Solving Traveling Salesman Problems via a Parallel Fully Connected Ising Machine

Qichao Tao
Department of Electrical & Computer Engineering
University of Alberta
Edmonton, Canada
qichao@ualberta.ca

Jie Han
Department of Electrical & Computer Engineering
University of Alberta
Edmonton, Canada
jhan8@ualberta.ca

ABSTRACT

Annealing-based Ising machines have shown promising results in solving combinatorial optimization problems. As a typical class of these problems, however, traveling salesman problems (TSPs) are very challenging to solve due to the constraints imposed on the solution. This article proposes a parallel annealing algorithm for a fully connected Ising machine that significantly improves the accuracy and performance in solving constrained combinatorial optimization problems such as the TSP. Unlike previous parallel annealing algorithms, this improved parallel annealing (IPA) algorithm efficiently solves TSPs using an exponential temperature function with a dynamic offset. Compared with digital annealing (DA) and momentum annealing (MA), the IPA reduces the run time by 44.4 times and 19.9 times for a 14-city TSP, respectively. Large scale TSPs can be more efficiently solved by taking a k-medoids clustering approach that decreases the average travel distance of a 22-city TSP by 51.8% compared with DA and by 42.0% compared with MA. This approach groups neighboring cities into clusters to form a reduced TSP, which is then solved in a hierarchical manner by using the IPA algorithm.

KEYWORDS

Ising machine, combinatorial optimization, traveling salesman problem, simulated annealing, fully connected Ising model

ACM Reference Format:
Qichao Tao and Jie Han. 2022. Solving Traveling Salesman Problems via a Parallel Fully Connected Ising Machine. In *San Francisco '59: ACM/IEEE Design Automation Conference (DAC), Jul 11, 2022– Jul 15, 2022, San Francisco, CA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Combinatorial optimization problems exist in many applications, such as drug discovery, Internet of Things technology, and machine learning [1]. However, such a problem is non-deterministic polynomial time (NP)-hard and time-consuming to solve using the enumeration method on a conventional computer [2]. For example, $(M-1)!$ possibilities need to be traversed to solve a traveling salesman problem (TSP) with M cities. The traversal time will substantially increase as M becomes large, making solving TSPs very difficult. In fact, an approximation method is often used to obtain a suboptimal or good enough solution in many industrial applications. Recently, an efficient approximate system called an Ising machine has been considered for solving combinatorial optimization problems [3].

An Ising model mathematically describes the ferromagnetic interactions of magnetic spins. In the classical 2D Ising model, a spin is considered to only interact with the closest neighbor spins on a square lattice, as shown in Fig. 1(a). An Ising machine solves a combinatorial optimization problem by mapping it to the Ising model and then searching for the ground state via annealing with random flips [4], as shown in Fig. 2. The energy of the Ising model tends to converge to a minimum value during the annealing process; the random flip is applied to avoid being stuck in a local minimum. An Ising machine does not traverse all possibilities and thus is more efficient than conventional computers for solving large-scale combinatorial optimization problems. Various annealing-based Ising machines have been studied, including the quantum Ising machine [5] and the coherent Ising machine [6] based on quantum mechanics and optical parametric oscillators, respectively. In this paper, we focus on CMOS Ising machines as CMOS circuits are easy to manufacture and scale [4].

Simulated annealing (SA) [7] serves as the basis of various annealing algorithms, such as CMOS annealing [4, 8], digital annealing (DA) [9, 10], momentum annealing (MA) [11], and stochastic cellular automata annealing (SCA) [1]. SA mimics the thermal annealing in metallurgy, whereas CMOS annealing implements SA on CMOS circuits and the random flip is realized by utilizing the variability of static random access memory (SRAM) cells under a low supply voltage [4, 8]. However, CMOS annealing is implemented on a sparse spin-to-spin structure, which is inefficient in solving complex combinatorial optimization problems. Hence, DA has been developed to implement a fully connected Ising model, as shown in Fig. 1(b). DA further speeds up the annealing process by a parallel search and using an escape mechanism referred to as a dynamic offset [9, 10]. Nevertheless, only one spin can be flipped per iteration in the DA because connected spins cannot be simultaneously updated in the Ising model. This increases the average time required for the Ising machine to find a solution. In order to perform a parallel spin-update in the fully connected Ising model and accelerate the annealing process, a two-layer spin structure with self-interactions has been proposed in [1, 11], as shown in Fig. 1(c), and referred to

DAC '59, Jul 11, 2022– Jul 15, 2022, San Francisco, CA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/1122445.1122456

as SCA and MA. They belong to the class of parallel annealing (PA) algorithms, and the Ising machines that use PA are called parallel fully connected Ising machines.

Constrained combinatorial optimization problems, including TSPs, are more challenging than unconstrained combinatorial optimization problems (such as the max-cut problem). Although DA can solve TSPs, only sequential spin-update is possible, which makes the annealing speed a performance bottleneck. Both SCA and MA can perform parallel spin-update; however, they have only been considered to solve the max-cut problem.

This paper proposes an improved parallel annealing (IPA) algorithm to solve constrained combinatorial optimization problems such as the TSP using parallel fully connected Ising machines. To the best of the authors' knowledge, this work is the first attempt to do so. The contributions lie in the following novelties aimed at improving the performance of fully connected Ising machines in solving TSPs: (1) using an exponential temperature function with a dynamic offset and (2) using k-medoids in Ising model-based TSP solvers to preprocess data for improving the quality of solutions.

The remainder of this paper is organized as follows. Section 2 reviews the background of parallel annealing algorithms and TSP mapping. The IPA algorithm and the k-medoids are discussed in Section 3 for solving the TSPs. Section 4 reports the experimental results on TSP benchmarks. Section 5 concludes the paper.

2 PRELIMINARIES

2.1 The Ising model

In the Ising model, each spin can be in either an upward (+1) or downward (−1) state. The interactions among the spins and external magnetic fields affect the states of spins. In an N-spin system, the Hamiltonian in an Ising model is defined as [12]:

$$H(\sigma_1, ..., \sigma_N) = -\sum_{i,j} J_{ij}\sigma_i\sigma_j - \sum_i h_i\sigma_i, \tag{1}$$

where σ_i ($\in \{-1, +1\}$, $i \in \{1, 2, ..., N\}$) denotes the state of the ith spin, J_{ij} indicates the interaction between the ith spin and the jth spin, and h_i is the external magnetic field for the ith spin.

2.2 Mapping the Traveling Salesman Problem

An n-city TSP can be expressed in Hamiltonian as [3]:

$$\begin{aligned} H_{TSP} &= A\sum_{k\neq l}\sum_i W_{kl}a_{ik}a_{(i+1)l} + B\sum_i\Big(\sum_k a_{ik} - 1\Big)^2 \\ &\quad + C\sum_k\Big(\sum_i a_{ik} - 1\Big)^2, \end{aligned} \tag{2}$$

where a_{ik} ($\in \{0, +1\}$) indicates whether the kth city is visited (+1) or not (0) at the ith step, and W_{kl} denotes the distance between the kth city and the lth city. The first term in (2) is the objective function of the TSP, which computes the total distance of the route. The second and the third terms in (2) are constraints that prevent visiting multiple cities in one step and visiting a city more than once, respectively. These two terms take the minimum value 0 when $\sum_k a_{ik} = \sum_i a_{ik} = 1$. A, B and C are the parameters (with positive values) that balance the weights between the objective function and the constraints.

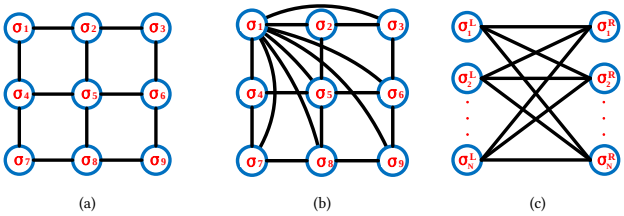

Figure 1: (a) A classical 2D Ising model, (b) a fully connected Ising model (illustrated using the spin in the top left corner), and (c) a two-layer spin structure for the Ising model [1, 11].

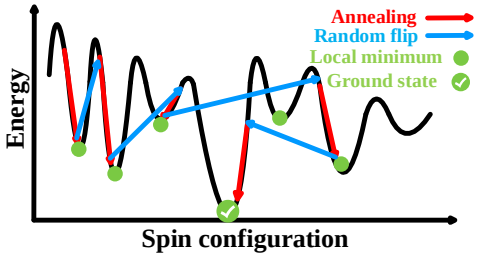

Figure 2: The energy profile of the Ising model during annealing with random flips.

Then the TSP can be mapped to the Ising model by converting a_{ik}($\in \{0, +1\}$) to σ_{ik}($\in \{-1, +1\}$) as follows [3]:

$$\begin{aligned} H_{TSP} &= \frac{A}{4}\sum_{k\neq l}\sum_i W_{kl}\sigma_{ik}\sigma_{(i+1)l} + \frac{A}{2}\sum_{k\neq l}\sum_i W_{kl}\sigma_{ik} \\ &\quad + \frac{B}{4}\sum_i\sum_k\sum_l \sigma_{ik}\sigma_{il} + \frac{(n-2)B}{2}\sum_i\sum_k \sigma_{ik} \\ &\quad + \frac{C}{4}\sum_i\sum_k\sum_j \sigma_{ik}\sigma_{jk} + \frac{(n-2)C}{2}\sum_i\sum_k \sigma_{ik} \\ &\quad + \frac{A}{4}\sum_{k\neq l}\sum_i W_{kl} + \Big(\frac{n^3}{4} - n^2 + n\Big)(B + C). \end{aligned} \tag{3}$$

The last two constant terms in (3) unrelated to the states of spins are ignored when minimizing H_{TSP}. The first, third, and fifth terms correspond to the interaction term in (1), and the other terms correspond to the external field term in (1).

2.3 Parallel Annealing (PA)

In the two-layer spin structure for the Ising model, the couplings between σ_i^L and σ_j^R are denoted as $J_{ij}(i \neq j)$, whereas the couplings between σ_i^L and σ_i^R are called self-interactions (denoted as ω_i). Thus, the Hamiltonian based on PA, H_P, is given by [1, 11]:

$$H_P = -\sum_{i,j} J_{ij}\sigma_i^L\sigma_j^R - \frac{1}{2}\sum_i h_i(\sigma_i^L + \sigma_i^R) + \omega_i\sum_i(1 - \sigma_i^L\sigma_i^R). \tag{4}$$

Only when the self-interactions ω_i are sufficiently large, are the spin configurations in both layers the same, i.e., $\sigma_i^R = \sigma_i^L$. Thus, the third term in (4) can be eliminated, so (4) becomes the same as (1)

[1, 11]. ω_i is given by [11]:

$$\omega_i = \begin{cases} \sum_{s_j \in S} |J_{ij}| - \frac{1}{2} \sum_{s_j \in C} |J_{ij}| & (s_i \in C) \\ \frac{\lambda}{2} & (s_i \notin C) \end{cases}, \quad (5)$$

where λ is the largest eigenvalue of $-\boldsymbol{J}$ ($\boldsymbol{J}$ is a matrix of J_{ij}), s_i is the ith spin, C is a subset of the set of all spins S and C satisfies $C = \{s_i | \lambda \geq \sum_{s_j \in S} |J_{ij}|\}$.

The spin-flip probability is calculated using the Metropolis algorithm [13]. If σ_i^L is flipped, the total energy will be increased by

$$\Delta_i = 2\sigma_i^L (\frac{h_i}{2} + \sum_j J_{ij}\sigma_j^R + \omega_i \sigma_i^R). \quad (6)$$

Then, the new spin-flip probability is $\min\{1, exp(-\Delta_i/T)\}$, where T is the temperature.

To improve the efficiency of annealing, dropout and momentum scaling are introduced in [11]. The dropout sets each ω_i as "0" with a decreasing probability. The momentum scaling multiplies every interaction ω_i by an increasing factor from 0 to 1. Thus, at the end of momentum annealing, every interaction ω_i will return to the value computed in (5) to ensure $\sigma_i^R = \sigma_i^L$ in (4).

3 IMPROVED PARALLEL ANNEALING FOR TSPS

3.1 Improved Parallel Annealing (IPA)

The IPA for solving TSPs is shown in Algorithm 1. An exponential temperature function is used in the IPA and a dynamic offset is applied to the temperature function. Firstly, the state of a spin is randomly initialized to " − 1" or " + 1", and the temperature increment (ΔT) due to the dynamic offset is initialized to "0". In Fig. 1(c), spins in the left layer are updated when the current step s is odd; otherwise, the spins in the right layer are updated. In each iteration ($s \in [1, iternum]$), the dropout rate (p_s) and the momentum scaling factor (c_s) are updated, where the $iternum$ is the total number of iterations. The temperature (T_s) is then recalculated, where r in Algorithm 1 is the cooling coefficient. During the annealing, the self-interaction (ω_{ik}) is set to "0" with the probability p_s or decreased to $c_s \cdot \omega_{ik}$. Then, the energy variation (Δ_{ik}) when σ_{ik} is flipped is evaluated using the spin interaction (J_{ikjl}) and the updated ω_{ik}. Subsequently, the spin-flip probability (P_{ik}) is calculated using the Metropolis algorithm. If P_{ik} is larger than a randomly generated number within $(0, 1)$, the spin will be flipped. Otherwise, the spin will remain unchanged. After each iteration, if no spin is flipped, ΔT will increase. Otherwise, ΔT will be reset to "0". Finally, the spin configuration (σ) at the end of iterations is output as the solution to the combinatorial optimization problem found by the IPA.

3.2 A Temperature Function

The classical annealing with parallel spin-update uses a logarithmic function in (7) as the temperature function to solve the max-cut problem [11]:

$$T_s = \frac{1}{\beta_0 \ln(1+s)}, \quad (7)$$

Algorithm 1 Improved Parallel Annealing for TSPs

Input: spin interaction: $\boldsymbol{J}$; external magnetic field: $\boldsymbol{h}$; the number of cities: M; self-interaction: $\boldsymbol{\omega}$; hyperparameters: $iternum$, T_{init}, T_{inc}, r
Output: spin configuration (σ)

1: Initialize spin configurations
2: $T_s \Leftarrow T_{init}$
3: $\Delta T \Leftarrow 0$
4: **for** $s = 1$ to $iternum$ **do**
5: **if** s is odd **then**
6: $A \Leftarrow L, B \Leftarrow R$
7: **else**
8: $A \Leftarrow R, B \Leftarrow L$
9: **end if**
10: Update p_s and c_s
11: $T_s \Leftarrow (T_s + \Delta T) \cdot r^{s-1}$
12: **for** $i = 1$ to M **do**
13: **for** $k = 1$ to M **do**
14: Temporarily set $\omega_{ik} \Leftarrow 0$ with the probability p_s, and temporarily decrease $\omega_{ik} \Leftarrow c_s \cdot \omega_{ik}$
15: $\Delta_{ik} \Leftarrow 2\sigma_{ik}^A (\frac{h_{ik}}{2} + \sum_{j,l} J_{ikjl}\sigma_{jl}^B + \omega_{ik}\sigma_{ik}^B)$
16: $P_{ik} \Leftarrow \min\{1, exp(-\Delta_{ik}/T_s)\}$
17: **if** $P_{ik} > rand$ **then**
18: $\sigma_{ik}^A \Leftarrow -\sigma_{ik}^A$
19: **end if**
20: **end for**
21: **end for**
22: **if** no spin is flipped **then**
23: $\Delta T \Leftarrow \Delta T + T_{inc}$
24: **else**
25: $\Delta T \Leftarrow 0$
26: **end if**
27: **end for**

where β_0 is a scaling factor for the inverted value of temperature and T_s denotes the temperature in the sth iteration. When the Ising model reaches a local minimum or ground state and T_s is sufficiently small, the flip probability for each spin, P_{ik}, is considered to be close to 0 (see lines 15 and 16 in Algorithm 1). With a proper β_0, however, the temperature only decreases to a value that results in low flip probabilities for all spins. Hence, it is possible for the Ising model to escape from local minima when solving max-cut problems.

To solve a TSP, the temperature required for maintaining low spin-flip probabilities is larger because Δ_{ik} is larger due to the constraints. However, an Ising model cannot reach a local minimum or ground state, or meet the constraints at such a temperature. Thus, the temperature needs to be sufficiently low at the end of annealing when solving TSPs. It will, therefore, be difficult for the Ising model to escape from a local minimum. Moreover, the temperature using a logarithmic function rapidly decreases, so it will prevent the Ising model from traversing additional local minima, thereby reducing the quality of solutions. Hence, an exponential function is used as the temperature function to solve the TSP, as

$$T_s = T_{init} \cdot r^{s-1}, \quad (8)$$

where T_{init} is the initial temperature and r is the cooling rate. The slower decreasing rate of the exponential function makes the Ising model stay longer at a high temperature, therefore improving the quality of solutions.

Considering that the number of local minima increases with the TSP scale, the Ising model is prone to be stuck in a local minimum during annealing. Reducing the time spent in a local minimum can improve efficiency. Thus, we consider introducing a dynamic offset, as in [9, 10, 14], into the temperature function. To increase the probability of escaping from a local minimum, the temperature T_s needs to be very large. Therefore, ΔT is added to T_s, where ΔT is increased by T_{inc} if the spin configuration is unchanged. Lastly, ΔT is reset to zero after a change of the spin state has occurred.

The improvement of the solution quality after using an exponential function with a dynamic offset and the details for setting a proper T_{inc} are discussed in Section 4.

3.3 A Clustering Approach

The solution quality drastically deteriorates when the number of cities in the TSP is large. We further consider a clustering approach [3] to improve the quality of the solution. The basic idea is to group the nearby cities into one cluster and use the centric point to represent each cluster. Then the TSP consisting of those central points can be solved by using the IPA. After learning the visiting order of each cluster, the original TSP can be more efficiently solved. This Ising machine can avoid producing solutions that conform to the constraints but with very long travel distances. For example, if cluster A contains three cities and is first visited among the clusters, then the visiting order of these three cities will be confined to the first three steps.

The k-medoids and k-means are two typical clustering approaches. To divide M vertices into k clusters, the first step of the k-means is to randomly generate k new vertices as the central points of the k clusters, while the k-medoids method chooses k vertices from the original set as the central points. In the second step, after the k central points are obtained, the other $(M-k)$ vertices in the set are assigned to the closest central point and form a cluster. In the third step, the k-means method generates a new central point for each cluster according to the mean value of the coordinates of the vertices in the cluster, while the k-medoids method chooses the vertex with the smallest sum of distances from the other vertices in the same cluster as the new central point. Then, the second and the third steps are repeated until there is no change in any cluster.

Compared with the k-means, the disadvantage of k-medoids is the computation time for each cluster in the third step, $O(m^2)$, where m is the number of vertices in one cluster. However, there are $M \times M$ accumulators in the circuit of an Ising model for $M \times M$ spins that solves an M-city TSP, where $M = \sum_{i=1}^{k} m_i$ and m_i is the number of vertices in the ith cluster. Thus, this computation time can be reduced to $O(m)$ as it can be calculated in parallel with m accumulators. Furthermore, calculation of the distances between the vertices is not required in an Ising machine as the distance values are included in the system's input, i.e., in the spin interaction matrix $\boldsymbol{J}$. In contrast, the k-means method needs extra arithmetic units to compute the distances between the vertices and the central points. Therefore, using k-medoids for clustering can achieve a higher hardware efficiency than using k-means with no performance trade-off in an Ising machine.

Algorithm 2 The k-medoids clustering

Input: distance matrix: $\boldsymbol{W}(M \times M)$; the number of clusters: k
Output: vertex indexes (with cluster labels)
Step 1
for $i = 1$ to M **do**
  $D_i = \sum_{j=1}^{M} W_{ij}$
end for
Choose k vertices (v) with the first k smallest D as the central points
Step 2
for $i = 1$ to M **do**
  Assign v_i to the closest central point and mark the label of v_i with the index of the corresponding central point
end for
Step 3
for each cluster **do**
  for each v_i in the cluster **do**
    $d_i = \sum_j W_{ij}$
  end for
  Choose v with the smallest d to be the new central point of the current cluster
end for
Repeat Step 2 and Step 3 until there is no change of elements in each cluster

The k-medoids clustering was proposed in [15], but it is first applied in an Ising model-based TSP solver in this work, as shown in Algorithm 2. A strategy for choosing the k vertices at the center of the map as initial central points is applied to improve the efficiency of k-medoids. The k vertices with the k smallest sums of distances from all the other vertices are selected as initial central points.

To implement the visiting restrictions, an M-by-M matrix $\boldsymbol{h_p}$ is added to the external magnetic field matrix, $\boldsymbol{h}$. For example, if the first three cities are confined to be visited at the first three steps, $\boldsymbol{h_p} = \begin{pmatrix} \boldsymbol{a} & \boldsymbol{b} \\ \boldsymbol{c} & \boldsymbol{d} \end{pmatrix}$, where $\boldsymbol{a}$ and $\boldsymbol{d}$ are 3-by-3 and $(M-3)$-by-$(M-3)$ zero matrices, respectively; $\boldsymbol{b}$ and $\boldsymbol{c}$ are 3-by-$(M-3)$ and $(M-3)$-by-3 matrices of $M \cdot max\{abs(\boldsymbol{J})\}$ (as a large value), respectively. The $max\{abs(\boldsymbol{J})\}$ returns the entry in matrix $\boldsymbol{J}$ with the largest absolute value.

4 EXPERIMENTAL RESULTS

Seven benchmark datasets are used in the experiments, including *burma*14, *ulysses*16, *ulysses*22, and four other sub-datasets randomly selected from the TSPLIB benchmark ($n = 6$ and 12 from *gr*431 and $n = 7$ and 10 from *ali*535). The average (Ave), maximum (Max), minimum (Min), and standard deviation (Std) of the travel distances are obtained after performing annealing by 100 times with an iteration number of $10k$. The simulation was run in MATLAB on an AMD processor Ryzen 5 3600X (3.8 GHz).

We evaluate the effect of the penalty parameters B and C on the performance of the IPA with six benchmark datasets, including *burma*14, *ulysses*16, $n = 6$ and 12 from *gr*431 and $n = 7$ and 10 from

*ali*535. The results are obtained with $T_{inc} = max\{abs(\boldsymbol{J})\}$, $r = 0.97$, and $T_{init} = 1 \times 10^7$. Here, T_{init} is an arbitrarily large value and r is selected to ensure $T_{inc} \cdot r^{s-1}$ close to 0 at the end of annealing. As shown in Fig. 3, for all six TSPs, more stable solutions with smaller Ave can be obtained when the penalty parameters decrease. It is due to the fact that for lower penalty parameter values, the Ising model escapes from the local minima with a higher probability. However, the parameter values must be large enough to ensure all solutions meet the constraints, which are violated when B and C are smaller than $0.85 \times \max\{\boldsymbol{W}\}$. The improvement in solution quality is less significant when B and C are smaller than $1 \times \max\{\boldsymbol{W}\}$. Therefore, we choose $B = C = 1 \times \max\{\boldsymbol{W}\}$ as the penalty parameter setting in our further experiments.

As a key to increasing the probability to escape from a local minimum, an appropriate setting of T_{inc} can optimize the efficiency of an Ising model. Therefore, we investigate the effect of different T_{inc} values on the Ave and Std of solutions found by the IPA. The results for the three benchmarks, *burma*14, *ulysses*16, and *ulysses*22, are obtained with penalty parameters $B = C = 1 \times \max\{\boldsymbol{W}\}$, $r = 0.97$, and $T_{init} = 1 \times 10^7$. As shown in Fig. 4, the Ave, Max, and Std of the travel distances found by the Ising models decrease when T_{inc} decreases from $10 \times max\{abs(\boldsymbol{J})\}$ to $max\{abs(\boldsymbol{J})\}/10$. Furthermore, the Ave, and Max tend to be stable when T_{inc} is between $max\{abs(\boldsymbol{J})\}/10$ and $\frac{max\{abs(\boldsymbol{J})\}}{90}$. The further decrease of T_{inc} results in an increase of Ave, so it degrades the quality of solutions. A small T_{inc} reduces the chance of the Ising models escaping from local minima, and thus a larger iteration number is required for finding a suboptimal solution.

To evaluate the performance of the proposed methods, MA [11] and DA [14] are considered for comparison. The MA implements parallel spin-update but using a logarithmic temperature function, while the DA employs a dynamic offset but without parallel spin-update. Three benchmarks, *burma*14, *ulysses*16, and *ulysses*22, are used. The results are obtained with $r = 0.97$, $T_{inc} = \frac{max\{|\boldsymbol{J}|\}}{90}$ and

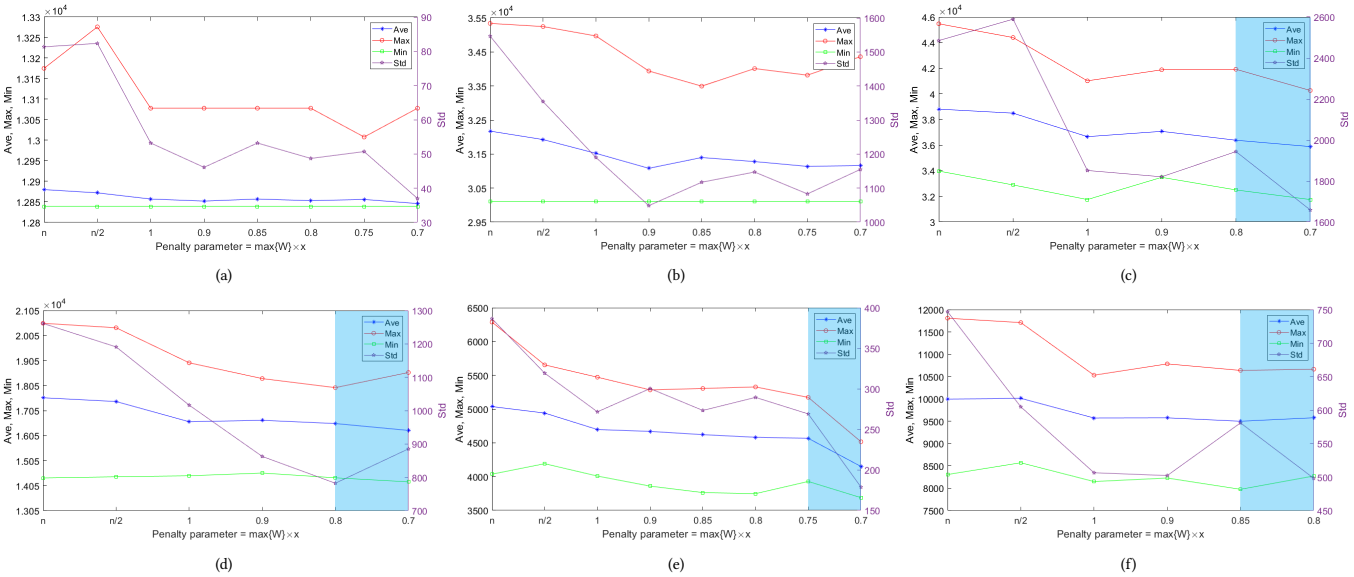

Figure 3: The effect of the penalty parameters B and C ($B = C$) on the quality of solutions: (a) for $n = 6$ from *gr*431, (b) for $n = 7$ from *ali*535, (c) for $n = 10$ from *ali*535, (d) for $n = 12$ from *gr*431, (e) for the benchmark *burma*14, and (f) for the benchmark *ulysses*16. The blue shadow area indicates the results that do not meet constraints.

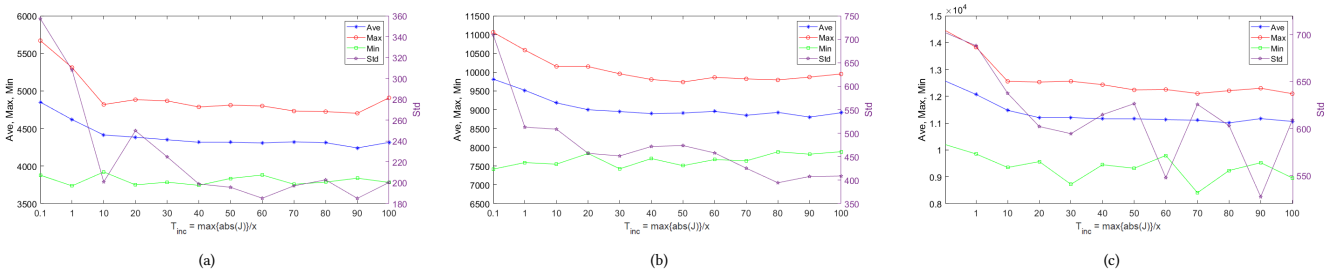

Figure 4: The effect of T_{inc} on the quality of solutions: (a) for the benchmark *burma*14, (b) for the benchmark *ulysses*16, and (c) for the benchmark *ulysses*22.

Table 1: (Unitless) Travel Distances by using the IPA, MA, and DA for Solving the TSP

Metrics	$iternum = 10k$			$iternum = 50k$		
	IPA	MA	DA	IPA	MA	DA
burma14						
Ave	4241.6	5322.4	8832.9	4018.5	5133.3	6451.8
Max	4703.0	7524.0	9655.0	4423.0	7443.0	8009.0
Min	3839.0	4178.0	7507.0	3580.0	4099.0	4945.0
Std	185.1	683.7	379.8	159.9	547.7	696.4
ulysses16						
Ave	8804.2	11513.0	12722.0	8387.6	11451.0	12040.0
Max	9869.0	13992.0	15454.0	9218.0	14366.0	14669.0
Min	7816.0	8859.0	9827.0	7554.0	9242.0	8815.0
Std	407.9	1113.6	1141.5	303.3	1057.4	1240.4
ulysses22						
Ave	11170.0	13811.0	16619.0	10389.0	13367.0	16435.0
Max	12301.0	18914.0	20316.0	11167.0	16799.0	18862.0
Min	9527.0	11154.0	13224.0	9163.0	9363.0	13000.0
Std	527.3	1622.3	1425.6	433.7	1284.8	1156.4

Table 2: (Unitless) Travel Distances by Using the k-Medoids Clustering in the IPA for Solving the TSP

Metrics	burma14 ($k_1 = 7, k_2 = 4$)	ulysses16 ($k_1 = 8, k_2 = 4$)	ulysses22 ($k_1 = 10, k_2 = 6$)
Ave	3813.8	7705.0	8011.4
Max	4334.0	8923.0	9371.0
Min	3345.0	6686.0	7219.0
Std	268.9	440.9	433.2

$T_{init} = 1 \times 10^7$ for the IPA and DA. For the MA, $\beta_0 = 9 \times 10^{-4}$ for *burma*14, $\beta_0 = 8 \times 10^{-4}$ for *ulysses*16, and $\beta_0 = 5 \times 10^{-4}$ for *ulysses*22. These β_0 values are chosen to produce the best solution quality in the experiment. The penalty parameters in (3) are set as $A = 1$, $B = C = 1 \times \max\{\boldsymbol{W}\}$. Table 1 shows the performance of the IPA, MA, and DA for solving the TSP. The algorithms with parallel spin-update (IPA and MA) obtain lower Ave than DA for all three benchmarks. However, the Ave obtained by MA can hardly be improved by increasing the number of iterations. This occurs because the annealing algorithm using a logarithmic temperature function is easily stuck in a local minimum when solving TSPs. On the contrary, the algorithms that employ a dynamic offset, such as the IPA and DA, can find shorter distances when the iteration number increases. For solving *burma*14, when the iteration number is $10k$, the IPA decreases the Ave by 52.0% compared with DA and by 20.3% compared with MA. The required iterations for the DA, MA, and IPA to produce an Ave around 4920 are $250k$, $20k$, and $1k$, respectively, while the runtimes are 4.44 seconds, 1.99 seconds, and 0.10 seconds, respectively. The IPA achieves a 44.4× speed-up in runtime compared with DA, and 19.9× compared with MA.

Further decrease in Ave is obtained by using the clustering approach. We applied k-medoids twice for each benchmark. The original TSP is clustered into a second-level TSP with k_1 centric points, and the second-level TSP is clustered into a third-level TSP with k_2 centric points. The iteration number for solving the third-level TSP is 1000; it is 2500 for the second-level TSP and 3000 for the original TSP, so the total iteration number is 6500. As shown in Table 2, the reduction in Ave is 51.8% or 42.0% compared to DA or MA for *ulysses*22 ($iternum = 10k$), 39.4% or 33.1% for *ulysses*16, and 56.8% or 28.3% for *burma*14, respectively.

5 CONCLUSION

This paper is the first to present an efficient solution for constrained combinatorial optimization problems such as the TSP using parallel fully connected Ising machines. Specifically, an improved parallel annealing (IPA) algorithm is proposed to leverage an exponential temperature function with a dynamic offset and a k-medoids clustering approach. The exponential temperature function with a dynamic offset can alleviate the problem of being stuck in local minima, while the k-medoids clustering significantly reduces the average travel distance. The IPA is at least an order of magnitude faster than DA and MA to find a similar average travel distance. A shorter average travel distance can further be found by the IPA with a smaller number of iterations due to the use of k-medoids clustering. These results pave the way for the development of energy-efficient circuit architectures for solving combinatorial optimization problems using the Ising model. The hardware design of a parallel fully connected Ising machine will be carried out in future work.

REFERENCES

[1] K. Yamamoto et al., "STATICA: a 512-Spin 0.25 m-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, 2020.

[2] K. Someya, R. Ono, and T. Kawahara, "Novel Ising model using dimension-control for high-speed solver for Ising machines," NEWCAS, Vancouver, BC, Canada, pp. 1-4, 2016.

[3] A. Dan, R. Shimizu, T. Nishikawa, S. Bian, and T. Sato, "Clustering approach for solving traveling salesman problems via Ising model based solver," the 57th ACM/IEEE Design Automation Conference (DAC), pp. 1-6, 2020.

[4] M. Yamaoka, C. Yoshimura, M. Hayashi, T. Okuyama, H. Aoki, and H, Mizuno, "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, 2015.

[5] M. W. Johnson et al., "Quantum annealing with manufactured spins," Nature, vol. 473, no. 7346, pp. 194-198, 2011.

[6] R. Hamerly et al., "Experimental investigation of performance differences between Coherent Ising Machines and a quantum annealer," Science Advances, vol. 5, no. 5, p. eaau0823, 2019.

[7] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," Science, vol. 220, no. 4598, pp. 671-680, 1983.

[8] 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, 2019.

[9] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura, and H. G. Katzgraber, "Physics-inspired optimization for quadratic unconstrained problems using a digital annealer," Frontiers in Physics, vol. 7, p. 48, 2019.

[10] S. Matsubara et al., "Digital annealer for high-speed solving of combinatorial optimization problems and its applications," the 25th Asia and South Pacific Design Automation Conference (ASP-DAC), pp. 667-672, IEEE, 2020.

[11] T. Okuyama, T. Sonobe, K. Kawarabayashi, M. Yamaoka, "Binary optimization by momentum annealing," Physical Review E, vol. 100, no. 1, p. 012111, 2019.

[12] A. Lucas, "Ising formulations of many NP problems," Frontiers in physics, vol. 5, no. 5, p. 5, 2014.

[13] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and H. Teller, "Equation of state calculations by fast computing machines," The Journal of Chemical Physics, vol. 21, no. 6, pp. 1087-1092, 1953.

[14] S. Tsukamoto, M. Takatsu, S. Matsubara, and H. Tamura, "An accelerator architecture for combinatorial optimization problems," Fujitsu Sci. Tech. J, vol. 53, no. 5, pp. 8-13, 2017.

[15] H. S. Park, and C. H. Jun, "A simple and fast algorithm for K-medoids clustering," Expert Systems with Applications, vol. 36, no. 2, pp. 3336-3341, 2009.