] to be robust to transistors that shut off intermittently. Third, distribute a computation across a pool of silicon neurons to be robust to transistors that shut off intermittently or permanently. Fourth, communicate spikes from pool to pool at a rate that scales linearly with the number of neurons per pool. And, finally, encode continuous signals in these spike trains with precision that scales linearly with the number of neurons per pool. With the successful execution of this plan, neuromorphic systems will achieve better energy efficiency than all-analog or all-digital solutions across a five-decade precision range (Figure 6, neuromorphic). This efficiency offers an attractive solution for numerous embedded computing applications that are energy constrained, ranging from autonomous miniature robots[18] to implantable neural prostheses.[19]

Before describing how neuromorphic computing can achieve linear scaling of precision with neuron count, let's review the progress made toward scaling interpool communication linearly

with neuron count and toward performing arbitrary computations with pools of silicon neurons. After three decades of effort, neuromorphs have gotten interpool communication to scale linearly with neuron count at scales that require memory and computation to be distributed over significant distances. They did this by using readily available CMOS (complementary metal-oxide-semiconductor) technology operating in mixed analog-digital mode. And they've developed a method to map systems described by arbitrary linear or nonlinear differential equations onto networks of silicon neurons that's tolerant to heterogeneity and stochasticity. And, finally, they have proven that this method is scalable by using it to build an artificial brain that perceives, cogitates, and acts. They now seek to encode continuous signals in silicon neurons' spike trains such that precision scales linearly with the number of neurons, rather than with its square root.

Neurogrid: Organizing Physical Primitives at Scale

A decade ago, in 2006, my group embarked on a project to scaleup neuromorphic systems by three orders of magnitude.[20] Through arguments presented in the previous section, I had reached the conclusion that nanotransistors provide better physical computational primitives than logical ones. At that time, however, we neuromorphs hadn't conceived a feasible strategy to organize our physical computational primitives—analog silicon neurons and synapses—at a consequential scale. In the early 1990s, to circumvent communication limits imposed by a chip's planar wiring layout, we invented a pragmatic scheme to share a small set of wires among a population of silicon neurons instead of dedicating a wire to each silicon neuron's axon. Sharing is possible because signals travel much faster along wires than along axons. However, the address-event bus, as this scheme was called, had reached its limits.

The address-event bus had been successfully used to build networks that had from a few hundred thousand[21] to a few million[22] synaptic connections. When a neuron spikes, an address unique to it is encoded by a logic circuit called an arbiter. This address is then sent on a digital bus to a random access memory (RAM), where it is used to look up addresses of neurons this neuron is connected to. These addresses are decoded by another logic circuit called a selector that triggers synaptic input to those neurons.[23,24] The RAM's maximum read frequency, however, limits the number of synaptic connections to a few million—even when the RAM is placed on chip, right next to the silicon neuron array.[22] This number is for a 14-year-old fabrication process (180 nm); it would be an order-of-magnitude larger for a state-of-the-art one.

The fundamental problem is that we were trafficking addresses at a rate that scales as the product of the number of neurons in a pool and the number of synaptic connections each neuron makes. This rate is proportional to nPN when each neuron in an N-neuron pool connects to n neurons in each of P other N-neuron pools. As both n and P appear to scale as $\sqrt{N}$ in the brain,[25] it is as if the address-event bus is emulating a nerve with a number of axons that grows as N^2. We reduced the traffic by emulating two organizational principles the cortex uses to scale its axon count as N (linear scaling): overlapping dendritic trees within a pool of neurons and hierarchical axonal arbors between pools of neurons (see the "Mimicking Overlapping Dendrites and Hierarchical Axons" sidebar).

Emulating the cortex's overlapping dendritic trees and hierarchical axonal arbors enables Neurogrid to accommodate columnarly organized cortical networks with thousands of synaptic connections per neuron. Columnar organization describes the expedient manner in which neurons are arranged and interconnected in some cortical areas.[26] In sensory cortical areas, which have been extensively studied, neurons respond to certain stimuli with higher spike rates. For instance, when shown bars of different orientations, neurons in visual cortex respond most strongly to a specific orientation. Their orientation preference changes gradually as we move across the cortical sheet (x, y) but not along its depth (z).[27] This lack of change with depth led to the notion that cells in a column cooperate to process the stimulus. It's true that a column's cells prefer the same stimuli, but they don't have the same physiology, nor do they have the same anatomy. There are distinct cell types connected in distinct wiring patterns. This cellular specialization led scientists to hypothesize that a canonical cortical microcircuit instanced in each column mediates interactions with areas at lower, the same, or higher levels of the cortical hierarchy.[28]

Neurogrid's software stack maps each cortical area onto a group of Neurocores spanned by a subtree of its interchip-routing network. Circular pools of neurons at corresponding (x, y) locations on these Neurocores' 2D silicon neuron arrays model a column, using as many Neurocores (addressed by z) as there are distinct cell types

Mimicking Overlapping Dendrites and Hierarchical Axons

Overlapping dendritic trees extend dendritic branches to meet incoming axons, eliminating the axons' terminal branches (Figure A1).[1] This strategy is efficient because higher-level dendritic branches recursively aggregate signals from more and more axons. Neurogrid emulates this organizational principle by interposing a resistive grid—instead of a *crossbar*—between its synaptic circuits and its silicon neurons' cell bodies, cutting traffic by a factor equal to the number of neighboring neurons (n) that thereby receive input (Figure A2, $n = 3$).

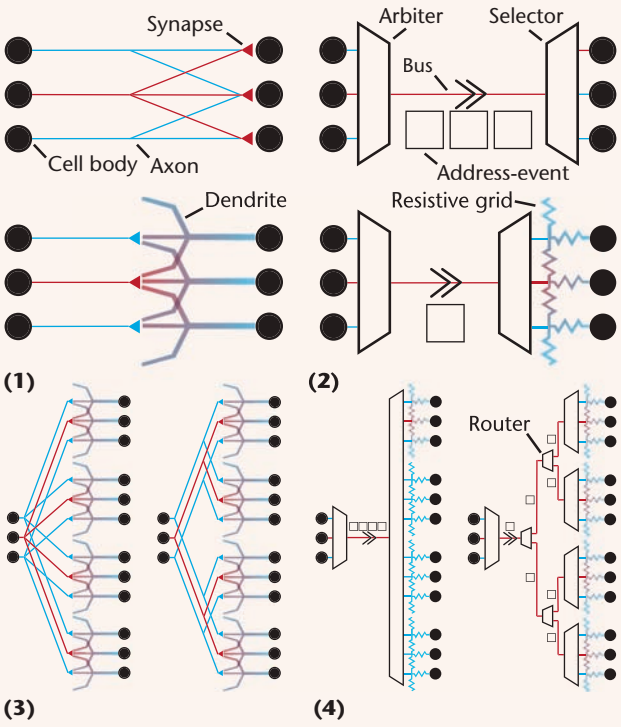

Figure A. Mimicking overlapping dendrites and hierarchical axons. (1) Overlapping dendritic trees (bottom) eliminate axons' terminal branches (top). (2) Mimicking them with a resistive grid allows three address-events (top) to be replaced with one (bottom). (3) Hierarchical axonal branches minimize axonal arbors' lengths (right versus left). (4) Mimicking them with routers enables four address-events (left) to be recreated by copying one (right).

Hierarchical axonal arbors minimize wiring by recursively replacing p long branches with a single long branch that feeds p short branches, thereby amortizing the energy expended to communicate across the long branch over many synapses (Figure A3, $p = 2$).[1] Neurogrid emulates this organizational principle by interconnecting its silicon neuron pools with a tree-like routing network[2]—instead of a *mesh*—cutting traffic at the tree's root by a factor equal to the number of copies (P) that arrive at the tree's leaves (Figure A4, $P = 4$). These two factors compounded cut traffic nP-fold, thereby achieving linear scaling.

Neurogrid actually realizes the corresponding nP-fold increase in throughput for columnarly organized cortical networks. To see this, consider routing a spike to K columns, copying it to each of those column's L layers, and triggering postsynaptic potentials in n-neuron pools (that is, neighboring neurons) in P of those layers. In this case, KPn synaptic connections are made by looking up just K entries in each of $L + 1$ RAMs. That is, instead of looking up addresses at a rate of $K \times Pn \times Lnf_{spk} = KPLn^2f_{spk}$, where f_{spk} is the mean spike rate, they're looked up at a rate of $KLnf_{spk}$ (nP times slower). For example, with $K = 10$, $L = 6$, $n = 100$, and $P = 5$, each RAM is read at the rate of 6,000 f_{spk} instead of 3,000,000 f_{spk}. Hence, a 500-fold increase in throughput is realized by *pipelining* the first RAM read with the remaining six, which occur concurrently. Thus, moderate improvements in the RAM's maximum read frequency, and in the digital bus's signaling rate,[3,4] suffice to increase the number of synaptic connections from millions to billions.

References

1. D. Chklovskii et al., "Synaptic Connectivity and Neuronal Morphology—Two Sides of the Same Coin," *Neuron*, vol. 43, no. 5, 2004, pp. 609–618.
2. P. Merolla et al., "A Multicast Tree Router for Multichip Neuromorphic Systems," *IEEE Trans. Circuits and Systems*, vol. 61, no. 3, 2014, pp. 820–833.
3. P. Merolla et al., "Expandable Networks for Neuromorphic Chips," *IEEE Trans. Circuits and Systems I*, vol. 54, no. 2, 2007, pp. 301–311.
4. A. Chandrasekaran and K. Boahen, "A 1-Change-in-4 Delay-Insensitive Interchip Link," *IEEE Int'l Symp. Circuits and Systems*, 2010, pp. 3216–3219.

(Figure 7a). Intercolumn (primary) axonal branches are routed by using the spiking neuron's address (z_s, x_s, y_s) to retrieve the target columns' centers (x_c, y_c) —along with the path to the spanning subtree's root—from an off-chip RAM (on a daughterboard).[29] This RAM is programmed to replicate the cortex's preferred-feature-specific intercolumn connectivity. Intracolumn (secondary) axonal branches are routed by copying the spiking neuron's layer (z_s) together with the target columns' centers (x_c, y_c) to all of the subtree's Neurocores, including ones the axon doesn't connect to.[29] These unneeded copies are filtered based on a bit stored in an on-chip RAM. The same entry applies to all (x_c, y_c) locations—the decision to filter or deliver depends only on which layer (z_s) the axon originates

from—thanks to columnar organization. Finally, pool-spanning dendritic branches—arborizing over a circular disc centered at (x_c, y_c)—are realized using the (2D) resistive grid. A transistor-based implementation that elegantly exploits current flow by diffusion in the subthreshold region makes it possible to adjust the spatial decay rate electronically to match the column's radius.[30]

Neurogrid simulates such cortical models with up to a million neurons and billions of synapses in real time, the first system ever to do so. To simulate a cortical model on Neurogrid, computational neuroscientists specify properties of ligand- and voltage-gated ion channel populations[31,32] and of spike generation[33] for neurons in each of the model's layers, as well as their intercolumn connections, intracolumn connections,[29] and dendritic radius[30] (Figure 7b). They code up this information in a Python script that Neurogrid's software stack parses to configure its Neurocores. To run a model with 1 million neurons firing 10 spike/s, on average, connected by 8 billion synapses, Neurogrid's 16 mixed analog-digital Neurocores consume 3.6 W altogether,[6] or 45 pJ per synaptic activation—a useful measure of energy efficiency—as the rate of both computation and communication nominally scales with the product of the average spike rate and the total number of synapses.

Neurogrid's energy efficiency matches that of a board with 16 of IBM's all-digital TrueNorth chips, even though these chips were fabricated in a process 40 times denser.[34] The IBM board consumes 2.5 W to run a model with 16 million neurons firing 50 spike/s, on average, connected by 1 billion synapses,[35] which is 50 pJ per synaptic activation. Neither number—the TrueNorth board's or Neurogrid's—includes power consumed by the field programmable gate arrays (FPGAs). When the 40-fold density difference between the processes TrueNorth (28 nm) and Neurogrid's Neurocore (180 nm) were fabricated in is accounted for using *scaling principles*, Neurogrid is 23 times more energy efficient.

The secret to scaling interpool communication linearly with neuron count is in emulating the cortex's hierarchically branching wiring patterns. Hierarchical organization amortizes the cost of more energetically expensive memory lookups and address-event communications over greater numbers of synapses so as to minimize the average energy consumed per synaptic activation. This organizational principle recognizes that energy consumption is proportional to how far information travels: reading data from a large memory or moving it over a long distance is much more expensive than adding one number to another inside a logic circuit. By adopting this organizational principle, Neurogrid organizes its networks of silicon neurons scalably, and thereby simulates large-scale cortical models energy efficiently. While it worked well in that capacity, we presently lack a method that maps arbitrary computations onto the cortex's myriad cell types, canonical microcircuit, and columnar organization.

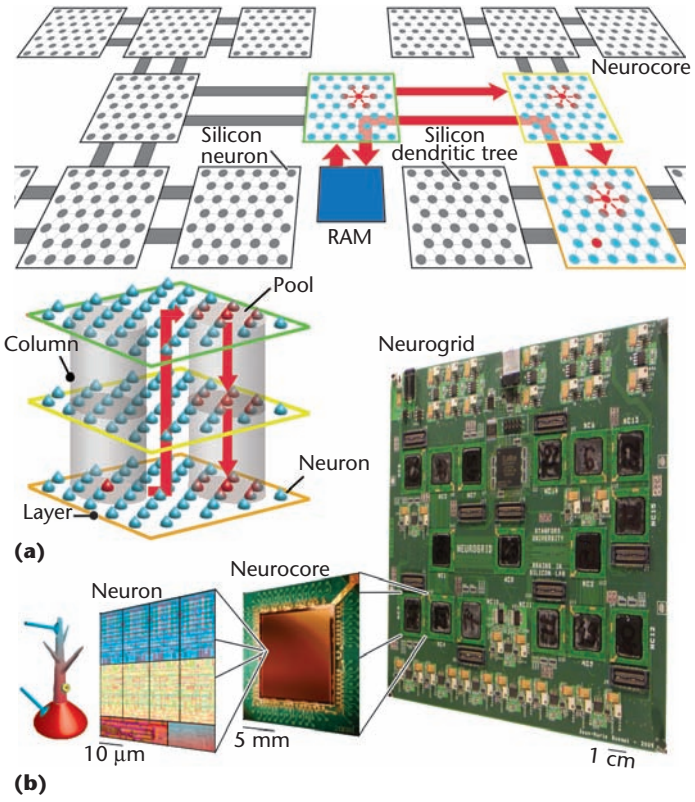

Figure 7. Neurogrid. (a) Cell layers (green, lemon, and orange) are mapped onto Neurocores, which are connected in a binary tree network. Inter- and intracolumn connections are programmed in off- and on-chip RAM, respectively. (b) A neuron has four ligand-gated and four voltage-gated ion-channel populations, a dendrite, and a cell body, all modeled using subthreshold analog circuits (color coded; see cell cartoon). Neurocore has 65,536 silicon neurons (256 × 256 array), as well as spike routers implemented using asynchronous digital circuits. Neurogrid holds 16 interconnected Neurocores.

Performing Arbitrary Computations with Physical Primitives

My group recently developed a procedure to map arbitrary dynamical systems onto networks of spiking silicon neurons in collaboration with Chris Eliasmith's group at the University of Waterloo. In 2003, Eliasmith and Charles Anderson made a

The Neural Engineering Framework

In the NEF, a neural ensemble's spike trains encode a vector (denoted by $\mathbf{x}$) of D continuous signals. Linear or nonlinear static transformations of this vector (defined by $\mathbf{f}$) are linearly decoded from the ensemble's spike rates a_i (Figure B1). To accomplish this decoding, the spike rates—estimated by filtering the spike trains with exponentially decaying synapses—are weighted by D-dimensional decoding vectors (labeled $\mathbf{d}_i$) assigned to each neuron and summed. The resulting vector of D continuous signals (labeled $\mathbf{f}(\mathbf{x})$) is encoded in the next ensemble's spike activity by taking its inner product with D-dimensional encoding vectors (labeled $\mathbf{e}_j$) assigned to that ensemble's neurons. For dynamic transformations (defined by $d\mathbf{x}/dt = \mathbf{f}(\mathbf{x}(t),\mathbf{c}(t))$, this vector is encoded in the very same ensemble's spike activity, forming a recurrent network. Whereas decoding vectors are chosen to approximate the desired transformation as closely as possible, encoding vectors are chosen randomly from a uniform distribution on the D-dimensional unit-hypersphere.[1] Note that decoding vectors ($\mathbf{d}_i^{\mathbf{M}}$) for any linear transformation (defined by $\mathbf{M}$, a $D \times D$ matrix) could be expressly obtained from decoding vectors ($\mathbf{d}_i^{\mathbf{I}}$) for the identity transformation ($\mathbf{d}_i^{\mathbf{M}} = \mathbf{M}\mathbf{d}_i^{\mathbf{M}}$).[1]

To illustrate the NEF network's advantages, imagine converting it into a conventional neural network (Figure B2). Connect each neuron in the first ensemble directly to each neuron in the second ensemble with a synaptic weight equal to the inner product of the first neuron's decoding vector with the second neuron's encoding vector ($w_{ij} = \mathbf{e}_j^{\mathrm{T}}\mathbf{d}_i$). This weight matrix transforms the spike activity pattern in the first ensemble, representing an input vector, directly into a spike activity pattern in the second ensemble, representing the specified transformation of that input vector. However, this approach uses N^2 synaptic weights versus the NEF's $2ND$ decoding and encoding vector components. Typically, $N/D \approx 50$ neurons per dimension, so the conventional network uses 25 times more memory if its synaptic weights have the same number of bits as the components of the NEF network's decoding and encoding vectors. Conversely, an NEF network whose decoding and encoding vector components have 8 bits would use the same amount of memory as a conventional network whose synaptic weights have 0.32 bits ($8 \times 2ND/N^2$). Even more impressive, for a linear transformation ($\mathbf{M}$), a $D \times D$ matrix replaces an $N \times N$ one, resulting in 2,500-fold compression ($(N/D)^2$).

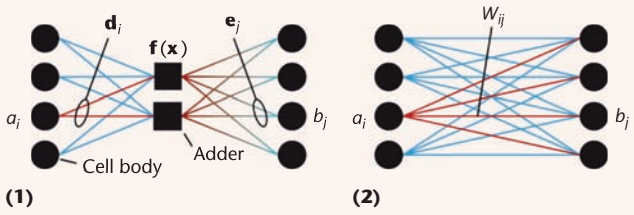

Figure B. NEF network and its equivalent two-layer network.

Reference

1. C. Eliasmith and C.H. Anderson, *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*, MIT Press, 2003.

radical breakthrough by throwing out most of the cortex's complicated anatomy and physiology. They identified a minimal set of neuronal properties—linear exponentially decaying synaptic responses and heterogeneous nonlinear somatic responses—sufficiently powerful to approximate any linear or nonlinear dynamical system.[36] That is, systems described by multidimensional differential equations of the form $d\mathbf{x}/dt = \mathbf{f}(\mathbf{x}(t),\mathbf{c}(t))$, where $\mathbf{x}$ and $\mathbf{c}$ are vectors of continuous signals and $\mathbf{f}$ is any vector-valued function. Their theory, the Neural Engineering Framework (NEF), provides a recipe that enables us to derive the weights applied to synaptic inputs (analog computation) triggered by spikes that silicon neurons emit (digital communication; see the "Neural Engineering Framework" sidebar).

We use the NEF to configure Neurogrid to perform a desired computation in a manner akin to how a logic synthesis tool configures an FPGA. This tool specifies input-output functions (truth tables) and connectivity (closed switches in a wiring matrix) of the FPGA's logical computational primitives (logic gates). Similarly, the NEF takes the physical computational primitives of heterogeneous spiking neurons and interconnects them with programmed weights to perform a desired computation. Only the connections among Neurogrid's silicon neurons are field programmable, however. The input-output functions of its analog silicon neurons (known as tuning curves) aren't programmable. Fortuitously, the NEF doesn't require programmable tuning curves. It only requires them to be nonlinear and heterogeneous (that is, different from each other).

In Neurogrid, the NEF harnesses nonlinearity in the silicon neurons' spiking mechanism and heterogeneity in this behavior introduced by transistor variability. The degree of heterogeneity is such that silicon neurons fed the same input current spike at extremely different rates (Figure 8). Hence, setting neuron i's input current to $\mathbf{e}_i^{\mathrm{T}}\mathbf{x}$ yields a tuning curve, $a_i(\mathbf{e}_i^{\mathrm{T}}\mathbf{x})$, that's different from

neuron j's—even if neuron j's encoding vector is the same as neuron i's ($\mathbf{e}_j = \mathbf{e}_i$). The NEF finds a weighted sum of these measured tuning curves that closely approximates the desired transformation, $\mathbf{f}(\mathbf{x})$. Specifically, it solves for the decoding vectors, $\mathbf{d}_i$, that minimize the mean squared error, $\left\{\Sigma_i \mathbf{d}_i a_i(\mathbf{e}_i^{\mathrm{T}}\mathbf{x}) - \mathbf{f}(\mathbf{x})\right\}^2$.

So far, we've used the NEF to configure Neurogrid to do two things: perform basic mathematical operations and control a robot arm. To gain experience using the NEF with Neurogrid, we configured 4,000 of a Neurocore's silicon neurons to perform three mathematical operations on scalars: identity, squaring, and integration.[37] This work demonstrated for the very first time a procedure to map arbitrary linear or nonlinear dynamical systems onto neuromorphic chips. Next, in collaboration with Oussama Khatib's group at Stanford University, we configured 1,280 of a Neurocore's silicon neurons to control a three-degrees-of-freedom robot arm in a 3D task space (Figure 9).[38,39] In both cases, we implemented the NEF's linear decoding by configuring a FPGA on Neurogrid's daughterboard to deliver spikes to a target with a probability proportional to the corresponding encoding or decoding vector component.[40] Our implementation proved robust to the stochasticity that results from randomly dropped spikes as well as to the heterogeneity that results from transistor variability.

Brains in Silicon: Spaun and Neurogrid

Are the NEF's minimalist neuronal properties—linear exponentially decaying synaptic responses and heterogeneous nonlinear somatic responses—sufficient to build systems capable of complex behaviors? Eliasmith's group addressed this question in 2013. A decade after proposing the NEF, they used it to build Spaun, the first behaving model brain that used spiking neurons.[41,42] Spaun demonstrates convincingly that NEF-synthesized spiking neural networks can perform visual, cognitive, and motor tasks—involving statistical inference, symbolic manipulation, and dynamical control, respectively—and do so at full scale (Figure 10). Spaun has 3.9 million neurons and 123 billion synapses. Running this scale of spiking neural network on commercially available digital hardware leaves a lot to be desired when it comes to real-time operation and energy efficiency, however.

Spaun currently runs 23 times slower than real time on a Nvidia Titan X GPU. This is an impressive achievement, given that four years ago it ran 9,000 times slower than real time on a PC. Elia-

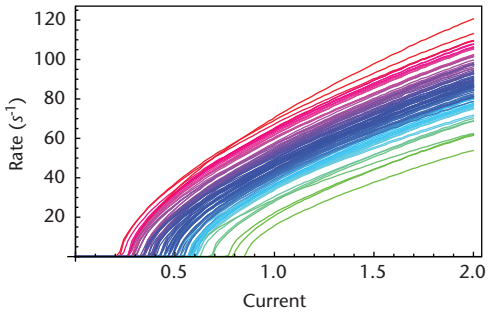

Figure 8. Silicon-neuron heterogeneity. Neurogrid's analog silicon neurons have similarly shaped current-to-spike-rate curves but start firing at wildly different input current levels, due to transistor variability. As a result, the distribution of spike rates across this 100-neuron sample is extremely broad. The input current is normalized such that a silicon neuron without any variability would start firing at 0.5.

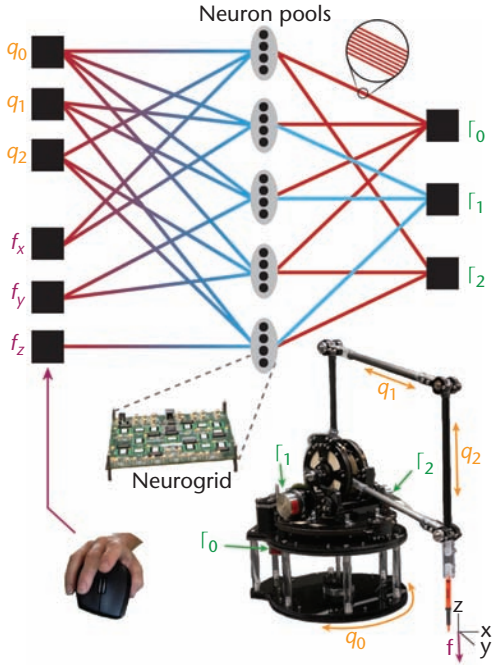

Figure 9. Robot control with Neurogrid. Desired end-effector force, $\mathbf{f} = [f_x\ f_y\ f_z]^{\mathrm{T}}$, and gravitational forces, $-\mathbf{g}$ (canceled by negation), are related to applied joint torques, $\mathbf{\Gamma} = [\Gamma_0\ \Gamma_1\ \Gamma_2]^{\mathrm{T}}$, by $\mathbf{\Gamma} = \mathbf{J}_{\mathbf{x}}^{\mathrm{T}}\mathbf{f} - \Sigma_i m_i \mathbf{J}_{\mathrm{com}_i}^{\mathrm{T}}\mathbf{g}$. The Jacobians, $\mathbf{J}_{\mathbf{x}}$ and $\mathbf{J}_{\mathrm{com}_i}$, relate changes in the robot arm's three joint angles ($\mathbf{q} = [q_0\ q_1\ q_2]^{\mathrm{T}}$) linearly to changes in its end-effector's position ($\mathbf{x}_{\mathrm{ee}}$) and its i-th link's center of mass' position ($\mathbf{x}_{\mathrm{com}_i}$): $\mathbf{dx}_{\mathrm{ee}} = \mathbf{J}_{\mathbf{x}}\mathbf{dq}$ and $\mathbf{dx}_{\mathrm{com}_i} = \mathbf{J}_{\mathrm{com}_i}\mathbf{dq}$. Note that $\mathbf{J}_{\mathbf{x}}$ and $\mathbf{J}_{\mathrm{com}_i}$ are nonlinearly related to $\mathbf{q}$ (for example, $J_{x0,0} = -0.35\sin(q_0)\cos(q_2)$). The torque–force relationship was decomposed into a set of five 3D functions, linear combinations of which yield the individual joint-torques. These functions are decoded from five 256-neuron pools. Each pool's input is a unique combination of joint angles (q_0, q_1, q_2) and force-components (f_x, f_y, f_z)—follow the links.

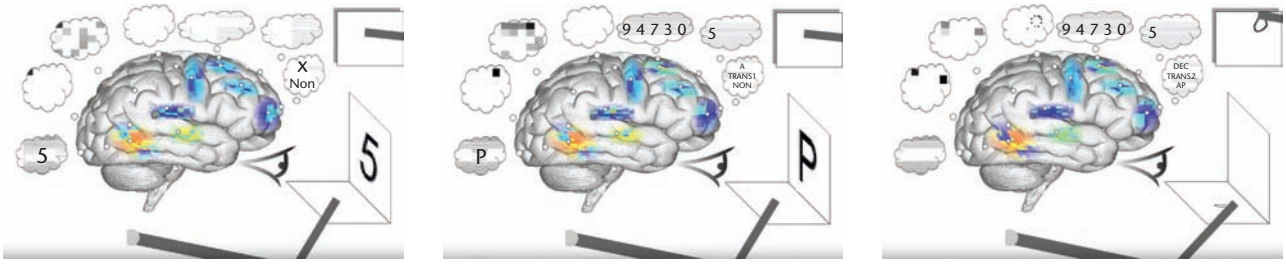

Figure 10. Spaun performs a question-answering task. Its 220,000-neuron visual hierarchy recognizes digits presented to its 28×28-pixel retina (left panel, thought bubble with "5"). Its 154,000-neuron prefrontal cortex stores this query (middle panel, bubble with "5") while its 1 million-neuron dorsolateral prefrontal cortex stores a list of digits presented afterwards (middle panel, bubble with "9 4 7 3 0"). Spaun is then asked to infer the queried digit's position in the list ("K" question) or to recall which digit is in the queried position ("P" question), the case in this trial. Its 500,000-neuron ventrolateral prefrontal cortex applies the right transformation (right panel, bubble with "TRANS2"). Spaun then writes down the correct answer ("0") by using its 35,000-neuron motor cortex to control its arm.

smith's group ported Nengo, their simulator for NEF-synthesized spiking neural networks, onto GPUs to exploit their parallelism.[43] The Titan X's 3,072 cores yielded a 390-fold speed-up. Even more impressive, they realized this speed-up while consuming energy at about the same rate as a PC—a two orders-of-magnitude improvement in energy efficiency. Assuming speed increases linearly, 23 Titan Xs could run Spaun in real time, consuming 23 × 250 W = 5.75 kW, whereas four Neurogrid boards could run Spaun in real time, consuming 4 × 3.6 W = 14.4 W—400 times less.[6] It isn't as simple as slapping these four boards—and their daughterboards—together, however.

While Spaun's decoding vectors, encoding vectors, and transform matrices will fit on just two Neurogrid daughterboards (each has 32 Mbytes), connecting each of Spaun's 61,100 ensembles to 316 others (on average) exceeds these boards' routing capacity. Thanks to the NEF, Spaun's 123 billion synaptic weights are compressed 2,650-fold—replaced by 8.6 million, 16.1 million, and 21.7 million encoding vector, decoding vector, and transform matrix components, respectively. Thus, at 8-bit resolution, they only occupy 46.4 Mbytes of RAM. However, Spaun's 3.9 million neurons fire a few hundred spikes per second each, on average, whereas the routing capacity of Neurogrid's Neurocores and its daughterboard's off-the-shelf chips was designed to accommodate up to 10 spikes per second per neuron—a rate chosen to match the brain's neurons.

To increase routing capacity dramatically, my group is migrating our mixed analog-digital circuit designs from Neurogrid's decade-and-half-old fabrication process (180 nm) to a state-of-the-art CMOS process. This migration will enable us to increase the number of neurons by over an order of magnitude, while at the same time increasing the amount of RAM on a Neurocore-sized chip from 512 bytes to 16 Mbytes—sufficient to store decoders, encoders, and transform matrices on chip. This on-chip RAM will cycle 50 times faster than Neurogrid's daughterboard (1 GHz versus 20 MHz). Compounding this speed-up with performing decoding concurrently in each Neurocore will increase throughput by close to three orders of magnitude and energy efficiency by two orders of magnitude. When these energy savings are combined with energy savings from routing spikes among (up to 10 million) neurons without going off-chip, and from encoding continuous signals in their spike trains efficiently, a 4 million-neuron network will consume just 0.14 W. Thus, instead of consuming as much electricity as four hairdryers to run in real time on 23 Titan Xs, Spaun can run for a whole day on a single 1.5-V, 2,400-mAHr AA battery.

Promising New Directions: Efficient Spike Coding

Novel neural engineering frameworks for approximating arbitrary linear or nonlinear dynamical systems with networks of spiking neurons are being developed that encode information with spikes efficiently, spiking at rates comparable to our brain's neurons. Minimizing the number of spikes used to transmit a signal minimizes the amount of energy consumed to communicate it as well as to lookup the synaptic (or decoding) weights applied to it, thereby reaping energy savings above and beyond those realized by amortizing the costs of more energetically expensive communications and memory lookups over greater numbers of synapses. With these new frameworks, a pool of neurons encodes

a signal with precision that scales linearly with its total spike rate—and hence linearly with neuron count—a dramatic improvement over current frameworks' square-root scaling.

With independent spiking codes, as in the NEF for instance,[36] precision scales as the square root of the total spike rate. While individual neurons' spike trains are periodic, when their spike trains are merged (superposed) during decoding the resulting spike train looks Poisson (Figure 11a). That is, for large neuron count, n, and low individual spike rates, $f \ll 1/T$, the variance of spike counts in T-second time bins becomes equal to the mean count. Hence, precision, defined as the mean over the standard deviation, is proportional to the square root of the mean. Since the mean is proportional to the total spike rate, to double precision, we must quadruple the spike rate. And therefore, as energy is consumed at a rate proportional to the total spike rate, we recapitulate classic analog computation's quadratic increase in energy with precision (see Figure 6, analog).

Recently developed coordinated spiking codes promise to scale precision linearly with total spike rate. With coordinated spiking, fast, strong synapses are introduced between neurons.[44,45] While these synapses make individual neurons' spike trains look Poisson, when these spike trains are merged during decoding, the resulting spike train looks periodic (Figure 11b). As a result, the variance of spike counts remains constant as the spike rate increases. Hence, precision is proportional to the mean count, which itself is proportional to the total spike rate (λ). Thus, spike rate—and hence energy—scales linearly with precision, outperforming classic analog computation (see Figure 6; compare neuromorphic with analog). For example, controlling a robot joint with precision $\lambda/\sigma = 10^2$ and latency $\tau = 10$ ms using independent spiking requires 500,000 spike/s while controlling it using coordinated spiking requires only 2,900 spike/s—173 times lower. To obtain this result, I set $\sigma^2 = \lambda/(2\tau)$ for independent spiking and $\sigma^2 = 1/(12\tau^2)$ for coordinated spiking, which assumes their merged spike trains are Poisson and periodic, respectively, and solved for λ in each case.

Neuromorphs are only just now beginning to explore realizing coordinated spiking neural networks with mixed analog-digital CMOS circuits. Toward this end, with its four ligand-gated ion channel populations, its conductance-based synapses, and its spatially decaying resistive grid, Neurogrid already supports the multiple synaptic time constants and saturating dendritic branches that coordinated spiking networks require (see the "Coordinated Spiking Network" sidebar). When these networks' promise of linear spike rate scaling with precision is realized in neuromorphic hardware, the number of silicon neurons per pool will scale linearly with precision, yielding better energy efficiency than all-analog or all-digital solutions across a five-decade precision range (see Figure 6; compare neuromorphic with analog and digital).

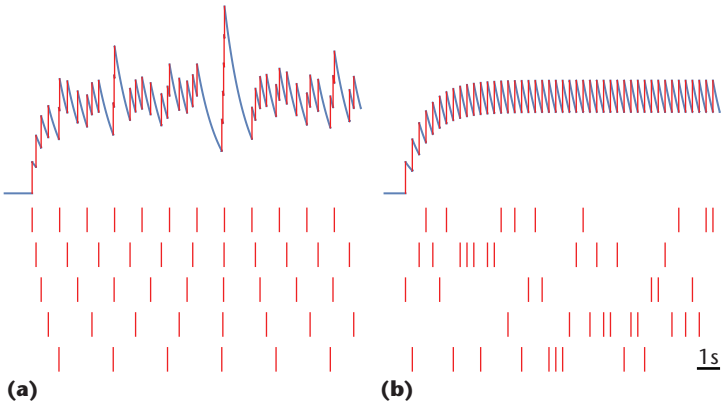

Figure 11. Encoding continuous signals with spike trains: (a) independent and (b) coordinated spiking. In both cases, five neurons increase their spike rate from zero to 0.6 s^{-1} (below), on average, to encode a step change in a continuous signal. When these spike trains are weighted equally and filtered with exponentially decaying synapses to decode the signal (above), the resulting waveforms' mean equals the total spike rate, 3 s^{-1}. However, the variance is 5.2 times lower when spiking is coordinated.

After three decades of effort, we neuromorphs are on the cusp of achieving the full promise of combining analog computation with digital communication like the brain does, a goal that requires us to realize linear—rather than square-root—scaling of precision with pool size. When we succeed, neuromorphic computing will offer greater energy efficiency than all-analog or all-digital computing. Realizing this goal will provide an attractive solution for numerous energy-constrained applications of embedded computing, ranging from autonomous miniature robots to implantable neural prostheses. ■

Acknowledgments

The US National Institutes of Health (NIH) supported this work under its Director's Pioneer Award (DPI-OD000965) and its Transformative Research Award (R01NS076460) programs. The US Office of Naval Research also supported

Coordinated Spiking Network

This network approximates the nonlinear dynamical system $\mathbf{dx}/\mathbf{d}t = -\lambda_{\mathbf{x}}\mathbf{x}(t) + \tanh(\mathbf{A}\mathbf{x}(t)) + \mathbf{c}(t)$ by incorporating two cortical features that the NEF threw out. First, in the cortex, time constants of both excitatory and inhibitory synapses span a vast range (1 ms to 100 ms). Coordinating spiking calls for a set of synapses with an extremely short—ideally instantaneous—time constant to update the neurons' membrane potentials immediately after each spike. These fast synapses' weights are chosen to ensure that a neuron's membrane potential tracks the error's projection onto its decoding vector (denoted by $\mathbf{\Gamma}_i$) through negative feedback.[1] The error is a vector of differences between the continuous signals ($\mathbf{x}$) and their approximations ($\sum_i \mathbf{\Gamma}_i a_i$). These fast synapses are in addition to the slow ones that estimate spike rates (a_i) by filtering spike trains with a run-of-the-mill time constant—just like the NEF does. Second, in the cortex, when a dendritic branch receives too much synaptic input, its membrane potential saturates. Approximating nonlinear dynamical systems calls for a nonlinearity to be applied to the continuous signals that an ensemble's spike activity encodes. While NEF networks exploit spike generation for this purpose, coordinated spiking networks exploit dendritic saturation (modeled by the tanh).[2] A linear transformation chosen to best approximate the desired nonlinear dynamical system is applied first (specified by $\mathbf{A}$). In the original formulation,[2] it was applied afterward, which requires the cell bodies to weigh a dendrite's saturated signal differently from its nonsaturated (that is, linear) signal. The resulting approximation's precision scales linearly with the network's total spike rate when fast synaptic coupling is instantaneous. We don't yet know how fast this coupling must be to preserve linear scaling.

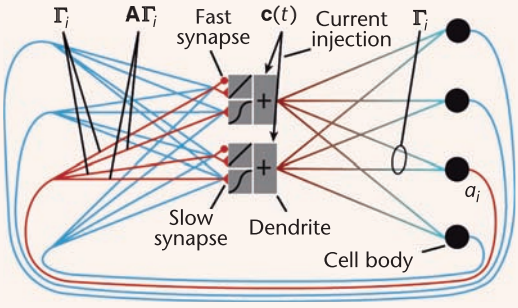

Figure C. A coordinated spiking network.

As it turns out, the differences between a coordinated spiking network (Figure C) and an NEF network (see the previous sidebar) are subtle. They look similar at first glance because there is a one-to-one correspondence between the former's dendrites' signals and the latter's adders' signals—both correspond to the D continuous signals ($\mathbf{x}$). However, there are two subtle differences. First, whereas a neuron's contribution to an adder's signal (specified by $\mathbf{d}_i$) is chosen to best approximate a desired transformation (defined by $\mathbf{f}$), a cell body's contribution to a dendrite's signal (specified by $\mathbf{\Gamma}_i$) is chosen randomly. Second, whereas the fraction of an adder's signal delivered to a neuron (specified by $\mathbf{e}_i$) is chosen randomly, the fraction of a dendrite's signal delivered to a cell body (specified by $\mathbf{\Gamma}_i$ on the figure's right side) is chosen to match the fraction of this cell body's signal delivered to that same dendrite (specified by $\mathbf{\Gamma}_i$ on the figure's left side). In the figure, $\mathbf{x}$, $\mathbf{\Gamma}_i$, and $\mathbf{c}$ are 2×1 while $\mathbf{A}$ is 2×2.

References

1. M. Boerlin, C.K. Machens, and S. Denève, "Predictive Coding of Dynamical Variables in Balanced Spiking Networks," *PLoS Computational Biology*, vol. 9, no. 11, 2013, article no. e1003258.
2. D. Thalmeier et al., "Learning Universal Computations with Spikes," *PLoS Computational Biology*, vol. 12, no. 16, 2016, article no. e1004895.

this work through a Long Range Science and Technology Grant (N00014-13-1-0419 & N00014-15-l-2827). O'Reilly Science Art helped with the figures.

References

1. "After Moore's Law," *The Economist*, 12 Mar. 2016.
2. N. Clément, X.L. Han, and G. Larrieu, "Electronic Transport Mechanisms in Scaled Gate-All-Around Silicon Nanowire Transistor Arrays," *Applied Physics Letters*, vol. 103, no. 26, 2013, article no. 263504.
3. D.J. Frank and H. Miki, "Analysis of Oxide Traps in Nanoscale MOSFETs Using Random Telegraph Noise," *Bias Temperature Instability for Devices and Circuits*, Springer, 2014, pp. 111–134.
4. C.A. Mead and L. Conway, *Introduction to VLSI Systems*, Addison Wesley, 1980.
5. C.E. Shannon and W. Weaver, *The Mathematical Theory of Communication*, Univ. Illinois Press, 1949.
6. B.V. Benjamin et al., "Neurogrid: A Mixed-Analog-Digital Multichip System for Large-Scale Neural Simulations," *Proc. IEEE*, vol. 102, no. 5, 2014, pp. 699–716.
7. W.R. Softky, "Fine Analog Coding Minimizes Information Transmission," *Neural Networks*, vol. 9, no. 1, 1996, pp. 15–24.
8. C. Allen and C.F. Stevens, "An Evaluation of Causes for Unreliability of Synaptic Transmission," *Proc. Nat'l Academy of Sciences*, vol. 91, no. 22, 1994, pp. 10380–10383.
9. S. Panzeri et al., "The Role of Spike Timing in the Coding of Stimulus Location in Rat Somatosensory Cortex," *Neuron*, vol. 29, no. 3, 2001, pp. 769–777.

10. E.A. Vittoz, "Future of Analog in the VLSI Environment," *Proc. Int'l Symp. Circuits and Systems*, vol. 2, 1990, pp. 1347–1350.
11. R. Sarpeshkar, "Analog versus Digital: Extrapolating from Electronics to Neurobiology," *Neural Computation*, vol. 10, 1998, pp. 1601–1638.
12. H. Reyserhove, N. Reynders, and W. Dehaene, "Ultra-Low Voltage Datapath Blocks in 28nm UTBB FD-SOI," *IEEE Asian Solid-State Circuits Conf.* 2014, pp. 49–52.
13. W. Wattanapanitch, M. Fee, and R. Sarpeshkar, "An Energy-Efficient Micropower Neural Recording Amplifier," *IEEE Trans. Biomedical Circuits and Systems*, vol. 1, no. 2, 2007, pp. 136–147.
14. C.A. Mead, *Analog VLSI and Neural Systems*, Addison Wesley, 1989.
15. C.A. Mead, "Neuromorphic Electronic Systems," *Proc. IEEE*, vol. 78, no. 10, 1990, pp. 1629–1636.
16. A.J. Martin, "Programming in VLSI: From Communicating Processes to Delay-Insensitive Circuits," *UT Year of Programming Series*, Addison-Wesley, 1990, pp. 1–64.
17. A.J. Martin and M. Nystrom, "Asynchronous Techniques for System-on-Chip Design," *Proc. IEEE*, vol. 94, no. 6, 2006, pp. 1089–1120.
18. T.S. Clawson et al., "Spiking Neural Network (SNN) Control of a Flapping Insect-Scale Robot," *Proc. IEEE Conf. Decision and Control*, 2016, pp. 3381–3388.
19. J. Dethier et al., "Design and Validation of a Real-Time Spiking Neural-Network Decoder for Brain Machine Interfaces," *J. Neural Eng.*, vol. 10, no. 3, 2013, article no. 036008.
20. R. Silver et al., "Neurotech for Neuroscience: Unifying Concepts, Organizing Principles, and Emerging Tools," *J. Neuroscience*, vol. 27, no. 44, 2007, pp. 11807–11819.
21. R.J. Vogelstein et al., "Dynamically Reconfigurable Silicon Array of Spiking Neurons with Conductance-Based Synapses," *IEEE Trans. Neural Networks*, vol. 18, no. 1, 2007, pp. 253–265.
22. J. Lin et al., "Programmable Connections in Neuromorphic Grids," *Proc. 49th IEEE Midwest Symp. Circuits and Systems*, 2006, pp. 80–84.
23. M. Mahowald, *An Analog VLSI System for Stereoscopic Vision*, Kluwer Academic Publishers, 1994.
24. K.A. Boahen, "Point-to-Point Connectivity between Neuromorphic Chips Using Address-Events," *IEEE Trans. Circuits & Systems II*, vol. 47, no. 5, 2000, pp. 416–434.
25. V. Braitenberg, "Brain Size and Number of Neurons: An Exercise in Synthetic Neuroanatomy," *J. Computational Neuroscience*, vol. 10, no. 1, 2001, pp. 71–77.
26. V.B. Mountcastle, "Modality and Topographic Properties of Single Neurons of Cat's Somatic Sensory Cortex," *J. Neurophysiology*, vol. 20, no. 4, 1957, pp. 408–434.
27. D.H. Hubel and T.N. Wiesel, "Ferrier Lecture: Functional Architecture of Macaque Monkey Visual Cortex," *Proc. Royal Soc. London, Series B, Biological Sciences*, vol. 198, no. 1130, 1977, pp. 1–59.
28. R.J. Douglas and K.A. Martin, "A Functional Microcircuit for Cat Visual Cortex," *J. Physiology*, vol. 440, 1991, pp. 735–769.
29. P. Merolla et al., "A Multicast Tree Router for Multichip Neuromorphic Systems," *IEEE Trans. Circuits and Systems*, vol. 61, no. 3, 2014, pp. 820–833.
30. A.G. Andreou and K.A. Boahen, "A 590,000 Transistor 48,000 Pixel, Contrast Sensitive, Edge Enhancing, CMOS Imager-Silicon Retina," *Proc. 16th Conf. Advanced Research in VLSI*, 1995, pp. 225–240.
31. B.V. Benjamin et al., "A Superposable Silicon Synapse with Programmable Reversal Potential," *Annual Int'l Conf. IEEE Eng. and Medicine in Biology Soc.*, 2012, pp. 771–774.
32. K.M. Hynna and K. Boahen, "Thermodynamically-Equivalent Silicon Models of Ion Channels," *Neural Computation*, vol. 19, no. 2, 2007, pp. 327–350.
33. P. Gao, B.V. Benjamin, and K. Boahen,"Dynamical System Guided Mapping of Quantitative Neuronal Models onto Neuromorphic Hardware," *IEEE Trans. Circuits and Systems I*, vol. 59, no. 10, 2012, pp. 2383–2394.
34. P. Merolla et al., "A Million Spiking-Neuron Integrated Circuit with a Scalable Communication Network and Interface," *Science,* vol. 345, no. 6197, 2014, pp. 668–673.
35. S.A. Cassidy et al., "Real-Time Scalable Cortical Computing at 46 Giga-Synaptic OPS/Watt with ~100× Speedup in Time-to-Solution and ~100,000× Reduction in Energy-to-Solution," *Proc. Int'l Conf. High Performance Computing, Networking, Storage and Analysis*, 2014, pp. 27–38.
36. C. Eliasmith and C.H. Anderson, *Neural Engineering: Computation, Representation, and Dynamics in Neurobiological Systems*, MIT Press, 2003.
37. S. Choudhary et al., "Silicon Neurons That Compute," *Int'l Conf. Artificial Neural Networks*, LNCS vol. 7552, Springer, 2012, pp. 121–128.
38. O. Khatib, "A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation," *IEEE J. Robotics and Automation*, vol. 3, no. 1, 1987, pp. 43–53.
39. S. Menon et al., "Controlling Articulated Robots in Task-Space with Spiking Silicon Neurons," *IEEE Int'l Conf. Biomedical Robotics and Biomechatronics*, 2014, pp. 181–186.

40. D.H. Goldberg, G. Cauwenberghs, and A.G. Andreou, "Probabilistic Synaptic Weighting in a Reconfigurable Network of VLSI Integrate-and-Fire Neurons," *Neural Network*, vol. 14, nos. 6–7, 2001, pp. 781–793.
41. C. Eliasmith et al., "A Large-Scale Model of the Functioning Brain," *Science*, vol. 338, no. 6111, 2012, pp. 1202–1205.
42. C. Eliasmith, *How to Build a Brain: A Neural Architecture for Biological Cognition*, Oxford Univ. Press, 2013.
43. T. Bekolay et al., "Nengo: A Python Tool for Building Large-Scale Functional Brain Models," *Frontiers in Neuroinformatics*, vol. 7, no. 48, 2013; doi:10.3389/fninf.2013.00048.
44. M. Boerlin, C.K. Machens, and S. Denève, "Predictive Coding of Dynamical Variables in Balanced Spiking Networks," *PLoS Computational Biology*, vol. 9, no. 11, 2013, article no. e1003258.
45. D. Thalmeier et al., "Learning Universal Computations with Spikes," *PLoS Computational Biology*, vol. 12, no. 16, 2016, article no. e1004895.

Kwabena Boahen is a professor of bioengineering at Stanford University, with a joint appointment in electrical engineering. He directs Stanford's Brains in Silicon Laboratory, which develops silicon integrated circuits that emulate the way neurons compute, linking the seemingly disparate fields of electronics and computer science with neurobiology and medicine. Boahen is an IEEE Fellow. Contact him at boahen@stanford.edu.