A Parallel Tempering Processing Architecture with Multi-Spin Update for Fully-Connected Ising Models

Yang Zhang, Xiangrui Wang, Dong Jiang, Zhanhong Huang, Gaopeng Fan, and Enyi Yao
School of Microelectronics, South China University of Technology, Guangzhou, China

***Abstract*—Combinatorial optimization problems (COPs) are notoriously difficult to solve for classic Von-Neumann computers, which are ubiquitous in various domains. As a state-of-the-art hardware acceleration scheme for COPs, Ising machines are one of the promising research directions for the next generation of computing, but still suffer from the low solution accuracy and speed due to the high complexity of the fully-connected Ising model. In this work, a novel parallel tempering processing architecture (PTPA) is proposed with the modified parallel tempering algorithm, aimed at reducing search time and improving the solution quality. Several techniques are developed to further reduce hardware overhead and enhance parallelism, including the independent pipelined spin update architecture, approximated probability equations, and compact random number generators. Its prototype is implemented on FPGA with eight replicas, each replica containing 1,024 fully-connected spins and at most 64 concurrent update spins. The proposed design achieves an average cut accuracy of 99.43% within 1ms solution time on various G-set problems. Compared with the CPU-based parallel tempering implementation, it enhances the speed of solving the max-cut problems by 5,160 times.**

***Index Terms*—Combinatorial optimization, hardware acceleration, annealing processing architecture, parallel tempering, Ising model**

I. Introduction

Combinatorial optimization problems (COPs) can be found in various fields such as economic management, machine learning and communication network. Due to the fact that most of them are non-deterministic polynomial time hardness (NP-hard) problems [1], the corresponding solution space will increase exponentially as the problem size grows. Therefore, these problems are notoriously difficult for modern Von-Neumann computers when a brute-force search method is used [2].

In recent years, some annealing architectures are specially designed to handle this issue based on the Ising model [3]–[12]. COPs can be mapped to the Ising model and the optimal solution can be obtained with its ground state [1] by an effective and hardware-implementation-friendly algorithm, as shown in Fig. 1. Specifically, CMOS-based annealing processors offer a significant potential for COPs with strong adaptability, low cost and high stability. However, most of the published works are more focused on the locally-connected Ising model [3], [4], [12], whose application scenarios are greatly limited by the sparsity of the connection between the spins. In [5], Fujitsu developed an annealing processing architecture based on the fully-connected Ising model on the FPGA platform, enabling effective solutions to COPs. However, this architecture's performance is still limited by the simulated annealing (SA) algorithm, which only allows a single-spin-update at a time due to data dependency. Other subsequent works have proposed different methods [7], [8], [11] to support the Ising model for multi-spin update during one iteration. However, as SA is a pseudo-Monte Carlo sampling method [13], these improvements do not circumvent the inherent limitations of traditional SA. Without defining ensemble averages in SA, annealing process typically requires a slow reduction of temperature so that the system approximately reaches equilibrium with each iteration to achieve a better solution performance.

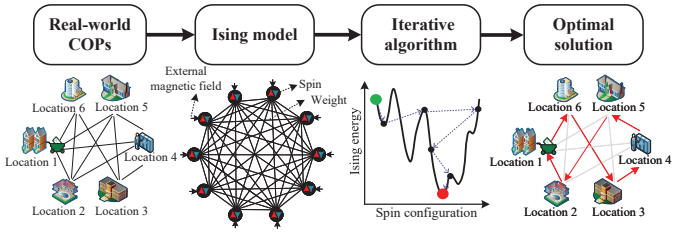

Fig. 1. The process of solving COPs using an iterative algorithm based on the Ising model.

In this paper, a novel parallel tempering processing architecture (PTPA) is proposed. This architecture is based on the fully-connected Ising model to solve the COPs. The proposed modified parallel tempering (PT) algorithm supports multi-spin concurrent updates per replica and employs an efficient temperature exchange strategy to accelerate the energy convergence of Ising model. Furthermore, an independent pipelined spin update architecture is designed for each replica, attaining an improved efficiency of parallel processing. The proposed PTPA is implemented on FPGA and compared with recent related works.

The rest of this paper is organized as follows. Section II presents the relevant theoretical basis. Section III describes the proposed modified PT algorithm. Section IV describes the hardware implementation of the proposed PTPA. Section V evaluates the performance of PTPA. Section VI summarizes this work finally.

II. Background

In the fully-connected Ising model, each spin has a connection with all the other spins. The total energy or Hamiltonian

This work was supported in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2022A1515110045 and Grant 2023A1515011241; and in part by the Foundation of Key Laboratory of Artificial Intelligence, Ministry of Education, China. Enyi Yao is the corresponding author (yaoenyi@scut.edu.cn).

979-8-3503-4859-0/DATE24/© 2024 EDAA

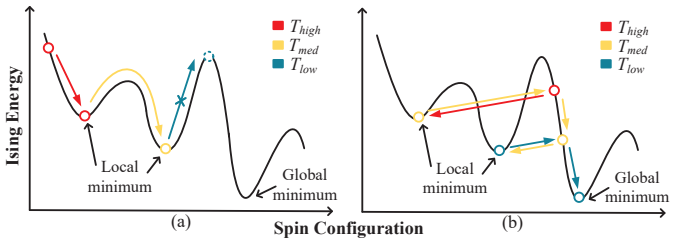

Fig. 2. Comparison of SA and PT processes. (a) SA process. (c) PT process.

of the system H, is described as follows:

$$H = -\sum_{i,j} W_{ij}\sigma_i\sigma_j - \sum_i b_i\sigma_i, \tag{1}$$

where $\sigma_i \in \{-1, 1\}$, represents spin i or its state, W_{ij} is the connection weight between σ_i and σ_j, and b_i represents the external magnetic field of σ_i. If the state of spin i is flipped, the system energy change will be

$$\Delta H_i = 2\sigma_i h_i, \tag{2}$$

$$h_i = \sum_j W_{ij}\sigma_j + b_i, \tag{3}$$

where h_i is the local field of σ_i. SA is often employed in the Ising model for energy minimization. The Metropolis sampling criteria is used to determine whether the spin should be flipped in each iteration step [14]:

$$P_{flip} = \min\left\{1, \exp\left(-\frac{\Delta H_i}{T}\right)\right\}, \tag{4}$$

where the temperature T decreases as the iteration proceeds. Note that this is a serial update scheme, and as the temperature decreases, the probability of the system moving towards higher energy decreases. If the system is at a local minimum with a low temperature, it will be difficult to escape the current state, i.e., SA will lose its effectiveness, as shown in Fig. 2 (a).

In PT, each system, or replica, undergoes a simulation in the canonical ensemble with distinct thermodynamic states. Higher-temperature systems are capable of traversing energy barriers, while lower-temperature systems primarily explore local energy minimum. PT attempts to swap these systems that belong to different thermodynamic states. It should be stressed that the swap moves maintain the Boltzmann distribution associated with a specific ensemble. Therefore ensemble averages from every individual ensemble can be determined just as an ordinary Monte Carlo simulation [13], [15], which means that real-equilibrium Monte Carlo sampling is reached.

M replicas iterate according to the Metropolis sampling criteria at M specific temperatures, and then a random temperature $T_k, k \in \{1, 2, ..., M-1\}$ is periodically selected to exchange configuration with its neighbour based on the possibility equation [13], [16]:

$$P_{swap} = \min\left\{1, \exp\left(\Delta\beta\Delta H\right)\right\}, \tag{5}$$

$$\Delta\beta = \frac{1}{T_{k+1}} - \frac{1}{T_k}, \tag{6}$$

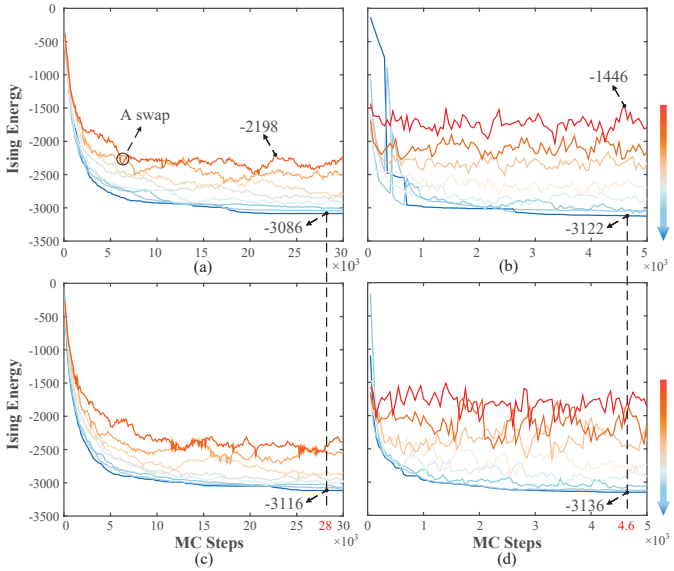

Fig. 3. The impact of the two schemes on the energy convergence. The color bar on the right shows the temperature from high to low. (a) PT. (b) PT + multi-spin concurrent update. (c) PT + multi-replica swap. (d) Modified PT.

$$\Delta H = H_{k+1} - H_k. \tag{7}$$

Fig. 2 (b) show the process of replica configuration swap between three temperatures. Through swapping, the replica at high temperature will have a higher probability of escaping the local minimum to explore more spin configurations, while the replica at low temperature will have a higher possibility to approach the global minimum.

III. Modified Parallel Tempering Algorithm

In this section, a modified PT algorithm is proposed built on two effective schemes to improve the solution speed and accuracy of PTPA.

Although PT can avoid local optimal solutions through parallel computation and replica exchange operations, each replica can still only update one spin per MC step, which ensures the convergence of the final result for the fully-connected Ising model. For this reason, a multi-spin concurrent update scheme for replicas to enhance the convergence speed is proposed. According to (5), the lowest-temperature replica will possess the lowest-energy spin configuration after multiple exchanges, so only this replica is considered as the final result. For replicas with other temperatures, the number of concurrent updates C could be gradually increased based on the temperature. A spin configuration with lower energy can be found quickly with higher concurrent updates. Subsequently, the swap operation transfers this configuration to a lower temperature, ensuring the convergence of the result.

In traditional PT, only a pair of adjacent temperatures is selected at a time to determine whether to exchange the replica configurations. This not only makes it challenging to identify suitable temperature pairs for exchange but also limits the possibility of exchanging configurations among multiple replicas. In particular, a multi-replica swap strategy is proposed with

faster convergence speed, higher exchange efficiency to match the concurrent update scheme. All adjacent temperature pairs are categorized into two groups, and the calculation of whether they should exchange the configurations can be completed in two steps. Specifically, for temperature T_{2i}, it will first try to swap configurations with T_{2i-1}, and then with T_{2i+1} in the next step.

Fig. 3 presents a clear overview of how the proposed two schemes work when solving max-cut problem G46. As a reference, Fig. 3 (a) employs an eight-replica PT. Multi-spin concurrent update scheme, multi-replica swap strategy, and both of them are adopted in PT respectively to experiment, as depicted in Fig. 3 (b), (c), (d). From Fig. 3 (b), it can be observed that the energy curves exhibit a wider range of fluctuations, allowing for a larger temperature interval to be set. By contrast, the traditional PT requires a smaller temperature spacing to ensure a suitable exchange probability between adjacent temperatures. It should be noted that these intersection points of the curves signify successful configuration exchanges for two adjacent replicas. By swapping, the lowest-temperature replica ultimately possesses the lowest-energy configuration. As shown in Fig. 3 (d), the modified PT demonstrates a better convergence speed (4.6×10^3 MC steps) and achieves a lower Ising energy (-3,136) benefiting from the incorporation of these two effective schemes. This indicates that a higher-quality solution is obtained in a shorter period of time.

IV. Hardware Implementation

In this section, the overall architecture is presented, followed by a comprehensive elucidation of the proposed PTPA. In particular, two key modules, the spin update module (SUM) and temperature swap module (TSM), designed for efficiently implementing the modified PT algorithm on hardware are detailed.

A. Overall Architecture

As shown in Fig. 4, the proposed PTPA consists of a global controller, a TSM, and eight replicas, with each replica implementing 1,024 fully-connected spins. The global controller is composed of an I/O module, iteration counters, a random selection unit, and finite state machines (FSMs). Data transmission between the user and the proposed architecture is carried out through the I/O module. The user data includes temperature parameters, initial spin configuration, random number seeds, initial local fields, the total number of swaps N_{swap} and the number of MC cycles L_{MC} between two swaps. The final result of the system is the spin configuration SC_MIN with the lowest energy upon completion of the run. During the process of updating the spins, each replica operates independently of the others, which significantly enhances the parallelism of the architecture. In each replica, the local field calculation modules (LCMs) are used to store and calculate the values of 1,024 local fields. The SUM determines the spins to be flipped according to the energy change. Additionally, the energy calculation module (ECM) accumulates the energy change resulting from the flipped spin. The accumulated energy in each replica's ECM is subsequently transmitted to the TSM, which determines

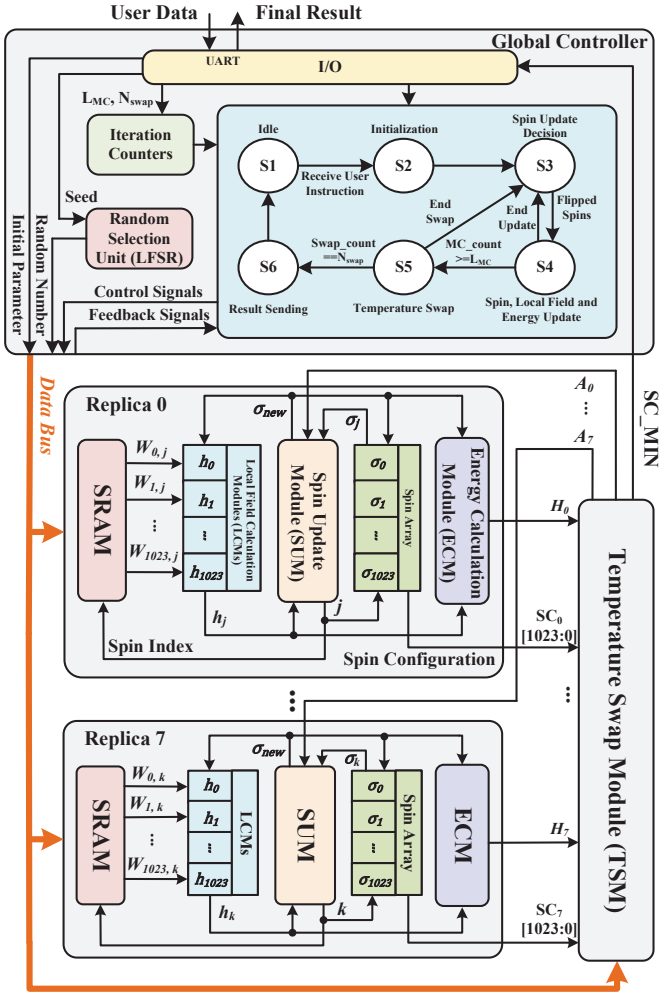

Fig. 4. Overall architecture of the proposed PTPA.

whether or not to exchange the temperature and returns the parameter A pertaining to the temperature to the replica. The spin configuration SC (i.e., 1,024 spin states) of the replica is also sent to the TSM. At the end of the run, the TSM returns SC_MIN to the user.

A pipelined design is employed to update the local fields for each replica. The pipeline comprises three stages, namely, producing flipped spins by SUM, reading the corresponding weight coefficients by SRAM, and calculating the local fields by LCM. Moreover, we execute the spin update, local field update, and energy update in parallel. In an MC step, considering the case of a flipped spin σ_j, the changes of local fields and energy are only affected by the connections between spin σ_j and other spins. As such, the local fields and energy are calculated by multiply-accumulate (MAC) operation in LCMs and ECM.

As shown in Fig. 5 (a), the timing diagram explains the datapath of the replica in detail. During the MC step, SUM reads 64 random local fields in parallel, determines whether the corresponding spins should be flipped, and sequentially outputs the flipped spins. The number of clock cycles required for an MC step is contingent on the quantity of flipped spins.

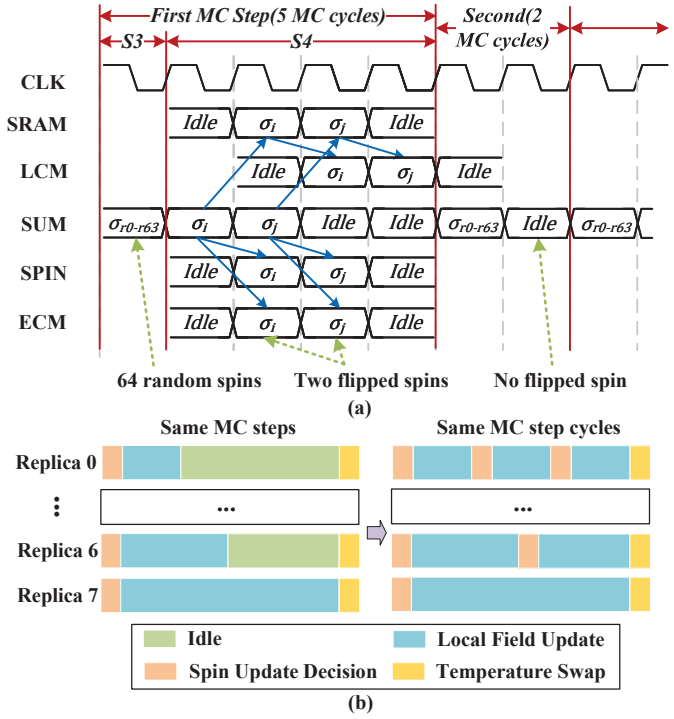

Fig. 5. (a) Timing diagram of a single replica. (b) Schematic diagram of the timing of eight replicas, with the proposed efficient iteration method shown on the right.

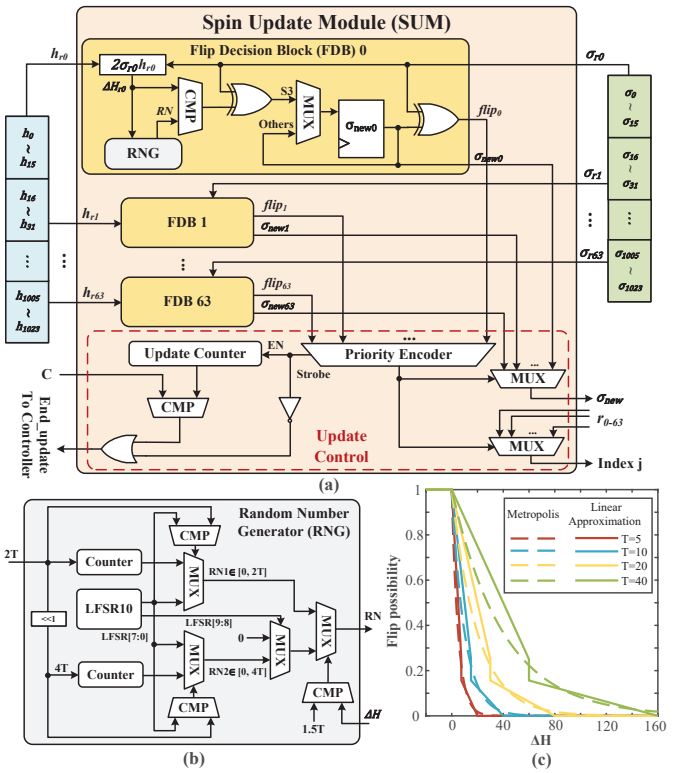

Fig. 6. (a) SUM. (b) RNG. (c) Spin flip probability curves.

Notably, in the absence of the flipped spins, an MC step can be completed in two cycles. Since different replicas may have different flipped spins, the number of MC cycles, denoted as L_{MC}, is used to specify the iteration between two swaps, instead of the number of MC steps N_{MC}. Fig. 5 (b) further explains this setup. If all replicas perform the same MC step, replicas with no flipped spins will have a lot of idle time. This is especially true for replicas at lower temperatures, where it is more difficult to search for flippable spins. By configuring an independent FSM for each replica and setting same MC step cycles, the timing of temperature swaps can be determined. This allows replicas without flipped spins to perform more MC steps, which greatly improves the iteration efficiency of each replica.

B. Spin Update Module

As shown in Fig. 6 (a), for the replica with 64 concurrent updates, it is necessary to implement 64 flip decision blocks (FDBs) for parallel decision. The flip flag bits $flip_{0-63}$ are then input into the priority encoder to decide the execution order of the flipped spins. The parameter C determines the maximum number of spins that can be updated by SUM per MC step.

The four-segment linear approximation of (4) is utilized to reduce the complexity of the hardware implementation:

$$P_{flip} = \begin{cases} 1, & (\Delta H_j{<}0) \\ -\frac{\Delta H_j}{2T} + 1, & (0 \le \Delta H_j{<}1.5T) \\ -\frac{\Delta H_j}{16T} + \frac{1}{4}, & (1.5T \le \Delta H_j{<}4T) \\ 0, & (\Delta H_j \ge 4T)\,. \end{cases} \tag{8}$$

Fig. 6 (c) illustrates a comparison between the original formula and the linear formula. When the probability $P \in (0, 1)$, the use of two different slopes provides a better approximation for both large and small values of ΔH. We approximate (5) in the same way, and conduct simulations to solve G46. The average cut value is 6,622.1 for the original formulation and 6,622.9 for the linear formulation, which demonstrates the effectiveness of employing a linear formula. The new state σ_{new} of σ_j is given by

$$\sigma_{new} = \begin{cases} -\sigma_j, & (\Delta H_j{<}1.5T \cap \Delta H_j \le RN1)\ or \\ & (\Delta H_j \ge 1.5T \cap \Delta H_j \le RN2) \\ \sigma_j, & others \end{cases} \tag{9}$$

where $RN1 \in [0, 2T]$ and $RN2 \in [-12T, 4T]$ are random numbers in the specified range. In FDB, the comparison between ΔH_j and RN determines whether the σ_j should be flipped. The new spin state can then be obtained by performing an XOR operation.

As shown in Fig. 6 (b), the RN is generated by a random number generator (RNG), which is an improvement on linear feedback shift register (LFSR) to generate a specified range of random numbers. $2T$ and $4T$ serve as the base of the two counters, respectively. LFSR uses the primitive polynomial to generate a maximum random sequence. Any number (LFSR[7:0]) above $2T/4T$ in the sequence will be replaced by the counter value, thus ensuring a uniform distribution of random numbers within the interval $[0, 2T]$ / $[0, 4T]$. It is unnecessary to generate

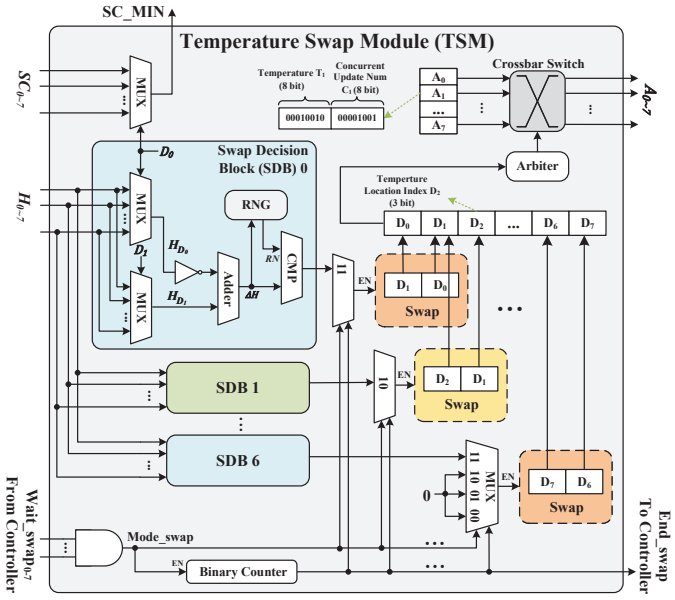

Fig. 7. Architecture of the proposed TSM.

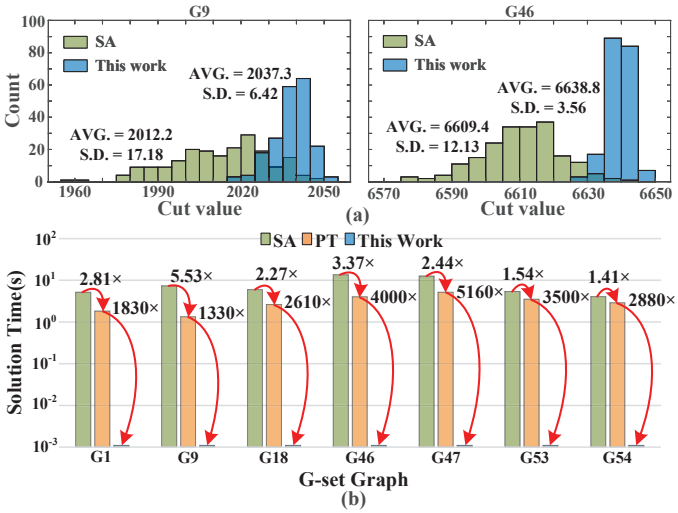

Fig. 8. (a) Comparision of the accuracy of SA and modified PT. (b) Comparison of the solution speed of the three algorithms, the speedup is shown next to the red line. The solution time of one experiment is used for comparison. The modified PT runs on FPGA, SA and PT run on the Intel i7-11700 CPU.

random numbers in $[-12T, 0]$ when $\Delta H_j \geq 1.5T$ according to (9). In the hardware implementation of RNG, $RN2$ is selected by a 2-bit random number (LFSR[9:8]) such that it has a $\frac{3}{4}$ probability of being replaced by 0, which achieves the same effect as a random number in $[-12T, 4T]$.

C. Temperature Swap Module

The detailed architecture of TSM is illustrated in Fig. 7. Seven swap decision blocks (SDBs) are used to determine whether the seven adjacent pairs of temperatures can be swapped. In the proposed TSM, we replace the replica configurations swap in the PT algorithm by temperatures swap, which means that the order of replicas does not correspond to the order of temperatures. Therefore, by introducing the temperature location index D, it ensures that the input of each SDB is the energy of a pair of adjacent temperatures. In the swap operation, a temperature exchange can be completed by exchanging the values of two index registers. Compared to swapping replica configurations (including 1,024 spin states, 1,024 spin local fields, and energy), this method significantly reduces the consumption of hardware resources.

According to the serial number of SDBs, they are divided into two types: even blocks (SDB 0, SDB 2, SDB 4 and SDB 6) and odd blocks (SDB 1, SDB 3 and SDB 5). Blocks with the same type can perform the swap operation simultaneously. After all replicas enter the temperature swap state, the swap of all adjacent temperature pairs can be done in two steps (i.e., two cycles). Finally, the exchanged temperature parameters A, which are a combination of two parameters T and C, are sent to the corresponding replica through the arbiter and crossbar switch.

V. Performance Evaluation

In this section, the performance of the proposed architecture is evaluated on FPGA. The PTPA prototype is designed with Verilog hardware description language (Verilog HDL). It comprises eight replicas, each consisting of 1,024 fully-connected spins and supporting at most 64 concurrent update spins. We synthesize and implement the design using Xilinx Vivado design suit. The resource utilization report for the hardware implementation shows that approximately 99k LUTs, 74k Registers, and 228 BRAMs are utilized in the proposed PTPA. The total power consumption is 6.16W, which contains of 3.19W static power and 2.97W dynamic power. We run the PTPA prototype on the Xilinx Virtex Ultrascale+ HBM VCU128 Evaluation Platform (XCVU37P-FSVH2892 FPGA) and controlled by the host computer through a UART. Benefiting from the linear approximation of the probability formulas and a pipelined design, this architecture runs at 200MHz.

We compared the proposed modified PT with conventional SA and PT on the G-set max-cut problems [17]. The experimental results demonstrate that our design achieves faster convergence rate and higher solution accuracy. In particular, the distributions of cut values obtained by running 200 times with SA (4×10^5 MC steps) and modified PT ($N_{swap} = 800$ and $L_{MC} = 250$) are shown in Fig. 8 (a). Furthermore, Fig. 8 (b) illustrates a comparison of the solution speed of these three algorithms in the event of them achieving the same average cut value (see Table I). For each G-set problem, PT has a faster convergence speed than SA, and the proposed one with a greater advantage. Since PTPA runs at 200 MHz on the FPGA, the time required for each experiment is about 1ms. Compared to the CPU implementation of the PT algorithm, the proposed algorithm achieves 5,160× speedup (G47) with PTPA.

We further demonstrate the excellent performance of PTPA through a comparative analysis with recent related works. Table I shows the best and average cut values obtained by using PTPA to solve multiple G-set max-cut problems. Moreover, the table includes the cut values acquired from [6] (FPGA)

TABLE I
COMPARISON OF THE ACCURACY ON THE MAX-CUT PROBLEM WITH RECENT RELATED WORKS

Graph	Vertices	Best Known Cut	[6] (Accuracy) Best/Average		[10] (Accuracy) Best/Average		This work (Accuracy) Best/Average	
G1	800	11,624	**11,624**	11,601 (99.80%)	-	-	**11,624**	11,610 (99.88%)
G2	800	11,620	**11,620**	11,591 (99.75%)	-	-	**11,620**	11,602 (99.85%)
G9	800	2,054	**2,054**	2,036 (99.12%)	2,050	2,010 (97.9%)	2,051	2,037.3 (99.19%)
G18	800	992	989	976 (98.39%)	987	974 (98.2%)	987	979.14 (98.70%)
G19	800	906	903	891 (98.34%)	902	890 (98.2%)	902	893.96 (98.67%)
G47	1,000	6,657	-	-	6,653	6,628 (99.6%)	**6,657**	6,647.6 (99.86%)
G51	1,000	3,848	3,840	3,819 (99.25%)	3,844	3,827 (99.5%)	3,830	3,819.1 (99.25%)
G52	1,000	3,851	3,842	3,821 (99.22%)	-	-	3,829	3,818.7 (99.16%)
G53	1,000	3,850	3,842	3,821 (99.25%)	3,844	3,831 (99.5%)	3,838	3,829.4 (99.46%)
G54	1,000	3,852	3,837	3,820 (99.17%)	3,845	3,828 (99.4%)	3,838	3,827.3 (99.36%)

TABLE II
COMPARISON BETWEEN PTPA AND OTHER ISING MODEL-BASED PROCESSING ARCHITECTURES

Design	Technology	Topology	Iterative Algorithm	Number of Spins	Power (W)	Clock (MHz)	Solution Time(ms)
[4]	CMOS 40nm	King's Graph	Simulated Annealing	30k/chip	N/A	100	0.022
[11]	CMOS 28nm	Fully-Connected	Pseudo Annealing	512	0.012	1	128
[5]	FPGA	Fully-Connected	Simulated Annealing	1,024	N/A	100	∼166
[6]	FPGA	Fully-Connected	Simulated Annealing	1,024	3.732	100	0.373-5.38
[7]	FPGA	Fully-Connected	Simulated Annealing	4k	4.7	200	68-121
[9]	FPGA	Fully-Connected	Simulated Adaibatic Bifurcation	2k	∼40	229-248	0.5
[18]	FPGA	Fully-Connected	Simulated Quantum Annealing	4k	N/A	244-256	4,610-5,810
This work	FPGA	Fully-Connected	Parallel Tempering	1k×8	6.16	200	0.5-1

and [10] (GPU). Each G-set problem is solved by PTPA with $N_{swap} = 800$ and $L_{MC} = 250$ (i.e., 1ms solution time), which is 3.2× and 20,600× faster than [6] (3.2ms) and [10] (20.6s), respectively. Compared to the best known cut, this work achieves an average cut accuracy of 99.43%, which is 0.36% and 0.93% higher than [6] and [10], respectively. In addition, an satisfactory average cut accuracy (99.19%) is also available with using $N_{swap} = 400$ and $L_{MC} = 250$ (i.e, 0.5ms).

Table II presents a comprehensive comparison between PTPA and several previous works. PTPA demonstrates a considerable solution speed. Compared to most Ising model-based processing architectures, while maintaining lower power consumption for such a large-scale circuit architecture.

VI. CONCLUSION

In this paper, we propose a novel design of the parallel tempering processing architecture, PTPA, based on the fully-connected Ising model, aiming at addressing COPs. PTPA significantly improves the speed and accuracy of the solution through concurrent updates of multiple spins per replica, as well as an efficient temperature exchange strategy. Experimental results indicate that, the modified PT algorithm achieves 5,160× speedup for solving the max-cut problem through PTPA compared to the PT algorithm implemented on CPU. Furthermore, the proposed PTPA achieves an average cut accuracy of 99.43% for G-set problems compared with recent related works, which demonstrates the effectiveness of the design.

REFERENCES

[1] L. Andrew, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, p. 5, 2013.

[2] I. L. Markov, "Know your limits (review of "limits to parallel computation: p-completeness theory"; greenlaw, r., et al; 1995) [book review]," *IEEE Design Test*, vol. 30, no. 1, pp. 78–83, 2013.

[3] M. Yamaoka *et al.*, "A 20k-spin ising chip to solve combinatorial optimization problems with CMOS annealing," *IEEE Journal of Solid-State Circuits*, vol. 51, no. 1, pp. 303–309, 2016.

[4] T. Takemoto, M. Hayashi, C. Yoshimura, and M. Yamaoka, "A 2 × 30k-spin multi-chip scalable CMOS annealing processor based on a processing-in-memory approach for solving large-scale combinatorial optimization problems," *IEEE Journal of Solid-State Circuits*, vol. 55, no. 1, pp. 145–156, 2020.

[5] S. Tsukamoto, M. Takatsu, S. Matsubara, and H. Tamura, "An accelerator architecture for combinatorial optimization problems," *Fujitsu scientific technical journal*, vol. 53, no. 5, pp. 8–13, 2017.

[6] Z. Huang, D. Jiang, X. Wang, and E. Yao, "An ising model based annealing processor with 1024 fully-connected spins for combinatorial optimization problems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, pp. 1–1, 2023.

[7] D. Jiang, X. Wang, Z. Huang, Y. Yang, and E. Yao, "A network-on-chip-based annealing processing architecture for large-scale fully connected ising model," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 7, pp. 2868–2880, 2023.

[8] K. Yamamoto *et al.*, "STATICA: A 512-spin 0.25m-weight annealing processor with an all-spin-updates-at-once architecture for combinatorial optimization with complete spin–spin interactions," *IEEE Journal of Solid-State Circuits*, vol. 56, no. 1, pp. 165–178, 2021.

[9] H. Goto, K. Tatsumura, and A. R. Dixon, "Combinatorial optimization by simulating adiabatic bifurcations in nonlinear hamiltonian systems," *Science Advances*, vol. 5, APR 2019.

[10] H. M. Waidyasooriya and M. Hariyama, "Temporal and spatial parallel processing of simulated quantum annealing on a multicore CPU," *Journal of Supercomputing*, vol. 78, pp. 8733–8750, APR 2022.

[11] R. Iimura, S. Kitamura, and T. Kawahara, "Annealing processing architecture of 28-nm CMOS chip for ising model with 512 fully connected spins," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 12, pp. 5061–5071, 2021.

[12] J. Zhang, S. Chen, and Y. Wang, "Advancing CMOS-type ising arithmetic unit into the domain of real-world applications," *IEEE Transactions on Computers*, vol. 67, no. 5, pp. 604–616, 2018.

[13] E. Marinari and G. Parisi, "Simulated tempering: a new monte carlo scheme," *Europhysics Letters*, vol. 19, pp. 451–8, 15 July 1992.

[14] N. Metropolis, "Equation of state calculations by fast computing machines," *Journal of Chemical Physics*, vol. 21, 2004.

[15] D. Frenkel and B. Smit, *Understanding molecular simulation : from algorithms to applications. 2nd ed.* Understanding Molecular Simulation: From Algorithms to Applications, 1996.

[16] R. H. Swendsen and J.-S. Wang, "Replica monte carlo simulation of spin-glasses," *Phys. Rev. Lett.*, vol. 57, pp. 2607–2609, Nov 1986.

[17] [Online]. Available:https://web.stanford.edu/∼yyye/yyye/Gset/.

[18] H. M. Waidyasooriya and M. Hariyama, "Highly-parallel FPGA accelerator for simulated quantum annealing," *IEEE Transactions on Emerging Topics in Computing*, vol. 9, no. 4, pp. 2019–2029, 2021.