Simulated bifurcation for higher-order cost functions

Taro Kanao and Hayato Goto

Frontier Research Laboratory, Corporate Research & Development Center, Toshiba Corporation, 1, Komukai-Toshiba-cho, Saiwai-ku, Kawasaki 212-8582, Japan

E-mail: taro.kanao@toshiba.co.jp

Received November 21, 2022; accepted December 13, 2022; published online December 30, 2022

High-performance Ising machines for solving combinatorial optimization problems have been developed with digital processors implementing heuristic algorithms such as simulated bifurcation (SB). Although Ising machines have been designed for second-order cost functions, there are practical problems expressed naturally by higher-order cost functions. In this work, we extend SB to such higher-order cost functions. By solving a problem having third-order cost functions, we show that the higher-order SB can outperform not only the second-order SB with additional spin variables, but also simulated annealing applied directly to the third-order cost functions. This result suggests that the higher-order SB can be practically useful.

© 2022 The Japan Society of Applied Physics

Special purpose hardware solvers for the Ising problem have been applied to combinatorial optimization problems. Such hardware solvers are referred to as Ising machines. Ising machines with analogue computers have attracted attention, such as quantum annealers with superconducting circuits and coherent Ising machines with pulse lasers. Also, Ising machines with digital processors have been developed based on, e.g., simulated annealing (SA), simulated bifurcation (SB), and other algorithms.

SB is a heuristic algorithm based on the numerical simulation of a classical counterpart of a quantum bifurcation machine, which is a quantum annealer using nonlinear oscillators. SB allows us to simultaneously update variables, and thus can easily be accelerated by parallel processing with, e.g., graphics processing units (GPUs) and field-programmable gate arrays (FPGAs). Two improved variants of SB have been reported: ballistic and discrete SBs (bSB and dSB). In addition, bSB and dSB with thermal fluctuations have been studied. SB has been benchmarked against other Ising machines.

Besides, SB has been applied to financial portfolio optimizations, financial trading machines, traveling salesman problems, clustering with a hybrid method, and wireless communications.

In the Ising problem, the aim is to minimize cost functions written in second-order polynomials of spin variables, for which Ising machines have been designed. However, some combinatorial optimization problems are naturally expressed with higher-order polynomials than second order (higher-order cost functions). To apply Ising machines to such problems, higher-order cost functions have been written in second-order polynomials of spin variables, for which Ising machines have been designed. However, some combinatorial optimization problems are naturally expressed with higher-order polynomials than second order (higher-order cost functions). To apply Ising machines to such problems, higher-order cost functions have been transformed to second-order ones, though such transformations usually require additional spin variables and thus increase computational costs.

In this work, we extend SB to apply it directly to a problem with higher-order cost functions. Since SB is based on simulation, this approach can easily be implemented. To benchmark the higher-order SB, we use a three-regular three-XORSAT (3R3X) problem, because this problem yields third-order cost functions and its solution can be known in advance. We compare the third-order SB with that transformed to second order. As a result, we find that third-order bSB and dSB show higher performance than the second-order ones (2bSB and 2dSB). Surprisingly, 3bSB performs better than 3dSB, while 2dSB does better than 2bSB as in the previous study. We further find that 3bSB can be superior to SA applied directly to the third-order cost functions. These results suggest that the higher-order SB will offer another useful approach to special purpose hardware solvers for combinatorial optimization problems.

A third-order cost function with \( N \) spin variables \( s_i = \pm 1 \) can generally be written as

\[
E(s) = -\sum_i h_i s_i - \sum_{ij} J_{ij} s_i s_j - \sum_{ijk} K_{ijk} s_i s_j s_k,
\]

where \( x \) is a vector consisting of \( s_i \) and \( h_i, J_{ij}, \) and \( K_{ijk} \) are coefficients. Since \( x_i^2 = 1 \), we assume without loss of generality that \( J_{ii} = 0 \) and \( K_{ijk} = 0 \) if two of \( i, j, \) and \( k \) are equal.

We here extend second-order bSB and dSB to third-order ones. The third-order bSB and dSB are based on the following equations of motion for positions \( x_i \) and momenta \( p_i \) corresponding to \( s_i \):

\[
y_i = -[a_0 - a(t)] x_i + c f_i,
\]

\[
x_i = a_0 y_i,
\]

where the dots denote time derivatives, \( a_0 \) is a constant parameter, \( a(t) \) is a bifurcation parameter increased from 0 to \( a_0 \) with the time \( t \), and \( c \) is a normalization factor determined from forces \( f_i \). The forces \( f_i \) are given by

\[
f_i = -\frac{\partial E(x)}{\partial x_i} = G_i(x) \quad (\text{bSB}),
\]

\[
f_i = \left. \frac{\partial E(x)}{\partial x_i} \right|_{x = x_0} = G_i(s) \quad (\text{dSB}).
\]

For 3bSB, \( f_i \) are the gradients \( G_i(x) \) of \( E(x) \), while for 3dSB, \( x \) in the gradients is substituted by \( s \), which is a vector of the signs of \( x_i \), \( s_i = \text{sgn}(x_i) \). Both 3bSB and 3dSB assume perfectly inelastic walls at \( x_i = \pm 1 \), that is, if \( |x_i| > 1 \), \( x_i \) and \( y_i \) are set to \( \text{sgn}(x_i) \) and 0, respectively. Equations (2) and (3) are solved with the symplectic Euler method, which discretizes the time with an interval \( \Delta t \) and computes time evolutions of \( x_i \) and \( y_i \) step by step. The signs of \( x_i \) at the final time give a solution. The solution at least corresponds to a local minimum of \( E(s) \) for 2bSB and 2dSB, because Eq. (2) becomes \( \dot{y}_i = c_0 f_i \) at the final time. Note that when all
\[ K_{ijk} \text{ are zero in Eq. (1), Eqs. (2)-(5) reproduce 2bSB and 2dSB in Ref. 19. We thus focus on the third term in Eq. (1) in the following.}

To reduce the computational cost of calculating \( f_r \), we utilize sparse connectivity of the 3R3X problem, that is, the number, \( M \), of nonzero \( K_{ijk} \) is much smaller than the total number of \( K_{ijk} \). We expect that sparse connectivities often appear in applications and therefore the following method will be useful. By denoting nonzero \( K_{ijk} \) by \( K_{nm} \) \((m = 1, 2, \ldots, M)\), \( E(s) \) is expressed as

\[ E(s) = -\sum_{m=1}^{M} K_{nm}s_{n1}s_{n2}s_{n3}, \tag{6} \]

where \( v_{nm} \) with \( n = 1, 2, 3 \) represent the indices of spin variables included in the \( n \)th term. To further reduce the time required for evaluating the third-order SB, we calculate the gradients by

\[ G_i(s) = \frac{1}{x_i + \epsilon} \sum_{m=1}^{M} a_{im}K_{nm}s_{n1}s_{n2}s_{n3} \tag{7} \]

\[ G_i(s) = \frac{x_i}{x_i + \epsilon} \sum_{m=1}^{M} a_{im}K_{nm}s_{n1}s_{n2}s_{n3} \tag{8} \]

\[ a_{im} = \delta_{i,n1} + \delta_{i,n2} + \delta_{i,n3}, \tag{9} \]

where \( \delta_{ij} = 1 \) if \( i = j \) and otherwise 0. Equation (8) exactly gives the derivative in Eq. (5), because \( a_{im} \) pick up terms including \( s_i \) from Eq. (6), and these \( s_i \) are canceled by multiplying \( x_i \) in Eq. (8) with \( s_i^2 = 1 \). This method is valid since Eq. (6) is linear with respect to each \( s_i \). Similarly, Eq. (7) yields the same value as Eq. (4) if \( x_i \neq 0 \) and \( \epsilon = 0 \). To stabilize calculations, we set \( \epsilon = 10^{-14} \). We empirically found that Eqs. (7) and (8) could reduce the evaluation time compared with direct differentiation, which may be because these equations gather three terms derived from \( x_{n1}x_{n2}x_{n3} \) together and thus reduce memory accesses for \( x_{nm} \) and \( s_{nm} \).

The 3R3X problem is mapped to a minimization problem of Eq. (6) with the following \( v_{nm} \) and \( K_{nm} \). Here, \( M \) equals \( N \). For each of \( n = 1, 2, 3 \), the vector of indices \( v_{nm} \) is a permutation of 1, 2, \( \ldots, N \). For each \( m \), three indices \( v_{m1}, v_{m2}, \) and \( v_{m3} \) are different from each other. For different \( m \), the combinations of \( v_{m1}, v_{m2}, \) and \( v_{m3} \) are different. \( K_{nm} \) is given by \((−1)^{s_i}b_m\) with \( b_m = 0, 1 \). Thus \( E(s) \geq −N \) holds.

For a 3R3X instance, a solution \( s_0 \) minimizing \( E(s) \) can be obtained with linear algebra,\(^{34}\) that is, \( s_0 \) is related to a solution, \( \xi_i = 0, 1 \), of linear equations,

\[ \sum_{i=1}^{N} a_{im}\xi_i = b_m, \mod 2, \tag{10} \]

by \( s_0 = (−1)^{\xi_i} \). However, depending on \( a_{im} \) and \( b_m \), Eq. (10) may have no solution. To ensure that Eq. (10) has at least one solution, we instead give \( a_{im} \) and \( \xi_i \), and determine \( b_m \) with Eq. (10). Then an exact solution \( s_0 \) is indicated by \( E(s_0) = −N \).

A third-order cost function of the 3R3X problem can be decomposed to second-order one by mapping a term \((−1)^{s_i}s_j s_k s_l\) to

\[ h(s_1 + s_2 + s_3) + h\bar{s} + J(s_1s_2 + s_2s_3 + s_3s_1) + J(s_1 + s_2 + s_3)\bar{s}, \tag{11} \]

where \( \bar{s} = ±1 \) is an ancillary spin variable, and \( J, J, h \), and \( h \) are coefficients determined so that \( s_1, s_2, \) and \( s_3 \) minimizing Eq. (11) coincide with ones minimizing \( −(−1)^{s_i}s_j s_k s_l \). We use \( J = 1/4, \bar{J} = (−1)^3/2, h = (−1)^3/4, \) and \( \bar{h} = −1/2 \), which normalize the minimum of Eq. (11) to −1. The ancillary spin variables increase the total number of spin variables to \( 2N \).

We also directly apply SA to third-order cost functions (3SA). In an update in SA,\(^3\) one calculates a change in a cost function \( ΔE_t \) caused by a flip of \( s_i \). The flip is accepted if \( \betaΔE_t < −\ln R \), where \( \beta \) and \( R \) are the inverse temperature increased during annealing and a random number in the interval \( (0, 1) \), respectively. In a step of SA, we sequentially update spin variables \( s_i \).\(^{30} \)

In numerical experiments, we measure the computational costs of finding exact solutions as follows. We randomly generate 100 3R3X instances for each \( N \). For these instances, we run the above-mentioned SB and SA many times with different initial conditions or random numbers, and obtain probabilities of finding exact solutions, \( P \). Then, using the number of steps in a run, \( N_\text{r} \), we estimate the number of steps necessary to obtain exact solutions with a probability of 99\% (step-to-solution) by\(^{1,20} \)

\[ S = N_\text{r} \log 0.01 \log(1 − P). \tag{12} \]

A smaller \( S \) indicates that an algorithm provides an exact solution more efficiently (or faster if computation time for each step is assumed to be the same).

We set the parameters as follows. For SB, \( a_0 = 1 \), and \( a(t) \) is increased linearly. To normalize the term \( c f_i \) in Eq. (2), we give \( c \) by\(^{40} \)

\[ c = c_0 \left( \frac{1 + \nu}{\nu} \right)^{-1/2}, \tag{13} \]

where \( c_1 \) and \( \nu \) are a constant parameter and the number of spin variables, respectively (\( \nu = 2N \) and \( N \) for the second- and third-order SBs, respectively). We optimize \( N_\text{r} \) for each \( N \) and \( \Delta t, c_t \) for certain large values of \( N_\text{r} \) to minimize the median of \( S \) for the 100 instances.\(^{23} \) Initial \( x_i \) and \( y_i \) are generated by uniform random numbers in the interval \( (−1, 1) \). To suppress a statistical error in \( P \), the number of runs \( N_\text{r} \) is set to be large such that \( N_\text{r}P \gg 1 \),\(^3\) though this condition is not necessarily met for the largest \( N \) in this work owing to limitations of computational resources. For SA, \( \beta \) is linearly increased from zero to a final value \( \beta_1 \) and \( \beta_1 \) and \( N_\text{r} \) are optimized as in SB.

We first present typical distributions of \( E(s) \) for a 3R3X instance, for which 3bSB gives \( S \) near the median of the 100 instances. Figure 1 shows the distributions by the rates of runs resulting in \( E(s) \). A higher value of the rate means a higher probability of finding solutions with certain accuracy, though here the parameters are optimized to minimize \( S \) as mentioned above and the rates can be improved by increasing \( N_\text{r} \). Interestingly, Fig. 1(a) shows that the rate for small \( N_\text{r} \) is higher for 3bSB than for 3dSB despite smaller \( N_\text{r} \) for 3bSB. On the other hand, Fig. 1(b) shows that the rate is higher for 2dSB than for 2bSB in accordance with much larger \( N_\text{r} \) for 2dSB. As shown in Fig. 1(a), the rate for 3SA is between
those of 3bSB and 3dSB. These rates and \( N_s \) suggest the highest performance of 3bSB.

We next evaluate the results for 100 3R3X instances. Figure 2 shows the medians of \( S \) as functions of \( N \). All the algorithms exhibit exponential increases of \( S \) with \( N \). The values of \( S \) are smaller for third-order bSB and dSB than for the second-order ones. Among the second-order ones, 2dSB gives the smaller values of \( S \) than 2bSB, which is consistent with a reported result that dSB more often reaches exact solutions for the Ising problem. Among the third-order SB, in contrast, 3bSB shows the smaller values of \( S \) than 3dSB. The higher performance of 3bSB might be because bSB can rapidly converge to a local minimum of a cost function and a local search algorithm is effective for the 3R3X problem. A comparison between 2bSB and 3bSB suggests that the local search by bSB is especially effective when directly applied to third-order cost functions with fewer spin variables. 3SA results in \( S \) between those of 3bSB and 3dSB, indicating that 3bSB is superior to 3SA.

Scaling can be measured by exponents \( \alpha \) and \( \beta \) such that

\[
S \approx 10^{\alpha N + \beta},
\]  

for large \( N \). A smaller \( \alpha \) allows an algorithm to be applied to larger instances. We fit the medians of \( S \) by Eq. (14) with the least squares method as the lines in Fig. 2. Here, only the data for intermediate \( N \) are used for the fittings, because errors due to finite-size effects and statistical errors are large for small and large \( N \), respectively. Table I shows \( \alpha \) and \( \beta \). The value of \( \alpha \) for 3bSB is the smallest. \( \alpha \) for 3dSB is a little larger than it, and \( \alpha \) for 2dSB is significantly larger. \( \alpha \) for 2bSB is by far the largest. The value of \( \alpha \) for 3SA is larger than those for 3bSB and 3dSB. These values of \( \alpha \) demonstrate that 3bSB scales the best among these algorithms.

Finally, we compare \( \alpha \) with the results of Ref. 22, where various Ising machines are benchmarked using the 3R3X problem expressed by second-order cost functions. Among the Ising machines, Digital Annealer reported the smallest exponent corresponding to \( \alpha = 0.0370(8) \). Compared with this value, \( \alpha \) for 3bSB seems to be smaller. \( \alpha \) for Simulated Bifurcation Machine (provided by Toshiba Digital Solutions Corporation) corresponds to 0.0434 (12), which coincides with that for 2dSB in this work within the range of errors. This coincidence may support the validity of our evaluation. In Ref. 22, a quasi-greedy algorithm called SATonGPU is also compared, which is specialized for the 3R3X problem and deals with third-order cost functions directly. As a result, SATonGPU provided the exponent corresponding to \( \alpha = 0.0341(2) \), which is close to an estimated optimal exponent. The value of \( \alpha \) for SATonGPU is smaller than \( \alpha \) for 3bSB in our work, but the difference is fairly small. This implies that 3bSB can achieve almost optimal scaling even though 3bSB is not specialized for the 3R3X problem.

In summary, we have demonstrated that third-order SB can perform better than second-order one for a 3R3X problem. Although 2dSB provides better performance than 2bSB, 3bSB instead does better than 3dSB. Furthermore, we have shown that 3bSB can be superior to 3SA. These results indicate that higher-order SB can be useful for solving combinatorial optimization problems expressed naturally by higher-order cost functions. Since we have formulated the higher-order SB by matrix-vector operations, we expect that the higher-order SB may be accelerated by parallel processing with, e.g., GPUs and FPGAs.

Table I. Scaling exponents. The numbers in the parentheses are standard deviations in the least squares method.

|       | 3bSB          | 3dSB          | 3SA           | 2dSB          | 2bSB          | 3dSB          |
|-------|---------------|---------------|---------------|---------------|---------------|---------------|
| \( \alpha \) | 0.0355(2)     | 0.0375(6)     | 0.0384(6)     | 0.078(1)      | 0.0433(2)     |               |
| \( \beta \)  | 3.36(4)       | 4.23(8)       | 3.69(8)       | 3.19(9)       | 4.72(2)       |               |

Fig. 1. (Color online) Distributions of a third-order cost function \( E(s) \) for a 3R3X instance with 100 spin variables. (a) 3bSB, 3dSB, and 3SA. (b) 2bSB and 2dSB. The lines are eye guides. The numbers of steps \( N_s \) are shown in parentheses. Here, \( N_s \) and the following parameters are determined to minimize step-to-solution \( S \). The parameters are (\( \Delta t, c_1 \)) = (1.1, 0.7) for 3bSB, (0.7, 1) for 3dSB, (0.8, 0.9) for 2bSB, (0.7, 1.6) for 2dSB, and \( \beta_1 = 2 \) for 3SA.

Fig. 2. (Color online) The medians of step-to-solution \( S \) for 100 3R3X instances as functions of the number of spin variables \( N \). The lines and error bars show fittings by Eq. (14) and statistical errors, respectively. Parameters \( \Delta t, c_1, \) and \( \beta_1 \) are the same as in Fig. 1.
Acknowledgments

We thank Y. Sakai, T. Kashimata, K. Tatsumura, and R. Hidaka for helpful discussions.

ORCID iDs

Taro Kanao https://orcid.org/0000-0003-3462-8288
Hayato Goto https://orcid.org/0000-0003-1145-6620

1) F. Barahona, J. Phys. A: Math. Gen. 15, 3241 (1982).
2) A. Lucas, Front. Phys. 2, 5 (2014).
3) N. Mohseni, P. L. McMahon, and T. Byrnes, Nat. Rev. Phys. 4, 363 (2022).
4) T. Kadowaki and H. Nishimori, Phys. Rev. E 58, 5355 (1998).
5) M. W. Johnson et al., Nature 473, 194 (2011).
6) A. Marandi, Z. Wang, K. Takata, R. L. Byer, and Y. Yamamoto, Nat. Photonics 8, 937 (2014).
7) S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Science 220, 671 (1983).
8) M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H. Mizuno, IEEE J. Solid-State Circuits 51, 303 (2016).
9) M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, Front. Phys. 7, 48 (2019).
10) H. Goto, K. Tatsumura, and A. R. Dixon, Sci. Adv. 5, eaav2372 (2019).
11) T. Leleu, F. Khoyratee, T. Levi, R. Hamerly, T. Kohno, and K. Aihara, Commun. Phys. 4, 266 (2021).
12) H. Goto, Sci. Rep. 6, 21686 (2016).
13) S. Puri, C. K. Andersen, A. L. Grimsmo, and A. Blais, Nat. Commun. 8, 15785 (2017).
14) H. Goto, J. Phys. Soc. Jpn. 88, 061015 (2019).
15) T. Kanao and H. Goto, npj Quantum Inf. 7, 18 (2021).
16) K. Tatsumura, A. R. Dixon, and H. Goto, IEEE Int. Conf. Field Programmable Logic and Applications (FPL), 2019, p. 59.
17) Y. Zou and M. Lin, ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays (FPGA’20), 2020, p. 65.
18) K. Tatsumura, M. Yamasaki, and H. Goto, Nat. Electron. 4, 208 (2021).
19) H. Goto, K. Endo, M. Suzuki, Y. Sakai, T. Kanao, Y. Hamakawa, R. Hidaka, M. Yamasaki, and K. Tatsumura, Sci. Adv. 7, eabe7953 (2021).
20) T. Kanao and H. Goto, Commun. Phys. 5, 153 (2022).
21) H. Oshiyama and M. Ohzeki, Sci. Rep. 12, 2146 (2022).
22) M. Kowalsky, T. Albash, I. Hen, and D. A. Lidar, Quantum Sci. Technol. 7, 025008 (2022).
23) K. Steinhauser, T. Fukudai, and S. Yoshida, arXiv:2009.08412 (2020).
24) J. Cohen and C. Alexander, arXiv:2011.01308 (2020).
25) K. Tatsumura, R. Hidaka, M. Yamasaki, Y. Sakai, and H. Goto, IEEE Int. Symp. Circuits and Systems (ISCAS), 2020, p. 1.
26) T. Zhang, Q. Tao, and J. Han, IEEE Int. SoC Design Conf. (ISOCDC), 2021, p. 288.
27) N. Matsumoto, Y. Hamakawa, K. Tatsumura, and K. Kudo, Sci. Rep. 12, 2699 (2022).
28) W. Zhang and Y.-L. Zheng, arXiv:2210.14660 (2022).
29) V. Kolmogorov and R. Zabih, IEEE PAMI 26, 147 (2004).
30) H. Ishikawa, IEEE PAMI 33, 1234 (2011).
31) R. Xia, T. Bian, and S. Kais, J. Phys. Chem. B 122, 3384 (2018).
32) N. Dattani, arXiv:1901.04405 (2019).
33) J. Fujisaki, H. Oshima, S. Sato, and K. Fujii, Phys. Rev. Res. 4, 043086 (2022).
34) I. Hen, Phys. Rev. Appl. 12, 011003 (2019).
35) M. Bernaschi, M. Bissou, M. Fatica, E. Marinari, V. Martin-Mayor, G. Parisi, and F. Ricci-Tersenghi, Europhys. Lett. 133, 60005 (2021).
36) M. Bellitti, F. Ricci-Tersenghi, and A. Scardicchio, Phys. Rev. Res. 3, 043015 (2021).
37) B. Leinikahler and S. Reich, Simulating Hamiltonian Dynamics (Cambridge University Press, Cambridge, 2004).
38) M. Leib, P. Zoller, and W. Lechner, Quantum Sci. Technol. 1, 015008 (2016).
39) S. V. Isakov, I. N. Zintchenko, T. F. Rønnow, and M. Troyer, Comput. Phys. Commun. 192, 265 (2015).
40) Y. Sakai, H. Goto, K. Tatsumura, K. Endo, and M. Suzuki, International Patent Application PCT/JP2020/008641 (2020).
41) In Ref. 22, scaling is given with respect to $\nu = 2N$.
42) See https://aws.amazon.com/marketplace/pp/prodview-F3hbaz4q3y32y for Simulated Bifurcation Machine by Toshiba Digital Solutions Corporation; accessed 21 November 2022.
43) In Ref. 35, scaling is expressed with $e^{aN}$. 

© 2022 The Japan Society of Applied Physics