nature electronics

ARTICLES

https://doi.org/10.1038/s41928-020-0436-6

Check for updates

Power-efficient combinatorial optimization using intrinsic noise in memristor Hopfield neural networks

Fuxi Cai[1,2,6], Suhas Kumar[1,6], Thomas Van Vaerenbergh[1,6], Xia Sheng[1], Rui Liu[1,3], Can Li[1], Zhan Liu[1], Martin Foltin[1], Shimeng Yu[4], Qiangfei Xia[5], J. Joshua Yang[5], Raymond Beausoleil[1], Wei D. Lu[2] and John Paul Strachan[1]✉

To tackle important combinatorial optimization problems, a variety of annealing-inspired computing accelerators, based on several different technology platforms, have been proposed, including quantum-, optical- and electronics-based approaches. However, to be of use in industrial applications, further improvements in speed and energy efficiency are necessary. Here, we report a memristor-based annealing system that uses an energy-efficient neuromorphic architecture based on a Hopfield neural network. Our analogue–digital computing approach creates an optimization solver in which massively parallel operations are performed in a dense crossbar array that can inject the needed computational noise through the analogue array and device errors, amplified or dampened by using a novel feedback algorithm. We experimentally show that the approach can solve non-deterministic polynomial-time (NP)-hard max-cut problems by harnessing the intrinsic hardware noise. We also use experimentally grounded simulations to explore scalability with problem size, which suggest that our memristor-based approach can offer a solution throughput over four orders of magnitude higher per power consumption relative to current quantum, optical and fully digital approaches.

Using memristors, or two terminal nonvolatile memories[1], to build higher-performance computing systems is an attractive approach to mitigating the imminent end of Moore's law[2] and addressing the current challenges of stalled power reduction and clock speed-up caused by the end of Dennard scaling[3]. Computing-intensive tasks, such as the class of non-deterministic polynomial-time (NP)-hard problems, are important in a range of applications, including traffic management, airline scheduling, wiring within electronic chips and gene sequencing. To tackle this class of problems, a variety of annealing-inspired computing accelerators based on different technology platforms have been proposed: adiabatic quantum annealing with super-conducting qubits[4], Boltzmann machines[5] or classical annealing[6] in memristor electronics; coherent Ising machines using a fibre-based optical parametric oscillator[7] or integrated optics[8–10]; and purely digital implementations using field programmable gate arrays[11] or graphics processing units (GPUs)[12].

These platforms emulate meta-heuristic algorithms that solve quadratic unconstrained binary optimization (QUBO) problems, with the ambition to more rapidly solve large problem sizes beyond those currently tackled by exact algorithms. Exact methods, such as branch-and-bound, are often limited to a few hundred variables per central processing unit (CPU) core for some typical QUBO instances[13,14] (see also Supplementary Discussion section 1). At the algorithmic level, hybrid approaches have been proposed that allow subdivision of industrial-scale problems into QUBO problems compatible with today's accelerator sizes[15,16]. Consequently, there is an incentive to find efficient accelerators compatible with this hybrid eco-system. Furthermore, quantum annealers have not yet demonstrated speed-ups for industrial applications, and bring high costs and complexity to deployment due to cryogenic cooling, adding a stronger motivation to leverage classical physics instead[7,11].

In this article, we report a hybrid analogue–digital approach to solve combinatorial optimization problems. This memristor-based annealing system uses an energy-efficient neuromorphic architecture based on a Hopfield neural network (HNN). An HNN is a fully connected recurrent neural network without self-connections, where a state of a neuron is dependent on the input received from all other neurons[17,18]. HNNs were initially proposed to implement associative memory for pattern recognition[17], and research on memristor-based HNNs (mem-HNNs) have explored this type of application, along with other broad applications such as analogue–digital converters[19], many of which require only a sparse connectivity[6,20–22] (connectivity being the degree of non-zero weights on the connection matrix). Here we use the HNN model to solve computationally hard QUBO problems[23–25], thanks to its ability to dynamically find minima (global or local) of implicitly defined cost-functions, referred to as the problem energy. Optimization problems are encoded in a weights matrix representing the set of objectives and/or constraints (for more details see Supplementary Discussion section 2).

The mem-HNN

Our hybrid analogue-digital mem-HNN architecture is mathematically equivalent to a conventional discrete-time binary HNN augmented with a noise contribution:

[1]Hewlett Packard Labs, Hewlett Packard Enterprise, Palo Alto, CA, USA. [2]Department of Electrical Engineering, University of Michigan, Ann Arbor, USA. [3]Department of Electrical and Electronics Engineering, Arizona State University, Tempe, AZ, USA. [4]School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. [5]Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, MA, USA. [6]These authors contributed equally: Fuxi Cai, Suhas Kumar, Thomas Van Vaerenbergh. ✉e-mail: john-paul.strachan@hpe.com

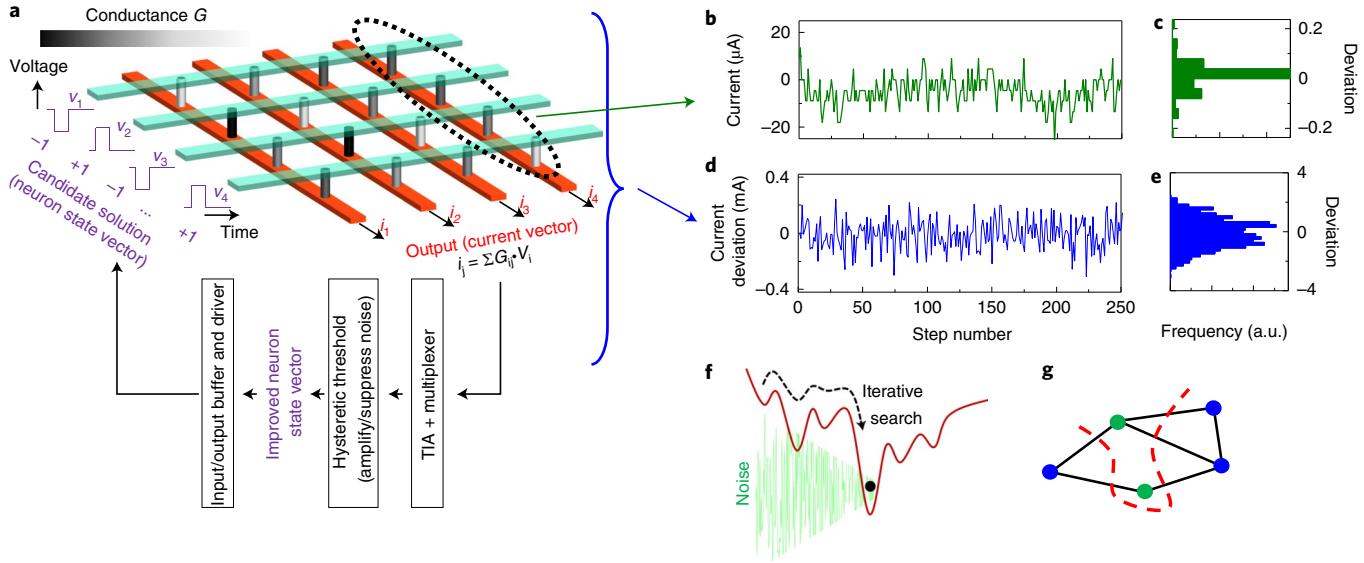

Fig. 1 | Overview of the mem-HNN, experimental noise measurements and the max-cut problem. a, Schematic of the mem-HNN design including in-memory vector–matrix multiplication in a memristor crossbar array and peripheral circuits (more details in Supplementary Discussion section 7). TIA, transimpedance amplifier. **b**, Dynamical noise in a 60 × 60 array. Current measured from the memristor crossbar array in a single column for repeated identical inputs, demonstrating noise originating from device-level and measurement fluctuations; **c**, the corresponding histogram. **d**, Analogue circuit errors in a 60 × 60 crossbar array. Deviations in current (compared to an ideal calculation) over many mem-HNN calculation steps (via recurrent feedback of updated outputs) measured over a single column, accompanied by a histogram in **e**. Analogue crossbar array deviations dominate in amplitude over the dynamical noise in **b**. The right ordinate in **c** is linked to the left ordinate in **b**, thereby providing a conversion between current and integer representations. Similar scales are provided in **d** and **e**. **f**, Schematic energy landscape of a typical NP-hard problem, also illustrating simulated annealing. **g**, A typical max-cut problem. The dashed red line cuts the graph into two sets of nodes, coloured blue and green, such that the edges between the different-coloured nodes are maximized.

$$u_i = \sum_{j \neq i} W_{ij} v_j, \quad v_i = \begin{cases} +1 & \text{if } (u_i + \eta_i) \geq \theta_i \\ -1 & \text{if } (u_i + \eta_i) < \theta_i \end{cases} \qquad (1)$$

where v is the state of the neuron, u is the weighted feedback accumulated at the neurons at the next time step, W is the zero-diagonal symmetric weights matrix, θ_i is a threshold and the vector η represents the added noise that is needed to improve convergence to optimal solutions. The binary threshold function serves as an activation function, which essentially implements a nonlinear filter to process the recurrent weighted feedback.

The most power-intensive task here, namely vector-matrix multiplication, can be implemented using crossbars of memristors, particularly oxide-based two-terminal devices with tuneable analogue resistance levels[1], exploiting Ohm's and Kirchoff's laws to obtain the vector–matrix product of input vectors and a stored weights matrix W (Fig. 1a). This enables highly efficient computations performed in the analogue domain[26] and substantially reduces data movement as operations are performed directly in memory[27,28]. The input vector is stored in the input/output buffer driving the rows of the array.

To perform the vector–matrix multiplication, a switch matrix and driver circuitry, instead of a decoder, is required to enable all rows of the memristor crossbar in parallel. For a given cycle, a multiplexer is used to select one (or several) columns of the crossbar of memristors to calculate dot products for the neurons in a single operation. The output of the dot products directly feed into the nonlinear filters to perform the threshold function. The results are sent back to inut/output buffers to update the binary status of the neurons, which are used as inputs for the next cycle (more details in Supplementary Discussion section 7).

In this work, we show that a memristor-based optimization network, a mem-HNN, is extremely fast, energy-efficient, and can leverage intrinsic analogue noise of the system to improve both solution quality and efficiency. Figure 1b-e shows experimental data illustrating the two major intrinsic effects in our system that can supply noise for equation (1). Figure 1b plots the time-varying current output of a single column with a repeated identical input. Dynamical noise is observed, and this is due to well known conductance fluctuations in the nanoscale memristive devices in that column undergoing atomic fluctuations and/or electronic trapping/de-trapping effects[29].

Figure 1d, where the input was allowed to vary, shows additional deviations in the measured analogue output compared to the precise expected output due to crossbar array non-idealities. This includes non-zero wire resistances interacting with the finite resistances of the memristor cells and the nonlinear current–voltage characteristics of the devices.

These are highly nonlinear interactions that are input-dependent and we observe here (see also Supplementary Figs. 4 and 7) and in previous work that it leads to Gaussian-shaped error distributions around the ideal target result. When used in appropriate architectures, such analogue nonlinear effects have been shown to enable generation of number sequences that pass National Institute of Standards and Technology (NIST) randomness tests[30].

Further, we show that over the course of typical HNN operations, analogue output deviations are identical in spectral distribution to white noise. We thus refer to both of these as intrinsic noise sources in our analogue system throughout this article, while Fig. 1c,e shows that the crossbar array noise dominates in size while also supporting fast operating speeds (Supplementary Discussion section 9.4 and Supplementary Fig. 13). However, this technique can, in principle, harness other noise sources depending on the hardware design. In Fig. 1b–e, we also provide a graphical illustration of the conversion between the experimentally measured current and the unit-free representation in the same scale as the weights matrix and the input and output integer vectors.

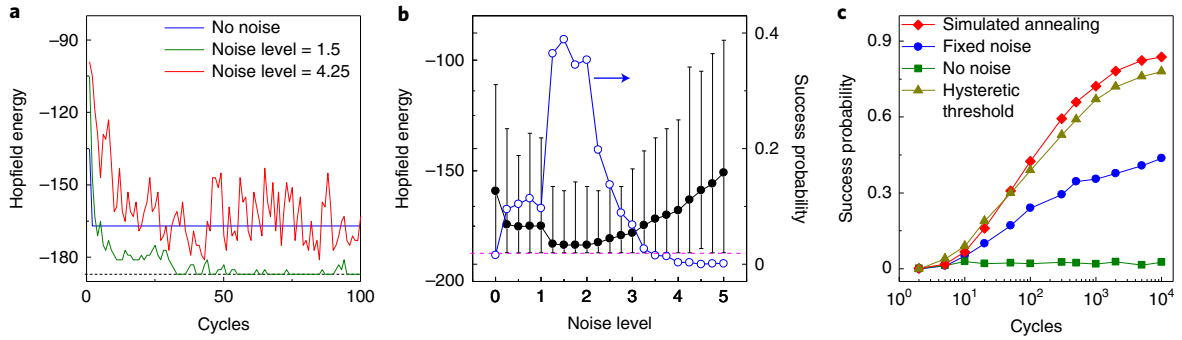

Fig. 2 | Utility of noise in HNNs to obtain better solutions, illustrated for 60-node instances of dense max-cut problems. a, The influence of different noise levels on the outcome of the final stable state. Three different noise levels are shown: 0, 1.5 and 4.25. The dashed horizontal line represents the minimum possible energy. **b**, Mean and range of max-cut energy results with different fixed noise levels. At each noise level, 1,000 instances with different random initial states are investigated. The error bars illustrate the maximum and minimum max-cut of all instances at the corresponding noise level. The success probability—the probability of achieving the known globally optimal solution—is also plotted as blue open circles versus noise level. The dashed pink horizontal line is the global minimum for the problem. **c**, Comparison of the effect of using a quadratic superlinear decaying noise (simulated annealing), a fixed noise, a fixed noise with a hysteretic threshold, and no noise.

For NP-hard problems, as problem sizes scale up linearly, there are at least two main challenges with today's best known algorithms and processors: (1) resource consumption (time to completion, energy consumed, and so on) grows exponentially and (2) an increasingly complex problem-energy landscape evolves that can contain multiple local minima, a particular challenge when globally optimal solutions are desired (Fig. 1f). When a noise-free ($\eta = 0$) discrete-time HNN is updated (one node is chosen and updated at a time), the update rule of equation (1) assures that the HNN evolves in a way that only reduces the following energy function:

$$E = -\frac{1}{2}\sum_{i,j}^{N} W_{ij}v_i v_j + \sum_{i}^{N} \theta_i v_i \qquad (2)$$

Through proper choice of the weight matrix W, an arbitrary optimization problem[23] can be encoded and solved with the HNN, which will eventually converge to a fixed point. However, noiseless discrete-time HNNs have not gained widespread adoption because, among other reasons, they could not achieve the energy ascent required to escape local minima, preventing them from solving large optimization problems with complex energy landscapes. Simulated annealing is a well known technique to address this issue[31–33], wherein a stochastic or uncorrelated process is used to perturb the state of a system, thereby enabling it to achieve an energy ascent sufficient to escape local minima. As the system approaches the global minimum, the magnitude of the noise is tuned down to trap the system in the optimal solution (Fig. 1f).

Implementing controlled pseudo-random number generation to achieve simulated annealing typically requires elaborate digital circuits with unfavourable size, power consumption and latency properties[34]. In contrast, here we propose a scheme to generate, modulate and utilize analogue noise arising from the crossbar circuit computations or peripheral elements, and experimentally demonstrate its feasibility. These can implement processes that mimic simulated annealing and are described below.

Given the diversity of existing and developing annealing or energy-minimization approaches for constrained optimization, there is a strong need for benchmarking[24]. We use the NP-hard graph problem "max-cut"[23] as a benchmarking task, as it has already been extensively used in recent literature[7,8,11,12]. As illustrated in Fig. 1g, in the max-cut problem, the objective is to obtain a partitioning of the nodes of a connected graph, such that the number of connections between the nodes of the two partitions is maximized. This problem has relevant applications across the industry, particularly in efficient circuit routing and minimizing the number of vias in integrated circuits.

Performance enhancement with controlled noise injection

The level of noise present in equation (1) that governs our mem-HNN plays a vital part in the network's ability to find globally optimal solutions. Figure 2a investigates the effect of noise on the mem-HNN, starting with simulations at three different noise levels (0, 1.5 and 4.25) but otherwise using identical conditions. We used unweighted max-cut instances provided in the Biq Mac (binary quadratic and max-cut) library for dense (50%) connectivity and node sizes of $N=60$ and larger[7,35]. We note that noise and energy levels have a common arbitrary unit scale that can be particular to the graph problem and connection density. As shown in Fig. 2a, without any noise the Hopfield network converges to a sub-optimal solution within just a few cycles. This is because such dense graphs have a complex energy landscape and a conventional HNN can easily get stuck at local minima. Utilizing a larger noise level, say 4.25, it is very easy to overcome barriers in the energy landscape and get out of local minima, but the downside is continued large fluctuations, even after discovering an optimal energy solution. Therefore, finding an appropriate noise level is a critical task. Here noise refers to perturbations to the system and fluctuations refer to the system's response.

To verify the existence of an optimal noise level, we ran a suite of simulations varying the noise levels (Fig. 2b) and starting states, while tracking the best, worst and mean energy of solution. In Fig. 2b, by varying the noise level from 0 to 5, we can observe that the optimal level is around 1.5, which not only has the best mean value for the max-cut energy, but also less variability and a peak in the probability of reaching the known globally optimal solution, which we term here the 'success probability'. We also observe that as the noise level goes to zero (or well above 2), the success probability can tend to zero with the mean max-cut energy also growing unfavourably.

Based on our studies in Fig. 2a,b, we know that noise is beneficial for escaping local minima, but converging to a final optimal solution is challenged if the network continues to fluctuate. To simultaneously harness the benefits of noise while improving network stability, a good approach[36] is to use a decaying noise profile that progressively reduces in magnitude in the final cycles so that the network may hold its state. This approach is leveraged in the

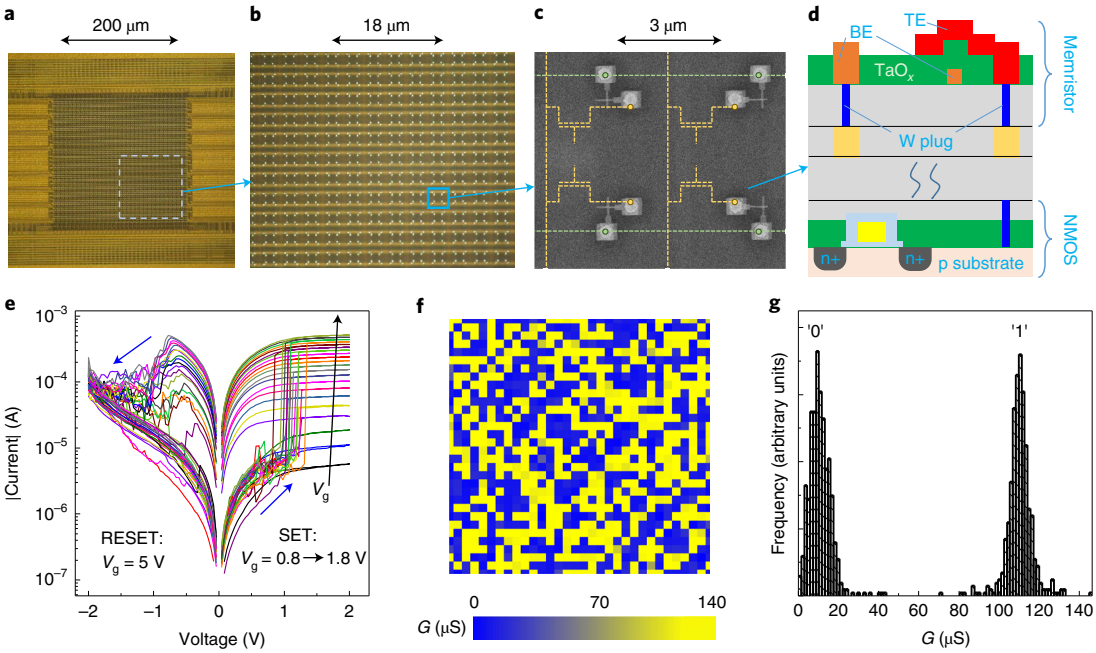

Fig. 3 | Experimental implementation of the mem-HNN. a, Optical micrograph of the memristor crossbar chip used. **b**, Magnified optical micrograph of **a**, where the devices in the crossbar array may be visible (a more magnified version of this image is provided in Supplementary Fig. 5). **c**, Scanning electron micrograph of four nanoscale tantalum oxide (TaO_x) memristors within the crossbar array. Interconnects and access transistors beneath the surface are indicated using dashed lines. **d**, Schematic cross-section of the structure of a single memristor along with its buried access transistor, which was experimentally fabricated. Additional chip images are provided in Supplementary Fig. 5. Top electrode (TE), bottom electrode (BE) and n-type metal-oxide semiconductor (NMOS) transistor layers are marked. **e**, Current versus voltage behaviour of a typical memristor, including SET (conductance increase) and RESET (conductance decrease) processes, exhibiting tight control over the conductances during the SET process via control of the gate voltage (V_g) of the control transistor associated with every device. Blue arrows indicate the direction of the curves. **f**, Experimentally programmed conductance weights matrix representing a problem of size $N = 32$; **g**, the associated histogram of conductance distributions.

simulated annealing algorithm, in which an initially high 'temperature' for the system is slowly 'cooled'[31]. Related approaches with modified annealing schedules (noise profile across iterations towards a solution) have been used, particularly for increasing parallelism and reducing time spent in local minima[37] (see more discussion in Supplementary Discussion section 6).

To investigate the effects of the noise profile, we repeated the tests of Fig. 2b with different cycle-dependent noise profiles (see also Supplementary Discussion section 5). In Fig. 2c, we compare no noise, a fixed (constant amplitude) noise, a decaying noise amplitude, and a fixed noise with a variable threshold, all plotted against increasing total cycles. In all cases, the amplitude of the noise was separately optimized for best solution accuracy. The network showed the best performance with a simulated annealing approach where a quadratically decaying noise versus cycle number was applied (Supplementary Fig. 3). As noted earlier, the benefit here is that the system slowly becomes stabilized at an optimal solution. Having no noise leads to the worst result, while a fixed finite noise gives substantial improvements. Interestingly, tuning the threshold function criteria (θ_i in equation (1)) has emerged as an effective tool with which either to enhance or suppress the intrinsic noise. This effectively enables solution convergence at the level of simulated annealing, but without having to tune the intrinsic noise of the system directly, which is typically the case with most analogue memristor networks. This idea was adopted in our experimental systems to mimic simulated annealing without requiring any other noise source than the threshold-modulated intrinsic noise from the crossbar array, detailed in the following section.

Experimental implementation and control of intrinsic noise

We implemented the mem-HNN by programming binary weight matrices representing max-cut problems of differing sizes ($N = 8$, 16, 32) and of about 50% density on a chip containing crossbar arrays of nanoscale tantalum oxide memristors (Fig. 3a–d) integrated with 180-nm Complementary Metal Oxide Semiconductor (CMOS) node control circuits, with each memristor paired with an access transistor. Each memristor was capable of representing a weight within the weights matrix via reprogrammable nonvolatile conductance states (Fig. 3e), further detailed elsewhere[38]. To illustrate the translation from a weights matrix containing binary bits to a conductance matrix, we display the conductance matrix along with the experimental distribution of the conductances (Fig. 3f,g) that represented a binary max-cut problem of size $N = 32$ (additional details in Supplementary Fig. 6). The core vector–matrix multiplication operations of equation (1) were performed within the memristor crossbar arrays, with peripheral sensing circuits implemented in printed circuit boards, and the nonlinear/hysteretic filtering and energy calculations performed within a control computer (detailed in Supplementary Fig. 7). Our mem-HNN performs the computations of equation (1) in the analogue domain in a single clock cycle, offering high parallelism and power-efficient computations[39].

As described earlier and shown in Fig. 1, an important aspect of the mem-HNN platform is the ability to harness intrinsic analogue noise in contrast to precise digital computing. However, this acts as a fixed source of noise with a constant amplitude, whereas a dynamically tunable (enhanced or suppressed) noise source is needed. To address this, we introduce an additional peripheral component in

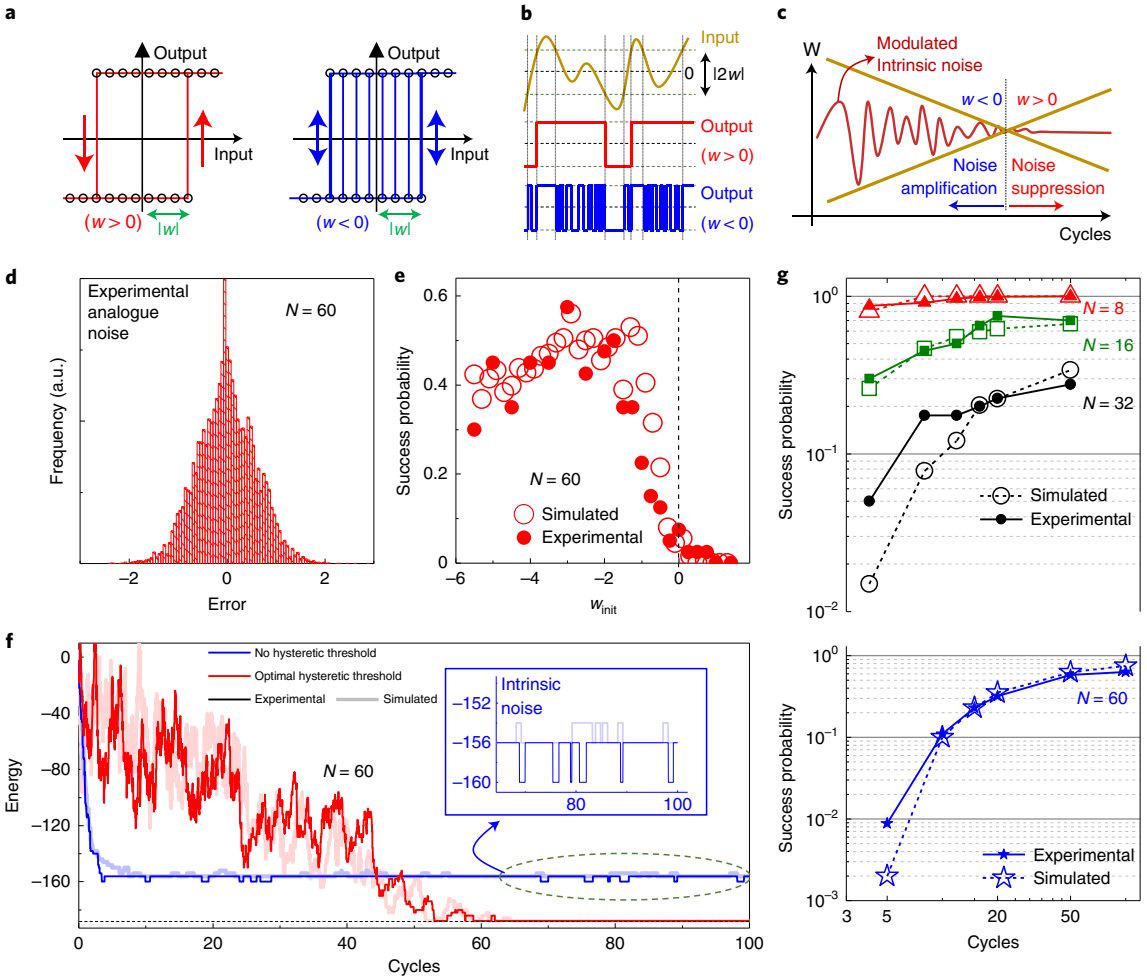

Fig. 4 | Experimental data from the mem-HNN. a, The scheme of hysteretic threshold function for two different cases: $w > 0$ and $w < 0$. **b**, Illustration of outputs corresponding to the two cases in **a** for a given input. **c**, Illustration of the effect of varying w. $w < 0$ amplifies intrinsic noise. $w > 0$ does not generate but instead suppresses intrinsic noise. **d**, Experimental measurement of the errors (intrinsic noise) in a 60 × 60 matrix for a problem of size $N = 60$, in the same units as the input vector. **e**, Success probability plotted against initial w (for a fixed final positive w of +1.4) shows that the highest success probability is achieved for a negative initial w of roughly −3, qualitatively consistent with **c**. Data from experiments and circuit simulations are shown superimposed on each other, exhibiting close agreement. **f**, Energy descent for a problem of size $N = 60$ for two cases: no hysteretic threshold used, and hysteretic threshold used with w varied from −4.5 to +1.4. As illustrated in **c**, varying w from negative to positive mimics simulated annealing and produces the optimal solution (minimum energy corresponding to the dashed horizontal line). Data from experiments and circuit simulations obtained for identical initial conditions are shown on top of each other, exhibiting close agreement. The inset is a magnified plot of the data contained in the dashed ellipse (with the same units as the main axes), highlighting the presence of intrinsic noise, originating from **d**. **g**, Success probabilities for four problems of different sizes ($N = 8, 16, 32, 60$) for varying number of cycles using a hysteretic threshold. Experimental results are shown alongside circuit simulations (which include a model for intrinsic noise in the crossbar array) for an identical process of solving the same problems, exhibiting close agreement.

the system to perform a hysteretic threshold function (Fig. 4a–c) with a finite width w, described by

$$\theta_{\mathrm{hys}:i,t} = \theta_{i,t} - (w_t)v_{i,t}, \quad (3)$$

where $\theta_{\mathrm{hys}:i,t}$ is the hysteretic threshold defined at the location i and timestep t, defined by hysteretic width w. When $w > 0$, noise inputs near zero magnitude within a width of w are held at their previous states, thereby suppressing a fluctuating response from noisy inputs. A larger positive w leads to a larger suppression of noise when $w > 0$. This is functionally identical to a Schmitt trigger. When $w < 0$, for all inputs near zero magnitude within a width of w, the output fluctuates between the two allowed binary states. This hysteretic threshold can be implemented with a simple circuit (see Supplementary Discussion sections 4 and 9). This generates fluctuations in a noise-free system, or can amplify the fluctuations due to intrinsic noise in a tunable manner, thereby enabling the system to escape local minima (Supplementary Discussion section 3). Therefore, continuously tuning w from a negative to a positive value as the Hopfield network converges toward a solution is similar to the process in simulated annealing going from effectively hot to cold temperatures to tune the level of noise and the resulting fluctuations of the system's response, as illustrated in Fig. 4c. Strong support for the similar effectiveness of simulated annealing and tuning of the hysteretic threshold is shown in Fig. 2c (simulated annealing and hysteretic threshold). We demonstrate the presence of intrinsic noise by plotting the experimental distribution of the calculation errors (Fig. 4d) that have origins in the various sources of noise, some of which may be dynamic and others static, but the emphasis here is our ability to harness and modulate all of these sources of

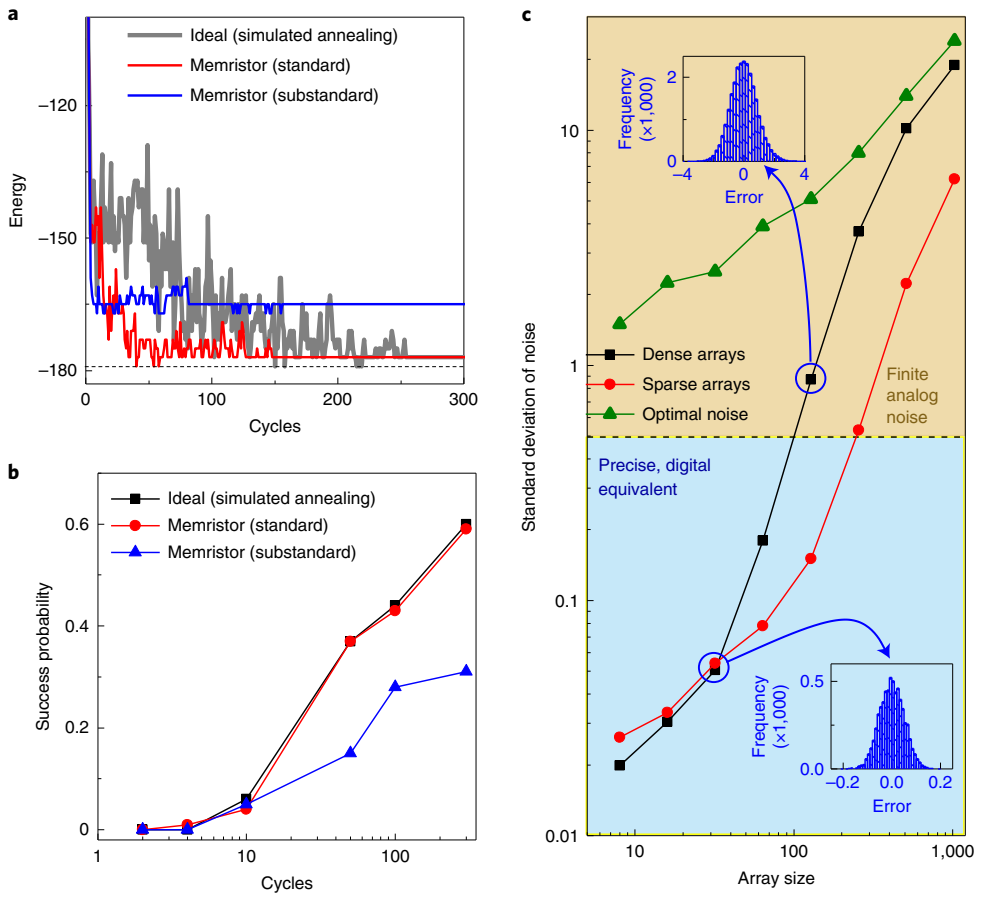

Fig. 5 | Circuit and array simulations of the mem-HNN. a, Circuit simulations of the mem-HNN showing energy descent, which includes both the ideal results obtained using simulated annealing (by employing a software-defined random number generator) along with circuit simulations employing a hysteretic threshold using routine (standard) memristor parameters and poor ones (substandard). The dashed horizontal line represents the minimum possible energy. It can be seen in **b** that the overall success probability with analogue memristors closely matches the exact results, although poor memristor parameters can degrade the computations. As array size grows, increasing non-idealities can lead to noisy computations that no longer match the exact ideal results. Simulations in **a** and **b** were performed for the first two problems of size $N = 60$ in the Biq-Mac library. **c**, The scaling of errors (standard deviation of noise) with array size (the number of components in the array being the square of the array size) for dense and sparse graphs, along with the amount of optimal noise needed in solving graph problems. It is seen that the intrinsic noise in analogue arrays is still below that needed up to arrays of 1024 × 1024, giving a favourable scalability of the mem-HNN. The insets in **c** show the error distributions corresponding to data points indicated by blue arrows for 32 × 32 and 128 × 128 (more details in Supplementary Discussion section 8).

noise via a dynamically tuned hysteretic threshold. Figure 4e shows that by tuning the hysteretic threshold w, the success probability can be improved.

In addition, Fig. 4f demonstrates an experimental implementation of the above idea by solving a problem of size $N = 60$ via energy minimization. We first allowed the Hopfield network to find a solution without using a hysteretic threshold ($w = 0$), which resulted in convergence to a sub-optimal solution (higher energy), as expected with Hopfield networks. We also use this data to highlight that without addition of any external noise, the system continually exhibited noisy behavior even after convergence (inset in Fig. 4f). This originates from the intrinsic noise in the system, which can be leveraged using a hysteretic threshold. In a subsequent solution of the same problem, When the width of the hysteretic threshold was varied from an initial to a final width (w_{init} to w_{final}) of -4.5 to +1.4, there was a presence of large amplified intrinsic noise in the initial cycles (for $w < 0$) and entirely suppressed noise towards the final cycles (for $w > 0$), again. This behavior resembles simulated annealing achieved via injection of decaying noise. The advantage of using a hysteretic threshold is that it amplifies intrinsic noise in the system using simple comparator-based circuits, as against needing elaborate pseudo-random number generation circuits for simulated annealing. Also as expected, this form of annealing using the hysteretic threshold resulted in the best solution (lowest energy).

To compare with experiments, we used circuit simulations, where circuit non-idealities and intrinsic analogue noise in memristor arrays can be accurately captured in circuit models by including the parasitic resistances, finite ON/OFF ratio, device nonlinearities and programming errors[39]. Simulations of such a circuit model of the experimental system exhibited very close agreement with the experimental data (Fig. 4f). Figure 4g is a collection of experimental data and associated circuit simulations plotting the success probabilities against the number of cycles for solving four problems of different sizes, using a hysteretic threshold. The specific problem of size $N = 60$ had a higher success probability than that of size $N = 32$, which was probably a difficult instance, and illustrates the range of difficulties optimization problems can present. These experiments support the validity of our techniques for harnessing and modulating the noise needed in a Hopfield network. The close match by circuit simulations also validates our simulation framework as well grounded in capturing the experimental system, and enables us to use this framework to project the larger-scale performance of the mem-HNN.

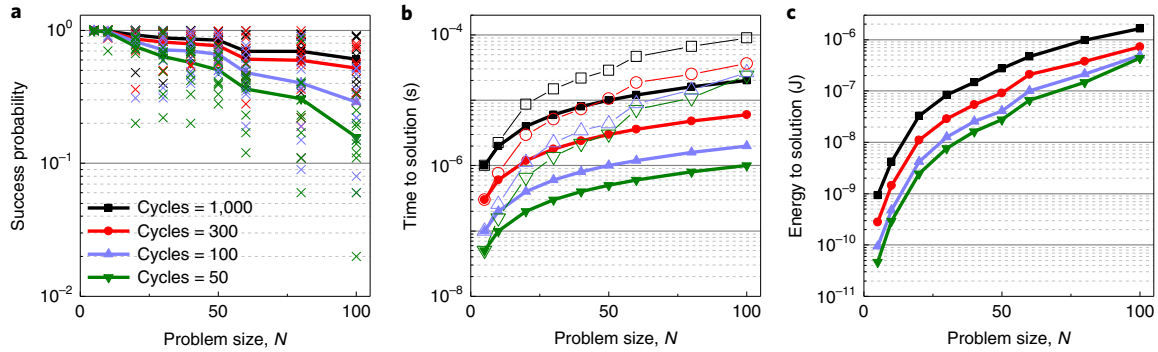

Fig. 6 | Simulations of the mem-HNN solving dense instances of the max-cut problem with varying graph sizes. a, Probability of reaching the globally optimal solution versus problem size. For each problem size, ten separate graph instances are computed and the total number of cycles used is varied from 50 to 1,000. **b**, The time to solution with 99% probability of success. Memristor crossbars can either be utilized in sequential runs, operating multiple times with differing starting states to ensure 99% success (open symbols). Or, multiple crossbars can run in parallel (filled symbols) at the cost of larger area. **c**, The energy required to reach 99% optimal solutions is computed as a function of problem size. Larger problems require larger memristor crossbars, more peripheral circuitry, and also more parallel crossbars to ensure a 99% success probability is attained.

With regard to device parameters, Fig. 5a,b shows circuit simulations of a 60 × 60 array using routinely manufacturable wire resistances and memristor parameters ('standard' values, corresponding to $R_{ON} = 10\,k\Omega, R_{OFF} = 1\,M\Omega, R_{wire} = 1\,\Omega$ per block) as well as parameters substantially more non-ideal ('sub-standard', corresponding to $R_{ON} = 2\,k\Omega, R_{OFF} = 100\,k\Omega, R_{wire} = 1\,\Omega$ per block), where the biggest effect is the 5× lower R_{ON}, which is known to degrade computations in analogue crossbars[40]. Both cases utilize a hysteretic threshold, compared to fully ideal ('ideal') results using simulated annealing. Importantly, additional data in Supplementary Figs. 9 and 10 demonstrate that the hysteretic threshold used in the experimental implementation is functionally identical to the more traditional simulated annealing scheme. A good match is seen between the ideal results and analogue circuit results with standard parameters. It is seen in Fig. 5b that analogue results can give success probabilities very comparable to those of the ideal results, whereas it is also evident that dramatically worse device and circuit parameters degrade the results, yet without a sharp drop.

As highlighted in the above studies, intrinsic variability and finite error distributions in analogue computations can be an advantageous resource, but too much noise may compromise solution convergence. The non-idealities that generate error distributions can grow with array size owing to more interactions and longer worst-case wire resistances across the rows and columns that can be amplified by other non-linearities and non-idealities in the array[40]. To quantify the limits on the scalability of our mem-HNN system, we simulated a vast ensemble of analogue computations performed in memristor crossbar arrays from a size of 8 × 8 up to 1,024 × 1,024 (Fig. 5c). We tracked how the intrinsic noise in these analogue arrays grows with size for both dense and sparse matrices. Higher-density matrices (≥50% non-zero values) lead to more noise than sparse matrices (≤10% non-zero values). In both cases, the analogue computations in smaller arrays give results very close to ideal results in such a way that there is no overlap of neighbouring integer distributions. In other words, the error distribution has a standard deviation that is sufficiently below 0.5 that any result can be rounded to the nearest integer and will match the exact integer (digital) result; see lower inset in Fig. 5c for the noise distribution of a 32 × 32 array. Novel error-correction codes developed specifically for crossbar computations[41] can further ensure perfect results with some redundancy overhead. On the other hand, larger arrays can develop errors of a magnitude that may no longer be tolerable or easily corrected; see upper inset to Fig. 5c for a 128 × 128 array.

It can be seen in Fig. 5c that digitally equivalent precise computations may be computed in analogue arrays of sizes up to 100 × 100 for dense graphs, and up to 256 × 256 for sparse graphs. Arrays larger than this will add a finite amount of noise to the analogue computations. We emphasize, however, that this does not preclude the use of the mem-HNN system for solving larger graph problems, but does require the sub-tiling of larger graph problems across multiple smaller arrays, very similar to what has been architected in machine learning accelerators with memristor analogue systems[27,42]. However, an important observation here is that when solving optimization graph problems that grow in size, these require increasing amounts of injected noise (or harnessing of intrinsic noise) during computations, as the size of the energy barriers also typically scales. This is captured in the Fig. 5c plot of the 'optimal noise' needed, where simulations across an ensemble of different dense graph problems of varying sizes were solved and the optimal amplitude of noise to achieve the best success probabilities was computed. This shows a favourable match between the need for intrinsic noise to accelerate and improve the solution of optimization problems and the capability of analogue arrays to supply such noise up to fairly large arrays. As shown, even arrays up to a size of 1,024 × 1,024 generate less analogue noise than that which is optimally required. In our proposed technique for controllably amplifying and suppressing intrinsic analogue noise, this enables extremely large arrays to be utilized. From an area and power perspective, such large arrays are desirable because they allow the expensive peripheral circuitry to be amortized across many more parallel computations[40] performed. This peripheral circuitry (particularly the digital/analogue conversions) dominates the system area and power consumption[43] in analogue deep learning accelerators that need to maintain fairly high bit precision across the feed-forward layers. Thus, the performance gains for analogue-based, recurrent, graph problem solvers may be expected to be even greater than for deep learning accelerators[27,42,44] owing to (1) the larger array sizes and reduced peripheral circuitry cost per computation performed, and (2) the reduced bit precision requirements in the analog–digital conversion circuits (even binary for many important problems) and benefits in having analogue noise.

Mem-HNN performance and scaling with problem size

Having validated our mem-HNN system with experimental measurements, we were able to explore and benchmark our system via simulations for problem sizes and timescales beyond what has been

Table 1 | Comparison of the mem-HNN and current state-of-the-art annealing accelerators

	mem-HNN (sequential) memristor	mem-HNN (parallel) memristor	Noisy mean-field annealing GPU	Parallel tempering single-core CPU	D-wave 2000Q superconducting qubits	Coherent Ising machine fibre optics
Clock frequency	500 MHz	500 MHz	1.582 GHz	2.6 GHz		1 GHz
Annealing time	600 ns	600 ns	12.3 μs ($N = 100$)	223.6 μs	1 ms ($N = 55$)	150 μs
Time to solution	6.6 μs	600 ns	10 μs	223.6 μs	10^4 s	600 μs
Power	10.9 mW	120.1 mW	< 250 W	20 W	25 kW	
Energy to solution	72 nJ	72 nJ	< 2.5 mJ	4 mJ	250 MJ	
Solutions per second per watt	1.38×10^7	1.38×10^7	> 400	250	4×10^{-9}	
Update mechanism	Hybrid	Hybrid	Synchronous	Asynchronous	Synchronous	Asynchronous
Connectivity	All-to-all	All-to-all	All-to-all	All-to-all	Sparse	All-to-all
Scaling of success probability for $N \times N$ problems	e^{-N}	e^{-N}	e^{-N}	e^{-N}	e^{-N^2}	e^{-N}
Cryogenic cooling	No	No	No	No	Yes	No

We compared with a GPU implementation of the noisy mean-field algorithm[12], with simulations using the previously suggested parallel tempering implementation on a CPU7 (see Supplementary Discussion section 10.3) and with experimental results for the D-wave annealer and the measurement-feedback coherent Ising machine discussed in ref. [7]. A hybrid update mechanism updates 10 nodes at a time in a given iteration. Values reported for the mem-HNN are for 50 cycles, estimated at a 16-nm technology node (see Supplementary Discussion section 7). Unavailable quantities are left blank.

experimentally feasible to date. Figure 6 explores the problem size scaling, speed and energy consumption in solving dense max-cut problems (connectivity of 50%) of increasing size. All studies leverage noise via a hysteretic threshold to enhance the solution quality and number of cycles (as shown in Fig. 2). Figure 6a shows that for increasing problem sizes (see graphs) it becomes more difficult to find the globally optimal solution, reducing the success probability, but that increasing the number of cycles can mitigate this effect. This observation is consistent with the experimental results and associated circuit simulations.

Figure 6b shows that the computation time of the mem-HNN has a 99% probability of reaching the optimal solution[11] for increasing sizes of dense graphs. As illustrated in Fig. 1, larger problem sizes require larger memristor crossbars and peripheral circuitry. Owing to the serial update of each node, the latency scales linearly with the problem size. However, batches of nodes can be updated in parallel with a small impact on the solution probability but a net reduction in the time to solution. Figure 6 uses batches of ten nodes at once to speed up computation parallelism, with future improvements possible with more optimal schemes[37]. Additionally, instead of running the same problem instance sequentially on the same crossbar, problems can be solved on multiple crossbars in parallel, with variable initial states, in order to reach the target 99% probability. In the latter case, the time to solution scales only with problem size, as shown in Fig. 6b (filled symbols).

Energy consumption for the mem-HNN is shown in Fig. 6c. This plot captures energy consumed in the peripheral multiplexors, decoders, comparators and the analogue crossbars themselves. Additionally, larger graph problems have a reduced probability of solution (Fig. 6a), thus requiring an increasing number of parallel mem-HNN units. Nonetheless, the efficiency of performing the needed multiply and add operations (equation (1)) of a Hopfield network in the analogue domain using memristor crossbars is shown here. Our system outperforms current digital systems (CPUs and GPUs) by at least 10,000×, with even larger gains compared to current quantum annealers (detailed below).

Performance comparison with other annealing accelerators

In Table 1, we compare the performance of our proposed mem-HNN with other technologies in solving dense max-cut problems with 50% connectivity and graph size $N = 60$. These values are based on experimentally grounded simulations for the mem-HNN, particularly extracted from Fig. 6, whereas fully experimental results for the other technologies are taken from the literature. We list some relevant metrics which affect the throughput of the system (connectivity, time required to reach solution), the energy efficiency (power and energy required to reach solution), and potential for scalability (connectivity, scalability of the success probability).

In Table 1, we track four different types of technologies: mem-HNN, a fibre-based coherent Ising machine with an field programmable gate array in the feedback loop[7], a state-of-the-art implementation of parallel tempering on a single Intel Core i7-3720QM CPU at 2.60 GHz (details in Supplementary Discussion section 10.3), D-wave's 2000Q quantum annealer (containing 2,048 qubits)[7] and an NVIDIA GeForce GTX 1080 Ti GPU running a noisy mean field annealing algorithm[12]. The mem-HNN system utilizes analogue computing with nano-electronics, the coherent Ising machine system is a hybrid electronic and optical accelerator, and the GPU and CPU platforms represent digital hardware running physics-inspired heuristics.

Previous comparisons between emerging annealing technologies have primarily focused on the run time required to reach a solution with high likelihood (time to solution). In these studies, the energy efficiency typically received less attention (as noted before[45]). Because energy efficiency is a critical metric in a post-Moore computing world, we compared this for the listed technologies based on estimated power consumption. We note that both the D-wave and coherent Ising machine system are proof-of-concept systems which are not at present optimized for energy efficiency, and most of their power consumption is due to cooling and elaborate control circuits (Supplementary Discussion section 10.2). In contrast, energy efficiencies for our mem-HNN are readily estimated on the basis of extensive studies of memristor crossbar arrays for matrix–vector multiplication applications[39,44,46]. Although we do not have access to the power consumption of the GPU during implementation of the noisy mean field algorithm, we use the available power specifications for this model as a realistic upper estimate. For the CPU, an effective power consumption of 20 W has been estimated[45,47]. We acknowledge that a full chip design, which is the next step of this work in progress, will lead to overheads that may degrade performance. However, we believe that the estimates provided here will be close to the performance of a fully designed chip, given our previous

demonstration of similar estimates in close agreement with a fully designed chip[27,42].

Overall, it is evident that electronics-based approaches currently outperform both the quantum annealing and the optical systems (see Supplementary Discussion section 10 for a more detailed discussion). Among electronic solutions, the mem-HNN is comparable to both the GPU and CPU in terms of time to solution, but substantially outperforms both of them in terms of energy to solution (by at least 10,000-fold). These performance gains can be attributed to (1) the highly efficient matrix–vector multiplication performed in the analogue domain, (2) intrinsic access to a tunable noise source within the memristor crossbar circuit, (3) high parallelism in the dot-product operations as well as multiple crossbar circuits running separate instances of the graph problem with different starting states to speed up solution convergence (an approach also harnessed across GPU cores), and (4) a substantial reduction of data fetching and communication through the use of in-memory computing[48,49]. The latter feature refers to the elimination of the well known von Neumann bottleneck, where computation speed and energy consumption is dominated by bringing data from local memory or storage.

Fundamentally, the D-wave system is the only system without all-to-all connectivity between its nodes, which leads to costly overheads in the required number of Ising spins to implement dense connectivity matrices. Consequently, a poor scaling as a function of problem size has been observed for dense problems[7]. We note that even for a relatively small $N = 60$ problem, D-wave's metrics are unfavourable compared to the other technologies. For these tasks, therefore, the benefits due to D-wave's potential use of quantum effects are negligible, although we acknowledge that quantum annealers may offer speed-ups in simulating quantum processes[45]. The mem-HNN has similarities in approach to the acceleration of machine learning operations in more modern neural networks using memristor crossbars[39,46,50,51]. An important difference is that the present system need not act as a fully precise digital replacement[27], which would come at a high performance cost. Instead, the mem-HNN shows even higher performance potential over fully digital systems (by at least 10,000-fold) by operating deeper in the analogue domain and harnessing desirable intrinsic noise. We further suggest that there is ample room for future improvements. The use of continuously tunable resistive cells allows our mem-HNN to solve graph problems requiring multiple bit precision in the connection matrices, without increasing the power consumption. Notably, over 6-bit resolution has been experimentally shown[39], while architectures using tiling for arbitrarily high bit precision have been previously described[27]. Future circuits and architectures that leverage other low-power analogue elements such as negative-differential resistance[52] could eliminate nearly all digital functions (for example, the nonlinear filter of Fig. 1a present in the current design. Un-clocked asynchronous designs have further potential for performance gains, but will require deeper co-design of the algorithm with the physical hardware implementation. Finally, we note that in order for the present design to support arbitrarily large problem sizes that are well beyond single crossbars, a careful hierarchical architecture analogous to that in memristor-based machine learning systems[5,27] is required. A future research direction is to assess how the introduction of hierarchy and tiling in such a combinatorial accelerator will affect the time and energy scaling.

Conclusions

We have shown that analogue-domain memristor crossbar computations can provide a very competitive approach to solving computationally intractable problems. We also introduced techniques to modulate and harness noise within analogue neural networks. The mem-HNN system is a hybrid analogue–digital computing platform that will benefit from future CMOS technology nodes owing to compatible semiconductor processing. Memristors have shown size scalability[53] to 2 nm and fast operations[54] well below 1 ns. Augmenting digital technology with increasingly energy-efficient analogue components further extends analogue scaling with the potential to continue even after the end of Moore's law, should future technological benefits not be found elsewhere.

Data availability

The data supporting plots within this paper and other findings of this study are available from the corresponding author upon reasonable request.

Received: 12 March 2019; Accepted: 1 June 2020;
Published online: 6 July 2020

References

1. Zidan, M. A., Strachan, J. P. & Lu, W. D. The future of electronics based on memristive systems. *Nat. Electron.* **1**, 22–29 (2018).
2. Williams, R. S. What's next? [The end of Moore's law]. *Comput. Sci. Eng.* **19**, 7–13 (2017).
3. Hennessy, J. L. & Patterson, D. A. A new golden age for computer architecture. *Commun. ACM* **62**, 48–60 (2018).
4. Johnson, M. W. et al. Quantum annealing with manufactured spins. *Nature* **473**, 194 (2011).
5. Bojnordi, M. N. & Ipek, E. Memristive Boltzmann machine: a hardware accelerator for combinatorial optimization and deep learning. In *2016 IEEE Int. Symp. High Performance Computer Architecture (HPCA)* 1–13 (IEEE, 2016).
6. Shin, J. H., Jeong, Y., Zidan, M. A., Wang, Q. & Lu, W. D. Hardware acceleration of simulated annealing of spin glass by RRAM crossbar array. In *IEEE Int. Electron. Devices Meet. (IEDM)* 63–66 (IEEE, 2018).
7. Hamerly, R. et al. Experimental investigation of performance differences between coherent Ising machines and a quantum annealer. *Sci. Adv.* **5**, eaau0823 (2019).
8. Roques-Carmes, C. et al. Heuristic recurrent algorithms for photonic Ising machines. *Nat. Commun.* **11**, 249 https://doi.org/10.1038/s41467-019-14096-z (2020).
9. Kielpinski, D. et al. Information processing with large-scale optical integrated circuits. In *IEEE Int. Conf. Rebooting Computing (ICRC'16)* https://doi.org/10.1109/ICRC.2016.7738704 (IEEE, 2016).
10. Tezak, N. et al. Integrated coherent Ising machines based on self-phase modulation in microring resonators. *IEEE J. Sel. Top. Quant. Electron.* **26**, 1–15 (2020).
11. Aramon, M. et al. Physics-inspired optimization for quadratic unconstrained problems using a digital annealer. *Front. Phys.* **7**, 48 (2019).
12. King, A. D., Bernoudy, W., King, J., Berkley, A. J. & Lanting, T. Emulating the coherent Ising machine with a mean-field algorithm. Preprint at https://arxiv.org/abs/1806.08422 (2018).
13. *A quadratic unconstrained binary optimization problem formulation for single-period index tracking with cardinality constraints* White Paper (QC Ware Corp., 2018); https://qcware.com/wp-content/uploads/2019/09/index-tracking-white-paper.pdf
14. Kochenberger, G. et al. The unconstrained binary quadratic programming problem: a survey. *J. Comb. Optim.* **28**, 58–81 (2014).
15. Booth, M., Reinhardt, S. P. & Roy, A. *Partitioning optimization problems for hybrid classical/quantum execution* (D-Wave, 2017); https://www.dwavesys.com/sites/default/files/partitioning_QUBOs_for_quantum_acceleration-2.pdf
16. Neukart, F. Traffic flow optimization using a quantum annealer. *Front. ICT* **4**, 1–6 (2017).
17. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl Acad. Sci.* **79**, 2554–2558 (1982).
18. Hopfield, J. J. Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Natl Acad. Sci.* **81**, 3088–3092 (1984).
19. Guo, X. et al. Modeling and experimental demonstration of a hopfield network analog-to-digital converter with hybrid CMOS/memristor circuits. *Front. Neurosci.* **9**, 488 (2015).
20. Hu, S. G. et al. Associative memory realized by a reconfigurable memristive Hopfield neural network. *Nat. Commun.* **6**, 1–5 (2015).
21. Yang, J., Wang, L., Wang, Y. & Guo, T. A novel memristive Hopfield neural network with application in associative memory. *Neurocomputing* **227**, 142–148 (2017).
22. Duan, S., Dong, Z., Hu, X., Wang, L. & Li, H. Small-world Hopfield neural networks with weight salience priority and memristor synapses for digit recognition. *Neural Comput. Appl.* **27**, 837–844 (2016).
23. Lucas, A. Ising formulations of many NP problems. *Front. Phys.* **2**, 5 (2014).

24. Coffrin, C., Nagarajan, H. & Bent, R. *Ising Processing Units: Potential and Challenges for Discrete Optimization* (LANL, 2017); https://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-17-23494
25. Hopfield, J. J. & Tank, D. W. 'Neural' computation of decisions in optimization problems. *Biol. Cybernet.* **52**, 141–152 (1985).
26. Boahen, K. A neuromorph's prospectus. *Comput. Sci. Eng.* **19**, 14–28 (2017).
27. Shafiee, A., Nag, A., Muralimanohar, N. & Balasubramonian, R. ISAAC: a convolutional neural network accelerator with in-situ analog arithmetic in crossbars. *ACM SIGARCH Comput. Archit. News* **44**, 14–26 (2016).
28. Le Gallo, M. et al. Mixed-precision in-memory computing. *Nat. Electron.* **1**, 246–253 (2018).
29. Ielmini, D., Nardi, F. & Cagli, C. Resistance-dependent amplitude of random telegraph-signal noise in resistive switching memories. *Appl. Phys. Lett.* **96**, 053503 (2010).
30. Mahmoodi, M. R., Nili, H. & Strukov, D. B. RX-PUF: low power, dense, reliable, and resilient physically unclonable functions based on analog passive rram crossbar arrays. In *2018 IEEE Symp. VLSI Technology* 99–100 (IEEE, 2018).
31. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P. Optimization by simulated annealing. *Science* **220**, 671–680 (1983).
32. Chen, L. & Aihara, K. Chaotic simulated annealing by a neural network model with transient chaos. *Neural Netw.* **8**, 915–930 (1995).
33. He, Y. Chaotic simulated annealing with decaying chaotic noise. *IEEE Trans. Neural Netw.* **13**, 1526–1531 (2002).
34. Katti, R. S. & Srinivasan, S. K. Efficient hardware implementation of a new pseudo-random bit sequence generator. In *2009 IEEE Int. Symp. Circuits and Systems* 1393–1396 (IEEE, 2009).
35. Wiegele, A. *Biq Mac Library—A Collection of Max-Cut and Quadratic 0-1 Programming Instances of Medium Size* (Univ. of Klagenfurt, 2007); http://biqmac.aau.at/biqmaclib.pdf
36. Liu, W., Wang, L. Solving the shortest path routing problem using noisy Hopfield neural networks. In *2009 WRI Int. Conf. Communications and Mobile Computing* 299–302, https://doi.org/10.1109/CMC.2009.366 (IEEE, 2009).
37. Matsubara, S. et al. Ising-model optimizer with parallel-trial bit-sieve engine. In *Conf. Complex, Intelligent, and Software Intensive Systems* 432–438 (Springer, 2017).
38. Sheng, X. et al. Low-conductance and multilevel CMOS-integrated nanoscale oxide memristors. *Adv. Electron. Mater.* **5**, 1800876 (2019).
39. Hu, M. et al. Memristor-based analog computation and neural network classification with a dot product engine. *Adv. Mater.* **30**, 1705914 (2018).
40. Hu, M. et al. Dot-product engine for neuromorphic computing: programming 1T1M crossbar to accelerate matrix-vector multiplication. In *Proc. 53rd Annual Design Automation Conf.* 19 (ACM, 2016).
41. Roth, R.M. Fault-tolerant dot-product engines. *IEEE Trans. Inform. Theory* **65**, 2046–2057 (2018).
42. Ankit, A. et al. Puma: a programmable ultra-efficient memristor-based accelerator for machine learning inference. In *Proc. 24th Int. Conf. Architectural Support for Programming Languages and Operating Systems* 715–731 (ACM, 2019).
43. Rekhi, A.S. et al. Analog/mixed-signal hardware error modeling for deep learning inference. In *Proc. 56th Ann. Design Automation Conf.* 299–302, https://doi.org/10.1145/3316781.3317770 (ACM, 2019).
44. Marinella, M. J. et al. Multiscale co-design analysis of energy, latency, area, and accuracy of a ReRAM analog neural training accelerator. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **8**, 86–101 (2018).
45. Mandra, S. & Katzgraber, H. G. A deceptive step towards quantum speedup detection. *Quant. Sci. Technol.* **3**, 1–11 (2018).
46. Ambrogio, S. et al. Equivalent-accuracy accelerated neural-network training using analogue memory. *Nature* **558**, 60 (2018).
47. Villalonga, B. et al. A flexible high-performance simulator for verifying and benchmarking quantum circuits implemented on real hardware. *npj Quant. Inform.* **5**, 1–16 (2019).
48. Linn, E., Rosezin, R., Tappertzhofen, S., Böttger, U. & Waser, R. Beyond von Neumann—logic operations in passive crossbar arrays alongside memory operations. *Nanotechnology* **23**, 305205 (2012).
49. Ielmini, D. & Wong, H.-S. P. In-memory computing with resistive switching devices. *Nat. Electron.* **1**, 333–343 (2018).
50. Prezioso, M. et al. Training and operation of an integrated neuromorphic network based on metal-oxide memristors. *Nature* **521**, 61–64 (2015).
51. Burr, G. W. et al. Experimental demonstration and tolerancing of a large-scale neural network (165000 synapses) using phase-change memory as the synaptic weight element. *IEEE Trans. Electron. Devices* **62**, 3498–3507 (2015).
52. Pickett, M. D., Medeiros-Ribeiro, G. & Williams, R. S. A scalable neuristor built with Mott memristors. *Nat. Mater.* **12**, 114–117 (2013).
53. Pi, S. et al. Memristor crossbar arrays with 6-nm half-pitch and 2-nm critical dimension. *Nat. Nanotechnol.* **14**, 35–39 (2019).
54. Torrezan, A. C., Strachan, J. P., Medeiros-Ribeiro, G. & Williams, R. S. Sub-nanosecond switching of a tantalum oxide memristor. *Nanotechnology* **22**, 485203 (2011).

Acknowledgements

We are grateful to S. Mandrà for performing the CPU simulations used in Table 1 and early review of the article. We acknowledge discussions with H. Katzgraber, P. L. McMahon, E. Rothberg, K. Roenigk, C. Santos, R. Slusher and J. Weinschenk. This research was based upon work supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract number 2017-17013000002.

Author contributions

F.C., S.K., T.V.V., C.L. and J.P.S. performed the simulations. T.V.V., S.K., R.L., Z.L., M.F. and J.P.S. contributed to performance benchmarking. S.K. and J.P.S. performed the experiments. X.S., C.L., Q.X., J.J.Y. and J.P.S. contributed to the chip fabrication and experimental system development. All authors supported analysis of the results and commented on the article.

Competing interests

The authors declare that they have no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41928-020-0436-6.

Correspondence and requests for materials should be addressed to J.P.S.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.