scientific reports

Check for updates

OPEN

Amoeba-inspired analog electronic computing system integrating resistance crossbar for solving the travelling salesman problem

Kenta Saito[1]✉, Masashi Aono[2,3,4] & Seiya Kasai[1,5]✉

Combinatorial optimization to search for the best solution across a vast number of legal candidates requires the development of a domain-specific computing architecture that can exploit the computational power of physical processes, as conventional general-purpose computers are not powerful enough. Recently, Ising machines that execute quantum annealing or related mechanisms for rapid search have attracted attention. These machines, however, are hard to map application problems into their architecture, and often converge even at an illegal candidate. Here, we demonstrate an analogue electronic computing system for solving the travelling salesman problem, which mimics efficient foraging behaviour of an amoeboid organism by the spontaneous dynamics of an electric current in its core and enables a high problem-mapping flexibility and resilience using a resistance crossbar circuit. The system has high application potential, as it can determine a high-quality legal solution in a time that grows proportionally to the problem size without suffering from the weaknesses of Ising machines.

Combinatorial optimization problems are computationally demanding tasks appearing in various practical applications, such as optimization of traffic flow, path planning, nurse scheduling and advertisement allocation[1–4]. Often these problems become intractable for conventional von Neumann-type computers, such as general-purpose CPUs; they need to evaluate an enormous number of candidate solutions in a serial manner, where the number of the candidates grows exponentially with the problem size, leading to "combinatorial explosion".

The travelling salesman problem (TSP) is one of the most widely investigated combinatorial optimization problems; given a map of N cities, the TSP is stated as a problem of finding the shortest route for visiting each city exactly once and returning to the starting city[5,6], where the number of all legal solutions (possible routes) grows factorially as $(N-1)!/2$. The TSP is a nondeterministic polynomial time (NP)-hard problem, that is, any exact algorithm to find the optimal solution (exactly the shortest route) for a general instance in polynomial time is not known so far. On the other hand, various nature-inspired approximation algorithms have been proposed to promptly derive a high-quality legal solution (a satisfactory short route), such as the k-opt algorithm with simulated annealing, genetic algorithm, particle swarm optimization algorithm and ant colony optimization algorithm[7–11].

Many of the nature-inspired algorithms are formulated to update multiple variables in parallel to achieve rapid search, whereas the serial process of the CPU that manipulates a single bit at a time can only simulate the parallelism in a limited way. Therefore, it is required to develop a novel domain-specific computing architecture to implement these algorithms to maximize their parallel search capabilities by exploiting physical processes of specific hardware, expecting to cultivate new potentials and markets of combinatorial optimization. The first physical computing system for solving the TSP comprises the Hopfield's recurrent neural network implemented by an electronic circuit[12,13]. This system, however, was not so useful because it frequently converges at a local minimum state (a low-quality solution) and sometimes cannot reach even a candidate solution for some problem instances[14–16]. In fact, for some instances of 10-city TSP, it was reported that the rate of finding a legal solution was at most 20%[15].

[1]Research Center for Integrated Quantum Electronics and Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan. [2]Amoeba Energy Co., Ltd., Fujisawa, Japan. [3]Graduate School of Media and Governance, Keio University, Fujisawa, Japan. [4]Graduate School of Science and Technology, Keio University, Yokohama, Japan. [5]Center for Human Nature, Artificial Intelligence, and Neuroscience, Hokkaido University, Sapporo, Japan. ✉email: k-saito@rciqe.hokudai.ac.jp; kasai@rciqe.hokudai.ac.jp

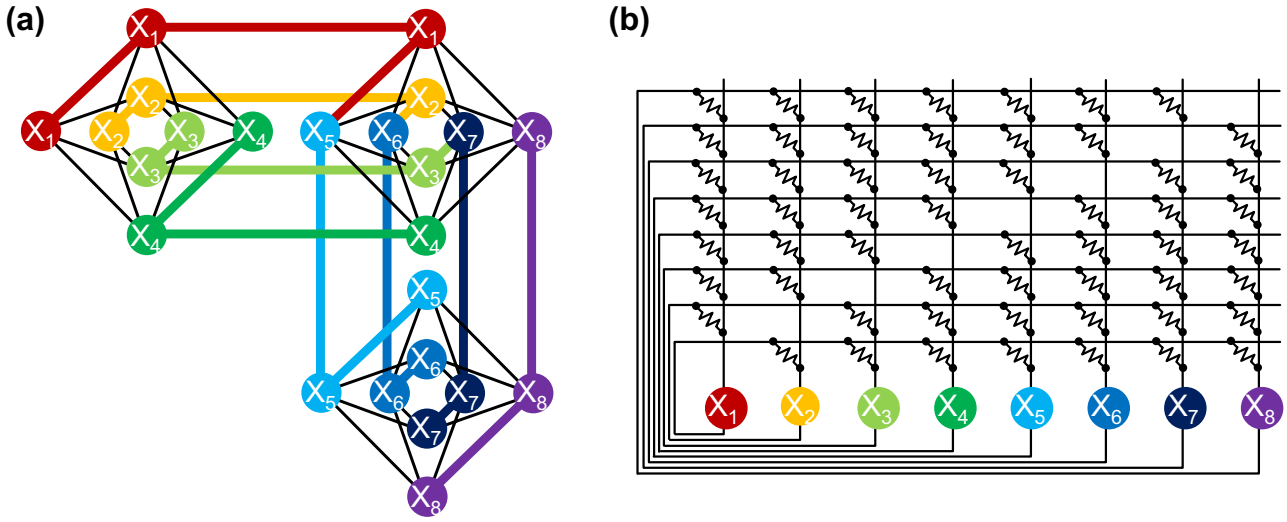

Figure 1. Physical mapping schematics of the TSP to (**a**) the Ising machine and (**b**) electronic amoeba. Each state variable X_i, taking a value of 1, determines where and when should a salesman visit. (**a**) Chimera graph for fully connecting an arbitrary pair of spin variables X_i and X_j, requiring redundant variables for each X_i. (**b**) Resistance crossbar of the instance-mapping circuit (IMC), achieving full connectivity without introducing any redundant variables.

In recent years, many proposals on rapid combinatorial optimization by physically implementing the "Ising machine" or "annealing machine" with several exploration mechanisms, such as quantum annealing (QA) and related methods, have attracted a lot of public attention[17–21]. Each of these machines explores an optimal solution by mapping the problem to a process of finding a minimum-energy spin assignment in the Ising model that abstracts a ferromagnetic material[17,22,23]. However, the problem mapping and parameter tuning of the Ising model are complex and costly. For the N-city TSP, the regular layout of spin variables with sparse connectivity requires redundant variables to be introduced in the order of N^4 to handle irregularly distributed cities, leading to a rapid increase in the circuit area[24–28]. Figure 1a shows a graph structure of the Ising model, which is referred to as a chimera graph[17]. In such a structure, the consistency between the redundant variables can be broken potentially. When the parameter tuning cannot be made appropriately, the Ising model sometimes converges at an illegal state in which constraints of the TSP to exclude the revisiting of a once-visited city and to exclude simultaneous visits to multiple cities are violated (see Supplementary Information [SI]). Actually, in the physical Ising machines solving the Max-cut problem, the probability of reaching a legal state varies depending on the connectivity among graph nodes, and the sparsely connected graph often results in even worse performance owing to the variable overhead[29].

Our approach is not based on the Ising model. We focus on a living amoeboid organism that performs trial-and-error behaviour to survive efficiently and resiliently in a harsh environment, deforming its gel-like body[30,31]. Here, we demonstrate, as a proof of concept, an analogue electronic computing system called an "electronic amoeba"[32,33], inspired by the food search and risk avoidance behaviour of a single-celled amoeboid organism, *Physarum polycephalum*[30,31,34–39]. In the electronic amoeba, an arbitrary TSP instance can be mapped on the resistor network of a crossbar structure shown in Fig. 1b, which we call the "instance-mapping circuit (IMC)". The architecture of the IMC is similar to that of the Hopfield's recurrent neural network[12,13]. However, as shown in Fig. 2a, it is connected with the "amoeba core", which contributes to avoiding the convergence at an illegal state. In the author's previous works[35,38,39], we formulated an algorithm representing a primitive idea of the electronic amoeba and predicted its potential performance without any physical implementation. In this paper, with results obtained from numerical simulations using a conventional computer and laboratory experiments using a physically fabricated circuit (Fig. 2b), for the first time, we show that the electronic amoeba finds a high-quality TSP solution in a time that is proportional to N. The electronic amoeba is highly scalable and energy-efficient as it comprises existing complementary metal-oxide semiconductor (CMOS) devices and is expected to be useful for wide range of applications.

Results

Figure 2a shows a schematic of the electronic amoeba composed of the amoeba core and IMC, which electronically mimics the solution-searching dynamics of a so-called "amoeba-based computer" that employs a living amoeba to search for a solution to the TSP[38,39] (see Supplementary Information). The state of each unit in the amoeba core represents the decision on where and when to visit. The IMC implements a type of feedback control, called the "bounceback control", which refers to the TSP constraints and intercity distances from the given map and sends a "bounceback signal" to each unit in the amoeba core[36–39]; the signal is determined in accordance with the recurrent neural network dynamics defined by Eq. (1) in "Methods".

We first conduct numerical simulations using a circuit simulator run on a conventional computer (see "Methods") to determine if the electronic amoeba can solve the 4-city TSP instance shown in Fig. 3a. Figure 3b

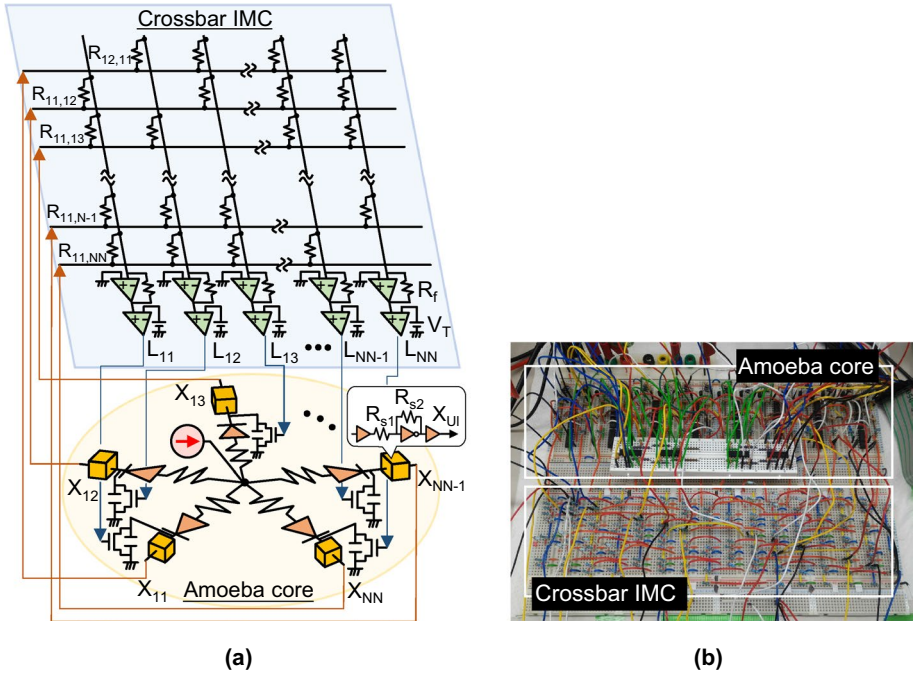

Figure 2. Electronic amoeba for solving the TSP. (**a**) Schematic of the system consisting of the amoeba core and IMC. For solving the N-city TSP, the amoeba core uses N^2 units to represent state variables, each of which performs the charging and discharging dynamics of the capacitor to express expanding and shrinking behaviour of a pseudopod-like branch of the amoeba. The IMC executes product-sum operations of recurrent neural network dynamics for the bounceback control to govern the interactions among the variables (see "Methods"). (**b**) Photo of fabricated electronic amoeba consisting of commercially available CMOS devices.

demonstrates an example of output waveforms obtained from the circuit simulation. The subscripts of the state of each unit $X_{V,k}$, V and k, mean that city V is visited at the kth order. Initially, every unit takes a state of 1 because the capacitor charge is set to zero. The IMC then sends the bounceback signals to all units to flip their states from 1 to 0, as the all-one states are violating constraints of the TSP. The state of each unit gradually approaches to 0 when charging the capacitor by injecting a current from the current source. After several flips have been induced by the bounceback control, the dynamics of all units became stable as shown in the hatching area in Fig. 3b. At this moment, the electronic amoeba finds an optimal solution, $D \rightarrow A \rightarrow B \rightarrow C \rightarrow D$, which corresponds to the shortest route.

Using a circuit simulator, we have investigated the solution-searching performance of the electronic amoeba for N ranging from 10 to 30. We performed 50 trials for each instance for N not greater than 20 but only once for each N more than 20 because the simulation time increased rapidly; it took 5 h for the 20-city instances but took 6 days for the 30-city cases. In each trial, resistances in the units were randomly assigned from 1 Ω to 10 kΩ to lead the electronic amoeba to explore a wider state space. To evaluate the rate of finding a legal solution in the electronic amoeba, we performed 560 trials for solving the TSP of 10–30 cities. The rate was found to be 100%; the system certainly converged to one of the legal solutions for every try. This is because the amoeba core always stabilizes at a steady state in which no variable violates the TSP constraints[39]; in such a state, no further change in all units in the amoeba core is induced by the bounceback signals. Figure 4a shows the length of the route obtained by the circuit simulator. In Fig. 4a, the vertical axis is normalized by the average route length obtained from random sampling of 10,000 trials; if a value on the vertical axis is less than 1.0, it implies that the quality of the legal solution found is higher than that found by random sampling. The results indicate that the electronic amoeba finds higher-quality solutions than random sampling. Moreover, the average of the route length was on a declining trend; the quality did not degrade even when the problem size N became larger. By introducing random variations in the resistances of the units, each unit varies the velocity of the transition from state 1 to 0, and a wide variety of legal solutions were found as shown in Fig. 4b; it reached different solutions even for an identical instance, although it did not guarantee to reach the optimal one.

Figure 4c shows that the average time required for the electronic amoeba to find a legal solution increases almost only linearly as a function of N. This result reproduces the observation confirmed in the amoeba-based computer well; the living amoeba reached an approximate solution of the TSP in linear time[38,39]. A hypothetical mechanism of the linear-time solution has been proposed with an abstract mathematical model of the solution-searching dynamics of the amoeba-based computer; the numerical simulation of the model, named "AmoebaTSP", suggested that the linear-time solution can be achieved if the hub of the single-celled organism could supply the intracellular resources to grow its branches with a constant rate, even while responding to the bounceback signals[38,39]. The linear-time operation is attributed to the design of the bounceback control together with parallel operations of all units in the amoeba core. The units try to choose a path between two cities having

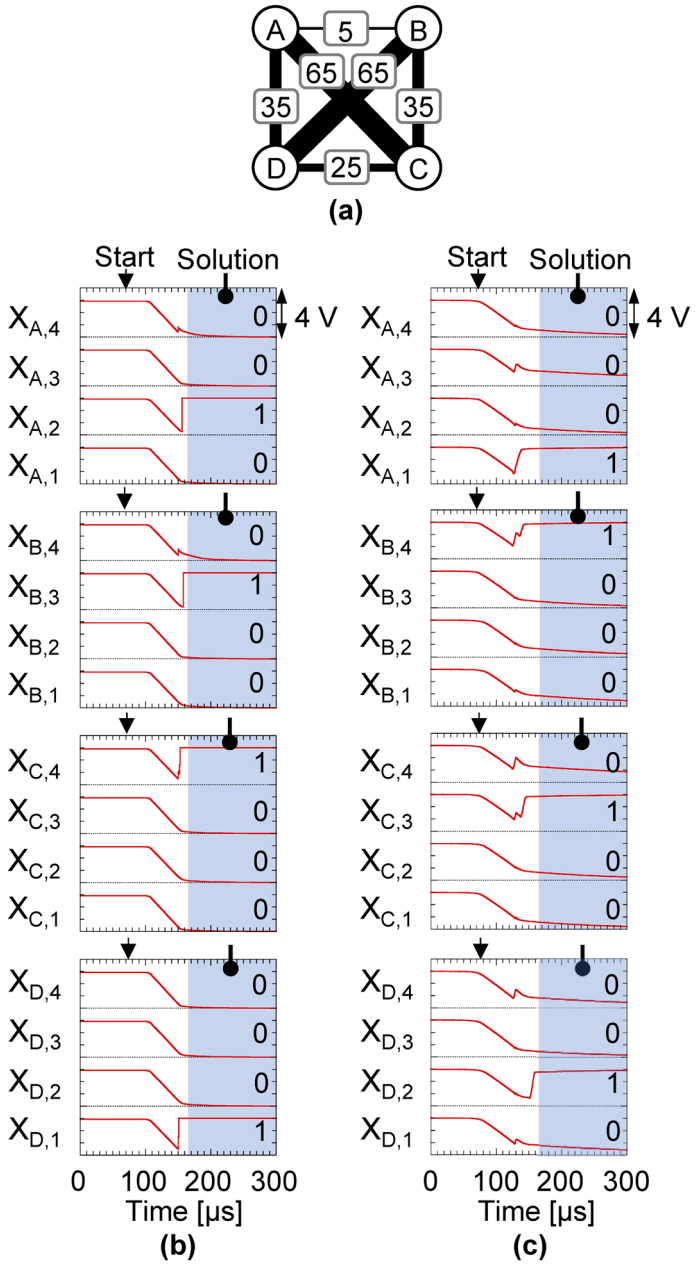

Figure 3. Solution-searching dynamics of electronic amoeba. (**a**) 4-city TSP instance. Each edge weight indicates the distance between corresponding cities. (**b**,**c**) The dynamics of state variables in the circuit simulation and fabricated system, respectively. Variable $X_{V,k}$ represents that city V is visited at the kth order. The shortest routes, $D \rightarrow A \rightarrow B \rightarrow C \rightarrow D$ and $A \rightarrow D \rightarrow C \rightarrow B \rightarrow A$, were found by the simulation and fabricated system, respectively.

a shorter distance, according to information on shortness of the distances of possible paths through accumulating and comparing experiences of being inhibited by the bounceback signals; the two units representing a shorter-distance path are less frequently inhibited compared to the others and are allowed to take relatively larger state values. The bounceback rule is designed so that once a path and their visiting orders are decided, the rule restricts the amoeba core from changing the decision after that. Thereby the system decides each path one by one, avoiding illegal paths.

The electronic amoeba will reach a high-quality solution in linear time even for larger-size instances if it follows a similar mechanism as AmoebaTSP. Moreover, the solution search time of the electronic amoeba can be reduced further as the flipping of the state of each unit can be accelerated by increasing the current and/or decreasing the capacitance (see Supplementary Information).

We compared the solution-searching performance of the circuit simulator-based electronic amoeba with that of a representative stochastic local search algorithm, the 2-opt (see "Methods"), which is a simple and fast

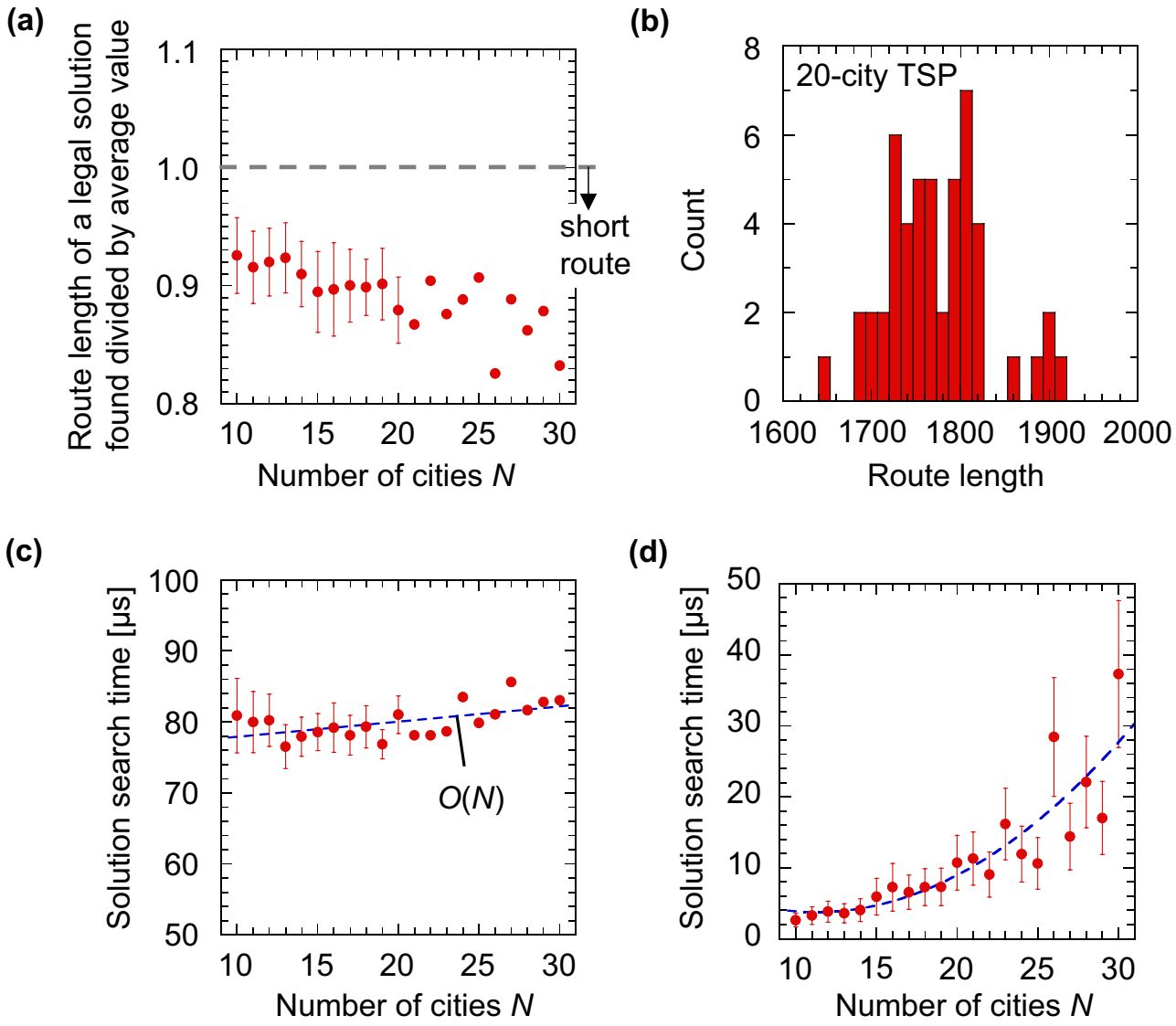

Figure 4. Solution-searching performance of circuit-simulator-based electronic amoeba and 2-opt evaluated as a function of the problem size. (**a**–**c**) The results obtained from the electronic amoeba and (**d**) from 2-opt. (**a**) Route length normalized by the average length calculated by random sampling. Error bars for the 10–20-city instances denote standard deviations obtained from 50 trials, whereas those for 21–30-city cases are not shown as only a trial was performed for each case. (**b**) Histogram of route lengths of the solutions found for a 20-city instance after 50 trials. The average route length calculated by random sampling was 2016, whereas that obtained by the electronic amoeba was 1780. (**c**) Solution search time estimated from the dynamics generated by the circuit simulator (see "Methods"). (**d**) Solution search time required for the calculation using a conventional computer (see "Methods").

method requiring no parameter optimization[7]. The quality of the solution obtained by the 2-opt gets higher (and saturates eventually) as its main operation iterates for a larger number of times, but we terminated the iteration when the quality become equal to that obtained by the electronic amoeba and measured the time required by the termination. As shown in Fig. 4d, the 2-opt required an amount of time that grows as a quadratic function of the problem size, whereas the electronic amoeba needed only linear time to present the solution with the same level of quality (Fig. 4c). Accordingly, the electronic amoeba, once physically implemented, will be more useful for finding a high-quality solution in a shorter search time than the 2-opt when run on a conventional computer and when the number of cities exceeds 50.

By fabricating the electronic amoeba physically (Fig. 2b) using CMOS devices, we have verified that the fabricated system can solve various 4-city TSP instances as shown in Fig. 5a–e, where the optimal and worst solutions are summarized in Table 1. The amoeba core comprises 16 branches, and we can map an arbitrary 4-city instance by changing the resistance values in the IMC. The time evolution of the state variables in the fabricated system is shown in Fig. 3c. At an initial stage, the variables behaved similarly to those in the circuit simulation (Fig. 3b), and they became stable after reaching a solution. The bottom of Fig. 5a–e shows that the

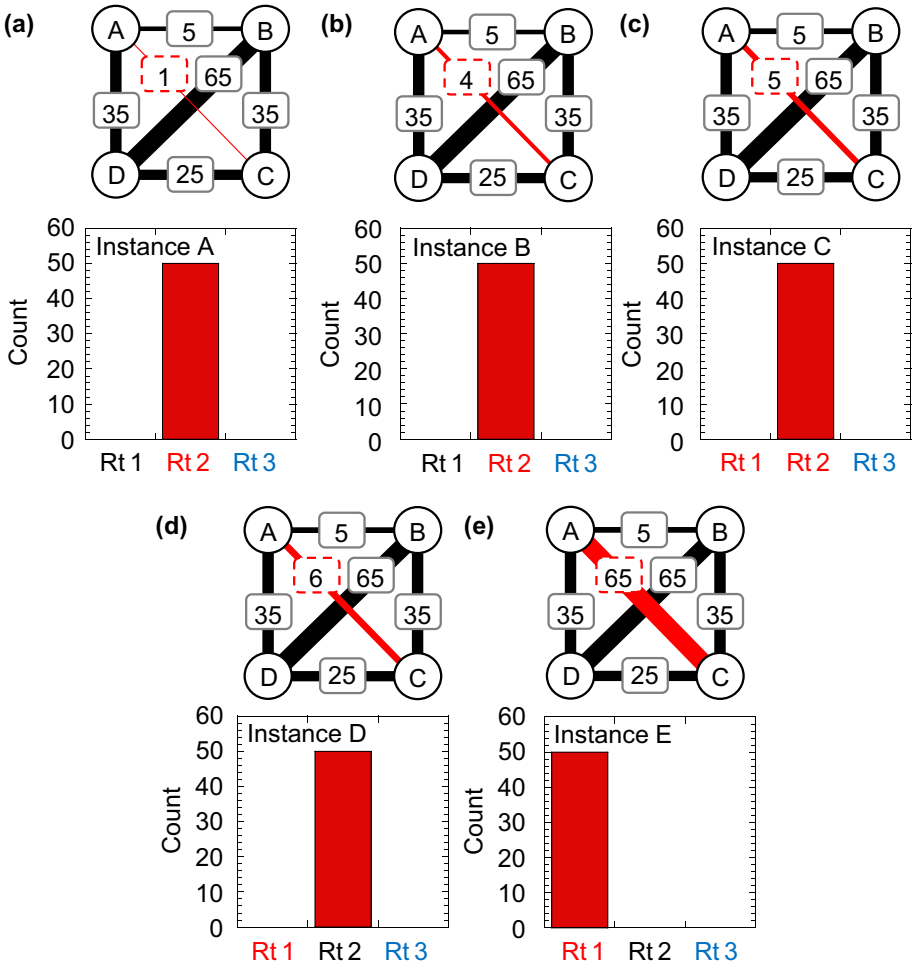

Figure 5. Results obtained from the fabricated electronic amoeba for 4-city TSP instances that gave three possible routes (legal solutions). (**a**–**e**) Histograms of the routes obtained from 50 trials. Abbreviations in horizontal axes are Rt 1: $A \rightarrow B \rightarrow C \rightarrow D \rightarrow A$, Rt 2: $A \rightarrow C \rightarrow D \rightarrow B \rightarrow A$ and Rt 3: $A \rightarrow D \rightarrow B \rightarrow C \rightarrow A$.

system found the shortest route for instances A–C and E where we performed 50 trials for each instance without changing the resistance values.

The fabricated system could not find the shortest route for instance D as shown in Fig. 5d, although the shortest route length of D equals that of C. This would be attributed to a number of device variations in the fabricated circuit, such as the threshold voltage variation in the CMOS inverter, offset voltage variation in the operational amplifier and difference in the wiring length in the IMC, which might create a preference when making a decision. However, for the cases where the route lengths are widely distributed such as instance E, the system reached the optimal solution, overcoming the preference. When solving larger-sized instances and introducing more noise, the system performance is expected to become reliable and robust (Supplementary Fig. S6), because the differences in qualities (route lengths) of legal solutions of those instances become relatively smaller than that of smaller-sized instances, depending on noise amplitude.

Note that it is necessary for the Ising machines to expend considerable efforts on problem-mapping and parameter-tuning processes prior to solving the problem. If this pre-processing could be made properly, the Ising

	Instance A	Instance B	Instance C	Instance D	Instance E
Rt 1: A→B→C→D→A	100	100	100	100	100
Rt 2: A→C→D→B→A	96	99	100	101	160
Rt 3: A→D→B→C→A	136	139	140	141	200

Table 1. Summary of TSP instances and their route lengths.

model combined with simulated annealing exhibits a better performance in finding a higher-quality TSP solution than the electronic amoeba, otherwise it fails to reach even a legal solution (see Supplementary Information). In contrast, there is no need for the electronic amoeba to execute complex and costly pre-processing because the IMC offers a high problem-mapping flexibility with its unrestricted connectivity between an arbitrary pair of variables and requires only simple numerical operations to determine the parameters (see "Methods"). Moreover, when the resistors in the IMC are replaced with memristors or atomic switches, enabling dynamic resistance rewriting[40–44], the TSP instance to be solved can be updated promptly even in the middle of the solution-searching process, and the bounceback control will enable the amoeba core to find a new solution of the updated instance effortlessly by slightly revising the previous one. Such a dynamically rewritable IMC will enable the electronic amoeba to respond resiliently to sudden changes in the problem constraints caused by unexpected failures occurring in ever-changing practical situations.

The scalability is an important issue for the implementation of the electronic amoeba. For solving N-city TSP, the amoeba core needs N^2 units, and the crossbar IMC requires $2N^2$ wires and N^4 resistors at cross points of the wires. The circuit area of the electronic amoeba, therefore, grows in the order of N^2, which is less costly than the Ising machines that require the area in the order of N^4. This order is apparently comparable to that of the Ising model-based digital machines with full connectivity in logical level. However, the physical crossbar in the electronic amoeba produces advantages in terms of execution time and energy consumption; the crossbar allows to execute the bounceback control for all variables in a fully parallel manner, whereas the FPGA-based digital Ising machine requires a lot of memory accesses to achieve the logical full-connectivity, consuming higher time and energy costs. Owing to the modern digital LSI technology, each capacitance in the amoeba core will be downsized to the minimum level where its charging time is distinguishable from that of the parasitic elements. It is expected that state-of-the-art nanotechnology to fabricate a nanoscale crossbar structure equipped with memristors to represent analogue resistance values[42,43] will suppress the increase in the physical size of the IMC. The electronic amoeba can be built and maintained with a significantly cheaper cost than the Quantum annealing machine that requires a lot of elaborate equipment for refrigeration and maintenance of quantum coherence. Furthermore, while the computing speed of other Ising machines are unscalable as they suffer from the "Neumann bottleneck" that limits the data transfer rate in a data bus between memory and operation sections, the electronic amoeba does not encounter such a limitation.

In this paper, we demonstrated that, given an arbitrary TSP instance, the electronic amoeba enables to start the computing readily after simple resistance determination operations in the IMC and to find a high-quality solution certainly in only linear time, exploiting the spontaneous dynamics of the electric current in the amoeba core. This reliable and swift solution-searching capability could be beneficial for particular applications that prioritize the search time over the quality of a solution found. For example, in a situation at a disaster site where presenting reliable evacuation routes for residents is necessary, making a swift announcement should be prioritized than deriving the exactly optimal routes. The electronic amoeba would be more useful than using conventional computers to run the stochastic local search algorithm when the number of cities exceeds a hundred or more. Moreover, the compactness of the IMC suggests that the system-on-chip approach supported by semiconductor LSI technologies will further enhance the scalability and energy-efficiency, making it useful for wider cloud- and edge-computing applications. One of our future subjects is to improve the quality of the solution found by the electronic amoeba. Possible approaches are to assign appropriate initial states to the amoeba core units using the genetic algorithm, to impose stochastic fluctuations using a hardware random number generator to forcibly escape local minimum solutions (see SI), and to introduce delays in the bounceback control to induce the oscillation of state variables[45].

Methods

Bounceback control of the electronic amoeba. To implement the bounceback control for solving the TSP on the crossbar IMC, we followed the basic scheme of the amoeba-based computing system (see SI)[36–38]. For the N-city TSP, $N \times N$ state variables are needed where each variable X_{Vk} with subscript Vk indicates that city V is visited at the kth order: when $X_{Vk} = 1$, the city V is visited at the kth order. The bounceback signal in the electronic amoeba is generated by the crossbar IMC shown in Fig. 2a. The IMC evaluates all bounceback signals in a parallel and continuous fashion. The IMC circuit computes the bounceback signal is as follows,

$$L_{Vk} = F\left(\sum_{Ul} \frac{R_f}{R_{Vk,Ul}} \cdot X_{Ul} - V_T\right), \tag{1}$$

where R_f is a feedback resistor for the op-amp and $R_{Vk,Ul}$ is a resistor at cross point of the output line Vk and the input line Ul in the crossbar corresponding to the intercity distance. V_T is a threshold value of the threshold function F in the output portion of the crossbar IMC, which is implemented by a comparator. In this study we used $V_T = 1.5$ V and $R_f = 10$ kΩ. $R_f/R_{Vk,Ul}$ is given by

$$\frac{R_f}{R_{Vk,Ul}} = \begin{cases} 0.5 & (\text{if } V = U \text{at } k \neq l \text{or } V \neq U \text{at } k = l) \\ \text{dist}(V,U)/\lambda & (\text{if } V \neq U \text{and } |k-l| = 1) \\ 0 & (\text{otherwise}) \end{cases} \tag{2}$$

where $\text{dist}(V, U)$ is an intercity distance between cities V and U and λ is a normalization factor. $R_f/R_{Vk,Ul}$ defines the interaction between the state variables so as to (1) prohibit the system from visiting the city once visited, (2) prohibit the system from visiting different cities at once, and (3) minimize the total length of a travel route. λ is a normalized coefficient and we use $\lambda = \max(\text{dist}(V', V'') + \text{dist}(V'', V'''))/\theta$, where θ is a threshold in the sigmoid function and $\text{dist}(V', V'') + \text{dist}(V'', V''')$ is the maximum value of the total distance between the cities V' and V''' via V''.

A CMOS inverter with resistors shown as square boxes in Fig. 2a represents a sigmoid function. Here, the slope of the sigmoid function is defined by the resistance values $R_{s1} = 390\,\mathrm{k}\Omega$ and $R_{s2} = 2.2\,\mathrm{M}\Omega$. The measured behaviour of a pseudopod-like branch in the fabricated amoeba core is shown in Supplementary Information.

The bounceback control ensures the system does not stabilize whenever there are constraints that remain unsatisfied as the bounceback signals flip the constraint-violating variables from 1 to 0. The IMC does not have any free parameters, although the degree of resistance variation in the amoeba core is adjustable. The electronic amoeba, therefore, does not require any complex and costly pre-processing for problem mapping and parameter tuning.

Circuit simulator. We used the electronic circuit simulator, LTspice XVII (Simulation Program for Integrated Circuit Emphasis simulator, SPICE, https://www.analog.com). The current source is $5N^2\,\mu\mathrm{A}$ for the number of cities N, capacitance C is 500 pF and initial voltage of capacitors is set to 1.5 V in the circuit simulation. In the fabricated electronic amoeba integrating the commercial discrete devices, C is the sum of discrete components near 470 pF and the current source is set to 80 μA.

2-opt. The stochastic local search algorithm, 2-opt, updates the route by iterating the following main operation starting from a randomly chosen route (a legal solution); it chooses two cities randomly and inverts a path between the two locally in the current route if the inversion resulted in a reduction in route length. We performed the 2-opt calculation using a conventional computer (Intel Xeon processor E5-1650 v2 @ 3.50 GHz).

Received: 26 August 2020; Accepted: 13 November 2020
Published online: 27 November 2020

References

1. Neukart, F. *et al.* Traffic flow optimization using a quantum annealer. *Front. ICT* **4**, 29 (2017).
2. Ohzeki, M., Miki, A., Miyama, M. J. & Terabe, M. Control of automated guided vehicles without collision by quantum annealer and digital devices. *Front. Comput. Sci.* **1**, 9 (2019).
3. Ikeda, K., Nakamura, Y. & Humble, T. S. Application of quantum annealing to nurse scheduling problem. *Sci. Rep.* **9**, 12837 (2019).
4. Nishimura, N., Tanahashi, K., Suganuma, K., Miyama, M. J. & Ohzeki, M. Item listing optimization for E-commerce websites based on diversity. *Front. Comput. Sci.* **1**, 2 (2019).
5. Kruskal, J. B. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Am. Math. Soc.* **7**, 48–50 (1956).
6. Bektas, T. The multiple traveling salesman problem: An overview of formulations and solution procedures. *Omega* **34**, 209–219 (2006).
7. Lin, S. & Kernighan, B. W. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.* **21**, 498–516 (1973).
8. Kirkpatrick, S. Optimization by simulated annealing: Quantitative studies. *J. Stat. Phys.* **34**, 975–986 (1984).
9. Whitley, D. A genetic algorithm tutorial. *Stat. Comput.* **4**, 65–85 (1994).
10. Wang, K., Huang, L., Zhou, C. & Pang, W. Particle swarm optimization for traveling salesman problem. *Proc. Second Int. Conf. Mach. Learn. Cybern.* **3**, 1583–1585 (2003).
11. Dorigo, M. & Gambardella, L. M. Ant colony system: A cooperative learning approach to the traveling salesman problem. *IEEE Trans. Evol. Comput.* **1**, 53–66 (1997).
12. Hopfield, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. USA* **79**, 2554–2558 (1982).
13. Hopfield, J. J. & Tank, D. W. 'Neural' computation of decisions in optimization problems. *Biol. Cybern.* **52**, 141–152 (1985).
14. Krivan, M. & Budinska, B. Efficiency comparison of Hopfield network with simulated annealing as optimization methods for solving the traveling salesman problem. *Math. Appl.* **4**, 109–121 (2015).
15. Wilson, G. V. & Pawley, G. S. On the stability of the travelling salesman problem algorithm of Hopfield and Tank. *Biol. Cybern.* **58**, 63–70 (1988).
16. Kamgar-Parsi, B. & Kamgar-Parsi, B. On problem solving with Hopfield neural networks. *Biol. Cybern.* **62**, 415–423 (1990).
17. Johnson, M. W. *et al.* Quantum annealing with manufactured spins. *Nature* **473**, 194–198 (2011).
18. Inagaki, T. *et al.* A coherent Ising machine for 2000-node optimization problems. *Science* **354**, 603–606 (2016).
19. Yamaoka, M. *et al.* A 20k-spin Ising chip to solve combinatorial optimization problems with CMOS annealing. *IEEE J. Solid-State Circ.* **51**, 303–309 (2016).
20. Aramon, M. *et al.* Physics-inspired optimization for quadratic unconstrained problems using a digital annealer. *Front. Phys.* **7**, 48 (2019).
21. Goto, H., Tatsumura, K. & Dixon, A. R. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems. *Sci. Adv.* **5**, eaav2372 (2019).
22. Kadowaki, T. & Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev. E* **58**, 5355–5363 (1998).
23. Martoňák, R., Santoro, G. E. & Tosatti, E. Quantum annealing of the traveling-salesman problem. *Phys. Rev. E* **70**, 057701 (2004).
24. Choi, V. Minor-embedding in adiabatic quantum computation: I the parameter setting problem. *Quantum Inf. Process.* **7**, 193–209 (2008).
25. Choi, V. Minor-embedding in adiabatic quantum computation: II Minor-universal graph design. *Quantum Inf. Process.* **10**, 343–353 (2011).
26. Cai, J., Macready, B. & Roy, A. *A Practical Heuristic for Finding Graph Minors*. arXiv:1406.2741 (2014).
27. Klymko, C., Sullivan, B. D. & Humble, T. S. Adiabatic quantum programming: Minor embedding with hard faults. *Quantum Inf. Process.* **13**, 709–729 (2014).
28. Okada, S., Ohzeki, M., Terabe, M. & Taguchi, S. Improving solutions by embedding larger subproblems in a D-Wave quantum annealer. *Sci. Rep.* **9**, 2098 (2019).
29. Hamerly, R. *et al.* Experimental investigation of performance differences between coherent Ising machines and a quantum annealer. *Sci. Adv.* **5**, eaau0823 (2019).
30. Nakagaki, T., Yamada, H. & Tóth, Á. Maze-solving by an amoeboid organism. *Nature* **407**, 470 (2000).
31. Tero, A. *et al.* Rules for biologically inspired. *Science* **327**, 439–442 (2010).

32. Kasai, S., Aono, M. & Naruse, M. Amoeba-inspired computing architecture implemented using charge dynamics in parallel capacitance network. *Appl. Phys. Lett.* **103**, 163703 (2013).
33. Saito, K., Suefuji, N., Kasai, S. & Aono, M. Amoeba-inspired electronic solution-searching system and its application to finding walking maneuver of a multi-legged robot. *Proc. 48th Int. Symp. Mult. Val. Log.* 127–131 (2018).
34. Aono, M., Hara, M. & Aihara, K. Amoeba-based neurocomputing with chaotic dynamics. *Commun. ACM* **50**, 69–72 (2007).
35. Zhu, L., Aono, M., Kim, S. J. & Hara, M. Amoeba-based computing for traveling salesman problem: Long-term correlations between spatially separated individual cells of *Physarum polycephalum*. *BioSystems* **112**, 1–10 (2013).
36. Aono, M. *et al.* Amoeba-inspired nanoarchitectonic computing: Solving intractable computational problems using nanoscale photoexcitation transfer dynamics. *Langmuir* **29**, 7557–7564 (2013).
37. Aono, M. *et al.* Amoeba-inspired nanoarchitectonic computing implemented using electrical Brownian ratchets. *Nanotechnology* **26**, 234001 (2015).
38. Zhu, L., Kim, S. J., Hara, M. & Aono, M. Remarkable problem-solving ability of unicellular amoeboid organism and its mechanism. *R. Soc. Open Sci.* **5**, 180396 (2018).
39. Aono, M. Amoeba-inspired combinatorial optimization machines. *Jpn. J. Appl. Phys.* **59**, 060502 (2020).
40. Chua, L. O. Memristor—the missing circuit element. *IEEE Trans. Circuit Theory* **18**, 507–519 (1971).
41. Strukov, D. B., Snider, G. S., Stewart, D. R. & Williams, R. S. The missing memristor found. *Nature* **453**, 80–83 (2008).
42. Hu, S. G. *et al.* Associative memory realized by a reconfigurable memristive Hopfield neural network. *Nat. Commun.* **6**, 7522 (2015).
43. Liu, B., Chen, Y., Wysocki, B. & Huang, T. Reconfigurable neuromorphic computing system with memristor-based synapse design. *Neural Process Lett.* **41**, 159–167 (2015).
44. Terabe, K., Hasegawa, T., Nakayama, T. & Aono, M. Quantized conductance atomic switch. *Nature* **433**, 47–50 (2005).
45. Saito, K. & Kasai, S. Effect of feedback delays on solution quality in amoeba-inspired computing system that solves traveling salesman problem. *Appl. Phys. Express* **13**, 114501 (2020).

Acknowledgements

We thank Prof. Tadao Nakamura at Keio University and Stanford University for his significant comments. This work was partly supported by JSPS KAKENHI Grant number H8H01487, PRESTO-JST Grant no. JPMJPR1321, and NEDO Grant no. 16101009-0.

Author contributions

All authors conceived and designed the experiments; K.S. carried out the experiments, collected the overall data, analyzed the data, and wrote the manuscript. All authors discussed the results and commented on the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information is available for this paper at https://doi.org/10.1038/s41598-020-77617-7.

Correspondence and requests for materials should be addressed to K.S. or S.K.

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.