Self-Adaptive Ising Machines for Constrained Optimization

Corentin Delacour
Department of Electrical and Computer Engineering
University of California Santa Barbara
delacour@ucsb.edu

***Abstract*—Ising machines (IM) are physics-inspired alternatives to von Neumann architectures for solving hard optimization tasks. By mapping binary variables to coupled Ising spins, IMs can naturally solve unconstrained combinatorial optimization problems such as finding maximum cuts in graphs. However, despite their importance in practical applications, *constrained* problems remain challenging to solve for IMs that require large quadratic energy penalties to ensure the correspondence between energy ground states and constrained optimal solutions. To relax this requirement, we propose a *self-adaptive* IM that iteratively shapes its energy landscape using a Lagrange relaxation of constraints and avoids prior tuning of penalties. Using a probabilistic-bit (p-bit) IM emulated in software, we benchmark our algorithm with multidimensional knapsack problems (MKP) and quadratic knapsack problems (QKP), the latter being an Ising problem with linear constraints. For QKP with 300 variables, the proposed algorithm finds better solutions than state-of-the-art IMs such as Fujitsu's Digital Annealer and requires 7,500x fewer samples. Our results show that adapting the energy landscape during the search can speed up IMs for constrained optimization.**

***Index Terms*—Ising Machines, Constrained Optimization, Lagrange relaxation, Knapsack problems, probabilistic bit**

I. Introduction

Owing to their parallel computing capability, Ising machines (IMs) hold great promises for accelerating challenging optimization problems [1]. Various technologies based on optics [2], [3], memristors [4], [5], coupled oscillators [6]–[8], digital electronics [9], [10] or quantum annealers [11] are currently being explored to implement Ising machines in hardware. The key principle is to harness the dynamics of distributed processing units $S_i = \pm 1$ called *spins*, that *naturally* minimize the Ising Hamiltonian:

$$H = -\sum_{i<j} J_{ij} S_i S_j - \sum_i h_i S_i \tag{1}$$

where J_{ij} and h_i are the couplings and spin fields, respectively. Since the Ising decision problem (does the ground state of H is negative?) is nondeterministic polynomial-time (NP)-complete [12], the hope is that some hard combinatorial optimization problems could be solved more efficiently using Ising machines than classical von Neumann architectures. For instance, minimizing Eq. (1) is equivalent to the NP-hard problem of maximizing the cut of a graph where its vertices correspond to spins and graph edges are weighted by $W_{ij} = -J_{ij}$ [12].

Ising machines have already shown promising results in solving quadratic unconstrained binary optimization problems (QUBO) such as Max-cut [3], [6]. However, solving *constrained* problems with Ising machines has been less studied despite the need to model constraints in various real-life applications [13]. To name a few, constraints on limited resources are

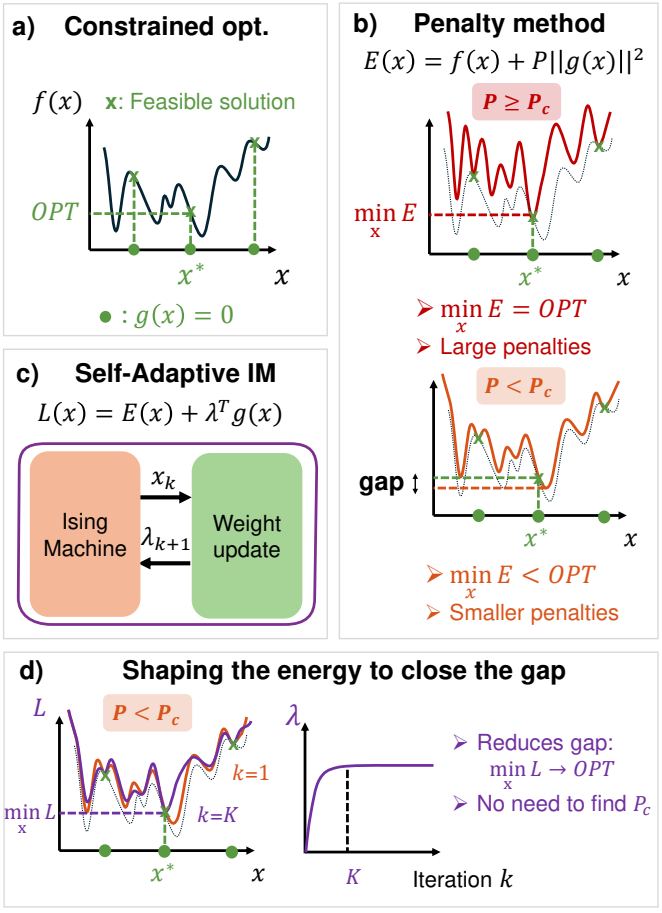


Fig. 1. a) A constrained optimization problem consists of finding a minimum of $f(x)$ among feasible states defined by $g(x) = 0$ (green dots). b) QUBO mapping for Ising machines. The classical penalty method induces positive penalties in the energy landscape. P determines the penalty magnitude and needs to be larger than a critical value P_C to ensure $\min_x E = OPT$. A smaller P favors $f(x)$ but produces a gap $OPT - \min_x E > 0$ where $\min_x E$ is unfeasible. c) To close the gap while keeping reasonable energy penalties, we propose a self-adaptive Ising machine that autonomously adjusts its energy landscape based on a Lagrange relaxation of constraints. d) From an arbitrary $P < P_C$, the algorithm iteratively shapes the energy landscape using measured samples and is compatible with any programmable Ising machine.

978-3-9826741-0-0/DATE25/© 2025 EDAA

found in capital budgeting, portfolio optimization, or production planning [14]. Constraints can also model forbidden paths in vehicle routing problems or impose sequences of operations for job shop scheduling problems [13]. Unfortunately, Ising machines do not naturally support constraints and are principally designed to minimize a cost function mapped to Eq. (1).

The state-of-the-art approach to handle constraints with Ising machines consists of positively penalizing the Ising energy when a state S is not feasible (unsatisfied constraints) and is called *the penalty method* [15]. The method has been applied to quantum annealers for various constrained problems such as finding network shortest paths [16], nurse scheduling [17], job shop scheduling [18], and as well as on digital hardware for the quadratic knapsack problem [19], [20]. Owing to its simplicity, the penalty method is very appealing for Ising machines since it only requires finding suitable Ising parameters J_{ij} and h_i from Eq. (1) before computation and is general for various problems with equality constraints. However, finding the optimal amount of energy penalty or deriving tight bounds is a hard problem that often leads to a tuning phase and worsens the global execution time [12], [19], [21].

In this paper, we propose to relax this requirement by harnessing a Lagrange relaxation of constraints [22] which allows the Ising machine to automatically find the optimal energy penalties in a *self-adaptive* manner. The key idea is to start the self-adaptive Ising machine (SAIM) with some initial energy penalties, and iteratively refine them after each measurement so that the ground state of Eq. (1) tends to a constrained optimal solution. Although it is possible to post-process the Ising machine output and force constraint satisfaction with problem-dependent heuristics [20], our approach rather focuses on finding good Ising parameters J_{ij} and h_i to fully exploit the Ising paradigm with a physical machine.

After describing the classical penalty method, we introduce the SAIM algorithm and benchmark it with two hard-constrained optimization problems: the quadratic knapsack problem (QKP) and the multidimensional knapsack problem (MKP) using a probabilistic-bit (p-bit) Ising machine emulation in software. Our results show that adapting the energy landscape during the Ising machine operation increases the accuracy and reduces the number of samples by at least two orders of magnitude compared to state-of-the-art Ising machines.

II. Constrained Optimization with Ising Machines

We focus on constrained optimization problems with optimal values expressed as:

$$OPT = \min_x f(x) \quad \text{s.t. } g(x) = 0 \tag{2}$$

where $f : x \in \{0;1\}^N \rightarrow \mathbb{R}$ is an objective function to minimize and $g : x \in \{0;1\}^N \rightarrow \mathbb{R}^M$ is a constraint function satisfied when $g(x) = 0$. For a standard Ising machine with quadratic interactions, f is at most quadratic and g is linear. However, one could design a high-order Ising machine supporting higher polynomial degrees for f and g [23].

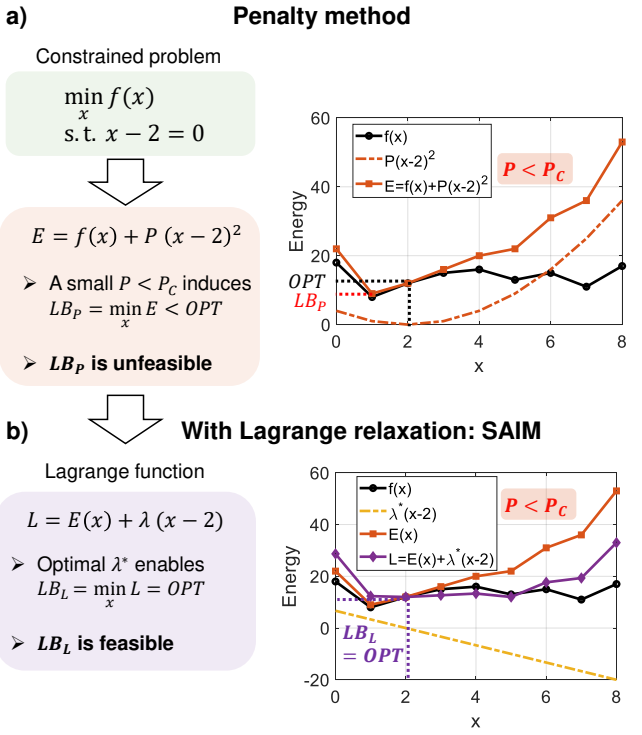


Fig. 2. Illustration of the penalty method and the Lagrange relaxation for a minimization problem subjected to a toy constraint $x = 2$ where P is smaller than the critical value P_C. a) A small $P < P_C$ is insufficient for the penalty method to satisfy $LB_P = OPT$ and the minimization process is likely to provide the unfeasible LB_P. b) Adding a Lagrange relaxation of constraints allows L to close the gap for λ^* where $LB_L = OPT$.

A. Penalty Method

The penalty method adds the constraints weighted by a real parameter $P > 0$ to the objective function as:

$$E = f(x) + P\,||g(x)||^2 \tag{3}$$

The second term *penalizes* the energy E when x lies in an unfeasible region ($g(x) \neq 0$) and is illustrated in Fig. 1b. Intuitively, a large P value favors feasible states and makes $f(x)$ negligible in E, challenging the search for OPT. Conversely, a small P value weakens the constraints and favors unfeasible x at low energy. The minimization process corresponds to:

$$LB_P = \min_x E \tag{4}$$

where LB_P is a lower bound on the original problem described by Eq. (2) since we have $LB_P = \min_x E \leq E(x^*) = OPT$ with x^* an optimal solution for Eq. (2). The equality $LB_P = OPT$ occurs for sufficiently large $P \geq P_C$ where P_C is a critical value that depends on each instance [12]. Thus, there is a clear trade-off between accuracy and feasibility since with small P values the Ising machine is more likely to find lower bounds that are unfeasible by definition, and with large P the rugged energy surface induces hard-to-escape local minima [19], [24]. However, we show next that it is possible to relax the $P \geq P_C$ requirement by adding a linear contribution of constraints.

B. Lagrange Relaxation for the Penalty Method

To avoid finding $P \geq P_C$ that can lead to impractical energy penalties, we propose to harness a Lagrange relaxation of constraints $g(x)$ with some $P < P_C$. Compared to the penalty method, the Lagrange relaxation *linearly* weights constraints by *Lagrange multipliers* $\lambda \in \mathbb{R}^m$ and adds them to the energy function as [22]:

$$L = E(x) + \lambda^T g(x) \tag{5}$$

For readers more familiar with Ising machines, the purpose of coefficients λ can be introduced in this way. Consider that by some mechanism, λ are variables evolving in time at a slower rate than the minimization process defined as:

$$LB_L = \min_x L \tag{6}$$

which provides $\bar{x} = \text{argmin}_x L$. If $g(\bar{x}) \neq 0$, $\bar{x}$ is unfeasible, and one way of penalizing the energy L would be to decrease (increase) λ if $g(\bar{x})$ is negative (positive) where variables λ act as adjusting forces to move the minimum $\bar{x}$ into a feasible region. Possible λ-dynamics could be:

$$\tau_\lambda \frac{d\lambda}{dt} = g(\bar{x}) = +\nabla_\lambda LB_L \tag{7}$$

where ∇_λ is a generalization of gradient defined for nondifferentiable functions [22], [25]. Note how this process can be interpreted as the maximization of the lower bound LB_L with respect to λ. For an arbitrary λ, LB_L can be smaller than OPT and the goal is to find the optimal λ^* such that ideally $LB_L = OPT$. In optimization theory, this is called the *dual problem* [26]:

$$D = \max_\lambda LB_L \tag{8}$$

Combined with the minimization process from Eq. (6), the additional maximization procedure Eq. (8) shapes the energy landscape to bring the minimum of L towards an optimal feasible point. Compared to the classical penalty method, adding Lagrange multipliers provides an additional degree of freedom for shaping the energy landscape and closing the gap $G = OPT - LB_L$, as illustrated in Fig. 2 with a toy example. Consequently, it is possible to get $G = 0$ with a smaller penalty parameter $P < P_C$ which greatly increases the accuracy for hard problems as we show next.

III. Self-Adaptive Ising Machine

A. Algorithm

We now present how to find the optimal Lagrange multipliers λ^* in a self-adaptive and iterative manner to get the minimum gap G and find good solutions for Eq. (2) during the process. The idea is to alternate the minimization process of Eq. (6) and the maximization of the lower bound Eq. (8) to bring the system to a feasible region. This mechanism is well-known in optimization and has been used extensively [22]. Since the dual function LB_L is a concave function of Lagrange multipliers λ [26], the optimal λ^* for the dual problem in Eq. (8) can be found by ascent in the λ space with a subgradient given by $\nabla_\lambda LB_L = g(\bar{x})$ [22]. Moreover, the iterative method also converges with a pseudo-minimum of L and is called the surrogate gradient method that ensures convergence when $L < D$ at each iteration [25]. This technique is useful for Ising machines since they are heuristic solvers and cannot guarantee to solve exactly Eq. (6) in practice.

Algorithm 1 Self-Adaptive IM for Constrained Optimization

Require: f and g
Ensure: Best feasible solution $(\bar{x}, f(\bar{x}))$
$(\lambda_0, P) \leftarrow (0, \alpha dN)$
for K iteration **do**
- Minimize L_k: $x_k = \text{argmin}_x L_k$ ▷ Ising Machine
- Store feasible $\hat{x_k}$ ▷ CPU
- Update: $\lambda_{k+1} \leftarrow \lambda_k + \eta\, g(x_k)$ ▷ CPU

end for
$\bar{x} \leftarrow \text{argmin}_k f(\hat{x_k})$

We now combine these techniques and adapt them to Ising machines. The proposed Algorithm 1 works as follows. First, the Lagrange multipliers are set to 0, and the penalty parameter is initialized to some value which can be problem-dependent. For the problems we solve hereafter, we follow the heuristic rule from [19], [21] and set $P = \alpha dN$ where d is the density of the Ising J matrix, N is the number of Ising spins (including slack spins), and α is a constant that can be adjusted for different problems. Then, for a fixed number of iterations K, the Lagrange function from Eq. (5) is minimized by an Ising machine, and the Lagrange multipliers are updated to maximize the lower bound Eq. (8) and potentially close the gap G. An iteration is similar to an epoch when training a neural network. It shapes the energy landscape such that an optimal solution of the initial problem from Eq. (2) tends to become a ground state of L. In practice, the Ising coefficients J_{ij} and h_i are consequently updated at each iteration k, and the Ising machine is re-programmed. Meanwhile, feasible solutions $\hat{x_k}$ are stored, and the best one is selected after the for-loop.

B. A probabilistic-bit proof-of-concept

The minimization process of Algorithm 1 is compatible with any Ising machine. As a proof of concept, we choose to emulate a probabilistic-bit (p-bit) Ising machine [27] in software since p-bit-based architectures are currently very scalable and compatible with various hardware platforms, digital [10], analog [28] or mixed-signal [29]. A p-computer is composed of stochastic neurons called *p-bit* that are interconnected by weights J_{ij} and biased by h_i. Each p-bit takes one of the two values $m_i = \pm 1$ and receives as input:

$$I_i = \sum_j J_{ij} m_j + h_i \tag{9}$$

which influences the p-bit state as:

$$m_i = \text{sign}\big[\tanh \beta I_i + \text{rand(-1,1)}\big] \tag{10}$$

where rand(-1,1) is a random number uniformly distributed between -1 and 1 modeling noise at the p-bit level. β is the inverse temperature parameter which sets the slope of

the p-bit activation function. Interconnected p-bits satisfying equations Eq. (9) and Eq. (10) are known to follow a Boltzmann distribution of state with probability [27]:

$$P\{m\} = \frac{\exp -\beta L\{m\}}{\sum_m \exp -\beta L\{m\}} \tag{11}$$

where L is our Lagrange function expressed in Eq. (5). In Matlab, we emulate the probabilistic Ising machine by sequentially updating equations Eq. (9) and Eq. (10) which corresponds to the Gibbs Monte Carlo sampling method for the probability distribution of Eq. (11) [27]. Assuming a system of N p-bits, we call Monte Carlo sweep (MCS) an update round for the N p-bits. To find good minima of L, we anneal the p-bits such as in simulated annealing (SA) [30] with a linear β-schedule swept from 0 to β_{max}. For each SA run k, we read the last sample of state $\{m\}$, corresponding to x_k in Algorithm 1. The Lagrange multipliers are updated from this last sample and so are J_{ij} and h_i before the next SA run.

IV. Results

We benchmark SAIM with knapsack problems that are generally hard and have many real-life applications such as resource allocation, capital budgeting, satellite management, etc. [14]. We first study the quadratic knapsack problem (QKP) [31] which is an Ising problem with a linear constraint and has been explored by several works using Ising machines [19]–[21], [24]. Finally, we focus on multidimensional knapsack problems (MKP) that are particular integer linear programs (ILP) with positive coefficients and multiple constraints [32]. The parameters used in the experiments are listed in Table I.

A. Quadratic Knapsack Problems

Fig. 3a illustrates QKP which is the generalization of the knapsack problem (NP-hard) where pairs of items also add value to the objective function [31]. It can be expressed as:

$$\begin{aligned} &\min_x -\frac{1}{2} x^T W x - h^T x \\ &x \in \{0;1\}^N \\ &\text{s.t. } A^T x \leq b \end{aligned} \tag{12}$$

where $h \in \mathbb{N}^N$ is the value vector for the items, W is a $N \times N$ positive integer and symmetric matrix representing the additional value when selecting pairs of items, $A \in \mathbb{N}^N$ are the weights of the items and $b \in \mathbb{N}$ the maximum capacity. By using additional slack variables x_S, we transform the inequality constraint into equality as $A^T x + x_S = b$ where $0 \leq x_S \leq b$. Using a binary decomposition, x_S is written as $x_S = x_S^0 + 2x_S^1 + ... + 2^{Q-1}x_S^{Q-1}$ where x_S^q are additional binary variables and $Q = \text{floor}(\log_2(b) + 1)$ is the number of additional variables. We include x_S in x of new dimension $N+Q$ and fill W and h with zeros accordingly. The extended vector $A \in \mathbb{N}^{N+Q}$ contains the additional binary coefficients.

TABLE I
Parameters used in QKP and MKP experiments.
MCS: Monte Carlo Sweep

Experiment	Penalty	MCS/run	Number of runs	β_{max}	η
QKP	$2dN$	1000	2000	10	20
MKP	$5dN$	1000	5000	50	0.05

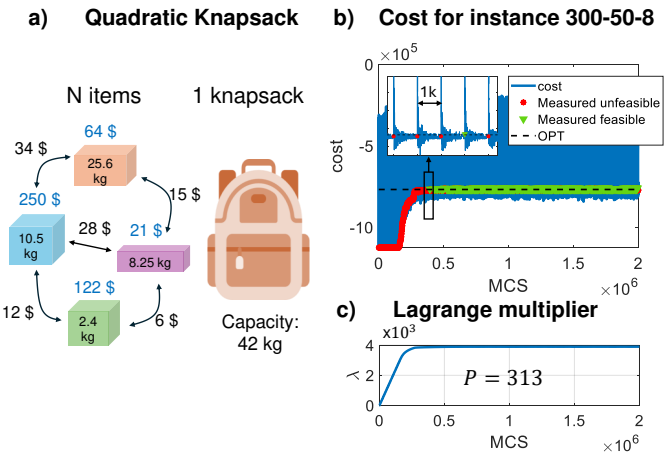


Fig. 3. a) The quadratic knapsack problem (QKP) consists of selecting items that maximize item value (blue $) and pairwise values (black $) given a capacity constraint (knapsack). b) Example of simulation result for 300 items and 50% of density. c) Corresponding Lagrange multiplier evolution. λ remains constant during each SA run of 10^3 MCS (staircase curve).

We benchmark with instances from [31] that consists of random QKP instances with various W-density $d \in \{0.25; 0.5; 0.75; 1\}$ and sizes $N \in \{100; 200; 300\}$. We normalize W, h by $\max(|W|, |h|)$ and A, b by $\max(|A|, |b|)$ to keep the same β schedule for all instances. For every measured sample x_k, we check feasibility as $A^T x_k \leq b$ and if feasible save its cost $c(\hat{x_k}) = -\hat{x_k}^T W \hat{x_k}/2 - h^T \hat{x_k}$. Since costs are negative, we measure the accuracy (%) of feasible samples as $Accuracy = 100\, c(\hat{x_k})/OPT$.

Fig. 3b presents an example of cost evolution for 300 items and 50% of density. Initially, λ is small and the measured samples are all unfeasible (red data points). During this transient time (at the λ time scale), the SAIM minimization produces unfeasible samples with a cost $c(x_k) < OPT$, highlighting that the chosen penalty parameter $P = 2dN = 313$ is too small. However, as shown in Fig. 3c, the Lagrange multiplier eventually converges to a steady value λ^* for which the Ising machine finds good feasible solutions (green triangles).

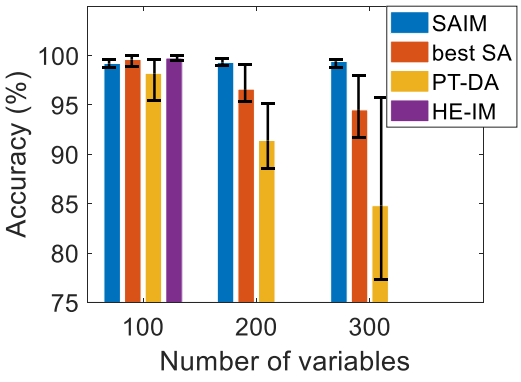
We now compare SAIM and the penalty method with the same total amount of 2×10^6 MCS for $N = 100$ $d = 0.25$ and $d = 0.5$ (Table II). For the penalty method, we run 10 SA runs of 2×10^5 MCS each with penalty parameters P tuned in the following manner. An initial small $P = 2dN$ was set and coarsely increased until getting a satisfactory feasibility ratio ($\geq 20\%$). We note that on average, a large P value implies a feasibility increase, as it has been observed in previous works [19], [24]. However, we did not find a clear correlation between P values and accuracies. Overall, finding a satisfactory P value is tedious and the tuning phase worsens the time-to-solution.

In contrast, the proposed SAIM is less parameter-sensitive as P is set once to $2dN$ for all instances and Lagrange

TABLE II
PENALTY METHOD VS. SAIM FOR QKP.
EACH INSTANCE IS NAMED WITH THE PREFIX N-d.
PARENTHESIS INDICATE FEASIBILITY.

	2000 SA runs of 10^3 MCS				10 SA runs of 2×10^5 MCS		
	SAIM		Penalty method		Penalty method		
Instance	Best	Avg	Best	Avg	Best	Avg	Tuned P
100_25_1	**100**	**99.6 (73)**	92.0	41.0 (87)	97	94.8 (30)	130dN
100_25_2	**100**	**98.5 (31)**	83.3	36.7 (90)	82.9	71.7 (80)	60dN
100_25_3	**100**	**98.0 (54)**	57.5	15.5 (96)	77.0	67.1 (50)	70dN
100_25_4	**100**	**99.2 (66)**	90.7	35.4 (97)	77.8	71.0 (70)	120dN
100_25_5	**99.2**	**99.2 (37)**	86.6	44.9 (92)	86.6	69.4 (90)	60dN
100_25_6	**100**	**99.2 (60)**	94.8	36.7 (95)	98.9	91.8 (30)	250dN
100_25_7	**100**	**99.0 (34)**	70.9	33.9 (94)	83.4	72.2 (40)	200dN
100_25_8	**100**	**97.3 (24)**	87.8	35.5 (96)	88.3	87.5 (30)	200dN
100_25_9	**100**	**99.3 (54)**	96.2	37.5 (95)	83.4	79.8 (50)	300dN
100_25_10	**99.98**	**99.3 (54)**	75.0	17.7 (98)	95.0	83.0 (60)	500dN
100_50_1	**99.8**	**98.9 (49)**	94.5	38.8 (95)	91.2	82.4 (60)	400dN
100_50_2	**99.6**	**99.5 (68)**	81.0	63.6 (74)	80.7	67.8 (20)	40dN
100_50_3	**99.1**	**97.4 (28)**	78.6	28.3 (95)	97.3	86.2 (20)	100dN
100_50_4	**99.9**	**99.8 (82)**	83.1	37.2 (92)	81.5	69.7 (40)	40dN
100_50_5	**99.9**	**97.5 (12)**	92.8	40.3 (92)	94.7	90.6 (40)	300dN
100_50_6	**100**	**99.9 (91)**	68.2	21.5 (97)	75.3	66.0 (40)	200dN
100_50_7	**99.9**	**99.7 (73)**	88.1	24.0 (96)	99.4	97.7 (40)	220dN
100_50_8	**99.9**	**99.1 (52)**	89.8	25.4 (95)	99.7	93.3 (30)	220dN
100_50_9	**100**	**99.8 (91)**	93.1	35.2 (94)	97.4	92.1 (40)	350dN
100_50_10	**99.4**	**99.0 (49)**	95.0	60.2 (94)	87.6	79.0 (50)	150dN
Average	**99.8**	**99.0 (54)**	85.0	35.5 (93)	88.8	80.7 (47)	195dN

a) Accuracy comparison for QKP



b) Number of Monte Carlo Sweeps

	SAIM	Best SA	HE-IM	PT-DA
MCS	2 M	200 M	19.5 G	15 G
Speedup	-	100x	9,750x	7,500x

Fig. 4. a) Accuracy comparison for QKP (quartiles). b) Number of reported Monte Carlo sweeps for each method and SAIM speedup.

multipliers are automatically updated. Looking at the best-obtained accuracies, we measure an average of 99.8% for SAIM against 88.8% for the tuned penalty method. We also tested the penalty method in the same setup as SAIM, i.e. with 2000 SA runs of 10^3 MCS each. The best accuracy for the penalty method decreased to 85% on average, highlighting that the high SAIM accuracy is not due to a large number of SA runs.

Next, we benchmark SAIM with 4 previous works using probabilistic Ising machines standalone that do not post-process their output. The accuracy can be enhanced using heuristics [20] but they are problem-dependent and beyond the scope of this paper. For each method, we compare the accuracy and the number of Monte Carlo sweeps reflecting the algorithmic performance. Time-to-solution depends on the hardware and is not addressed in this paper, although it can be estimated knowing the computation time per MCS.

All the previous works we have found use the penalty method. We first benchmark with work [21] that explores various QUBO encodings and solves QKP with SA. For each instance, we report the best accuracy reported in [21] (labeled best SA). We also benchmark with an implementation of the parallel tempering algorithm executed on Fujitsu's Digital Annealer using 26 replicas (PT-DA) [19]. Finally, SAIM results are compared against the work [24] that uses a hybrid encoding for the slack variables x_S for problems up to 100 variables, and uses SA to find good solutions (HE-IM).

Fig. 4 summarizes the results obtained with state-of-the-art Ising machines. The SAIM median accuracy is larger than 99.2% for all sizes and the solutions are consistently of high quality with interquartile ranges smaller than 0.8%. In contrast for 300 variables, the median accuracies for the best SA algorithm [21] and PT-DA [19] are 94.5% and 84.8%, respectively. As shown in Fig. 4b, SAIM requires much fewer samples. The best SA method from [21] has reported 2×10^5 trials of 10^3 MCS. The HE-IM [24] used 26 trials of 7.5×10^8 MCS. PT-DA [19] reported 20 trials of 7.5×10^8 MCS. Overall, SAIM requires 100x fewer samples than the best SA runs and 7,500x fewer than PT-DA.

B. Multidimensional Knapsack Problems

To test the applicability of SAIM with multiple constraints, we focus on the NP-hard MKP which consists of selecting items that are subjected to several capacity constraints (knapsacks) and expressed as [14]:

$$\begin{aligned} &\min_x -h^T x \\ &x \in \{0;1\}^N \\ &\text{s.t. } A\,x \leq B \end{aligned} \tag{13}$$

where $h \in \mathbb{N}^N$ is the value vector for the items, A is a positive integer $M \times N$ matrix representing the item weights, and $B \in \mathbb{N}^M$ the vector of maximum capacities. As for QKP, we transform the inequality constraints into equality constraints and normalize h, A, and B similarly. Since there are no quadratic interactions (W matrix), we approximate the problem density d as $N/(0.5N(N+1)) = 2/(N+1)$ as if the external fields h were pairwise connections from an additional fixed spin reference to the N initial spins. Compared to QKP, we set $P = 5dN$ rather than $2dN$ to compensate the lack of quadratic interaction in the initial cost function.

We benchmark SAIM with a state-of-the-art genetic algorithm for MKP (GA) [32] using instances with $N \in \{100; 250\}$

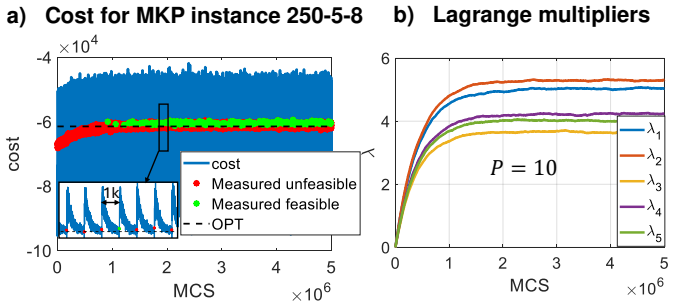


Fig. 5. a) Example of MKP simulation for a 250-variable instance and 5 knapsacks. b) Corresponding Lagrange multipliers dynamics for a fixed $P = 10$. λ remain constant during each SA run of 10^3 MCS (staircase curve).

and $M \in \{5; 10\}$. Fig. 5 shows an example of SAIM simulation for 250 items and 5 knapsacks. Initially, the constraints are unsatisfied with $A^T x_k > B$ (the total weights exceed knapsack capacities), and the five corresponding Lagrange multipliers λ are updated after each iteration (they increase since $A^T x_k - B \geq 0$). Then, after approximately 1000 updates, λ starts to stabilize and SAIM finds near-optimal solutions.

Table III shows the detailed results obtained with a total of 5×10^6 MCS. Optimal solutions are obtained with a branch and bound (B&B) algorithm (*intlinprog* Matlab function) and its execution time on a standard laptop is reported to compare the relative difficulty between instances. For three various classes of problems, the average best SAIM accuracy is 99.7%. Although the authors in [32] reported a mean lower bound of 99.1% for the GA accuracy, this suggests SAIM solutions are comparable to GA. Obtaining a similar accuracy is encouraging since SAIM does not *a priori* harness the problem structure, unlike GA [32] that is tailored for MKP.

However, the proportion of feasible samples for MKP (5.1%) is severely reduced compared to the previous QKP study (around 50%). We believe this is mainly because multiple constraints are harder to satisfy simultaneously. To increase feasibility, one could increase the initial penalties set by P. Another approach proposed in [21] would be to reduce the knapsack capacities B artificially as $B' < B$ so that the measured samples are more likely to satisfy the constraints.

V. Discussion

The proposed SAIM algorithm is based on the penalty method and adds a Lagrange relaxation of constraints. There is probably a connection to the *Method of Multipliers* [15] also called *Augmented Lagrange Method* [33] from the field of continuous optimization, although its derivation assumes differentiable functions. In this scheme, both the Lagrange multiplier λ and the penalty parameter P are iteratively updated and depend on each other, which could inspire further SAIM refinement for an automatic P-tuning mechanism.

Although SAIM found excellent solutions for knapsack problems, it should be tested on other constrained problems to assess its applicability, which we hope could be as broad as the Lagrange relaxation method [22]. Moreover, the algorithm targets physical Ising machines and still needs evaluation in practical settings involving non-idealities and limited precision.

TABLE III
MKP results. Instances are named with the prefix N-M.

			SAIM		GA [32]
Instance	B&B time (s)	Optimality (%)	Best	Avg	Avg
100_5_1	34	0.5	100	98.8 (7.5)	
100_5_2	8.5	16.4	100	99.1 (7)	
100_5_3	13	0	99.9	99.4 (6)	
100_5_4	42	0	99.8	98.3 (7.9)	
100_5_5	25	0	99.9	98.8 (7.7)	≥99.1
100_5_6	7	1.6	100	99.0 (7.7)	
100_5_7	5	2.9	100	98.8 (8.3)	
100_5_8	20	1.6	100	98.9 (11.6)	
100_5_9	9	0.2	100	98.9 (8.5)	
100_5_10	20	1.9	100	98.8 (7.2)	
100_10_1	425	0	99.97	98.4 (2.5)	
100_10_2	364	0	99.6	98.5 (2.1)	
100_10_3	189	0	99.7	98.1 (2.1)	
100_10_4	468	0	99.9	97.6 (2.2)	
100_10_5	172	0	99.3	97.7 (2.3)	≥98.4
100_10_6	899	0	99.4	97.6 (2.2)	
100_10_7	110	1.2	100	98.6 (7.6)	
100_10_8	84	0	99.6	97.8 (1.6)	
100_10_9	70	0	99.6	98.2 (2.1)	
100_10_10	76	0	99.98	98.0 (1.8)	
250_5_1	327	0	99.5	98.5 (5.1)	
250_5_2	617	0	99.6	98.5 (5.2)	
250_5_3	29	0	99.4	98.4 (4.7)	
250_5_4	1650	0	99.3	98.2 (4.4)	
250_5_5	777	0	99.5	98.4 (4.8)	≥99.8
250_5_6	1272	0	99.5	98.4 (4.7)	
250_5_7	348	0	99.5	98.4 (5)	
250_5_8	1580	0	99.7	98.3 (5)	
250_5_9	163	0	99.7	98.6 (4.4)	
250_5_10	49	0	99.4	98.2 (4.7)	
Average	328	0.9	**99.7**	98.4 (5.1)	≥99.1

VI. Conclusion

This paper introduces a self-adaptive Ising machine (SAIM) for constrained optimization that iteratively shapes its energy landscape using a Lagrange relaxation of constraints to avoid tuning energy penalties. Emulated with a probabilistic-bit Ising machine in software, we benchmark SAIM for hard quadratic knapsack problems (QKP) and multidimensional knapsack problems with multiple constraints. For QKP, SAIM finds better solutions than state-of-the-art Ising machines such as a parallel tempering algorithm executed on Fujitsu's Digital Annealer and produces 7,500x fewer samples. Compatible with any programmable Ising machine, SAIM has the potential to significantly speed up Ising machines for constrained optimization.

Acknowledgment

We acknowledge support from ONR-MURI grant N000142312708, OptNet: Optimization with p-Bit Networks, and we thank Kerem Yunus Camsari for the enriching discussions.

Data Availability

Benchmark data and Matlab codes are available at the following GitHub repository: https://github.com/corentindelacour/self-adaptive-IM

REFERENCES

[1] N. Mohseni, P. L. McMahon, and T. Byrnes, "Ising machines as hardware solvers of combinatorial optimization problems," *Nature Reviews Physics*, vol. 4, no. 6, pp. 363–379, May 2022, publisher: Springer Science and Business Media LLC.

[2] P. L. McMahon, A. Marandi, Y. Haribara, R. Hamerly, C. Langrock, S. Tamate, T. Inagaki, H. Takesue, S. Utsunomiya, K. Aihara, R. L. Byer, M. M. Fejer, H. Mabuchi, and Y. Yamamoto, "A fully programmable 100-spin coherent Ising machine with all-to-all connections," *Science*, vol. 354, no. 6312, pp. 614–617, Nov. 2016.

[3] T. Honjo, T. Sonobe, K. Inaba, T. Inagaki, T. Ikuta, Y. Yamada, T. Kazama, K. Enbutsu, T. Umeki, R. Kasahara, K.-i. Kawarabayashi, and H. Takesue, "100,000-spin coherent Ising machine," *Science Advances*, vol. 7, no. 40, p. eabh0952, Oct. 2021.

[4] Z. Fahimi, M. R. Mahmoodi, H. Nili, V. Polishchuk, and D. B. Strukov, "Combinatorial optimization by weight annealing in memristive hopfield networks," *Scientific Reports*, vol. 11, no. 1, p. 16383, Aug. 2021.

[5] M. Jiang, K. Shan, C. He, and C. Li, "Efficient combinatorial optimization by quantum-inspired parallel annealing in analogue memristor crossbar," *Nature Communications*, vol. 14, no. 1, p. 5927, Sep. 2023.

[6] T. Wang, L. Wu, P. Nobel, and J. Roychowdhury, "Solving combinatorial optimisation problems using oscillator based Ising machines," *Natural Computing*, vol. 20, no. 2, pp. 287–306, Jun. 2021.

[7] S. Dutta, A. Khanna, A. S. Assoa, H. Paik, D. G. Schlom, Z. Toroczkai, A. Raychowdhury, and S. Datta, "An ising hamiltonian solver based on coupled stochastic phase-transition nano-oscillators," *Nature Electronics*, vol. 4, no. 7, pp. 502–512, Jul 2021.

[8] M. K. Bashar, A. Mallick, and N. Shukla, "Experimental Investigation of the Dynamics of Coupled Oscillators as Ising Machines," *IEEE Access*, vol. 9, pp. 148 184–148 190, 2021.

[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, Apr. 2019, publisher: Frontiers Media SA.

[10] N. A. Aadit, A. Grimaldi, M. Carpentieri, L. Theogarajan, J. M. Martinis, G. Finocchio, and K. Y. Camsari, "Massively parallel probabilistic computing with sparse ising machines," *Nature Electronics*, vol. 5, no. 7, pp. 460–468, Jul 2022.

[11] A. D. King, J. Raymond, T. Lanting, R. Harris, A. Zucca, F. Altomare, A. J. Berkley, K. Boothby, S. Ejtemaee, C. Enderud, E. Hoskinson, S. Huang, E. Ladizinsky, A. J. R. MacDonald, G. Marsden, R. Molavi, T. Oh, G. Poulin-Lamarre, M. Reis, C. Rich, Y. Sato, N. Tsai, M. Volkmann, J. D. Whittaker, J. Yao, A. W. Sandvik, and M. H. Amin, "Quantum critical dynamics in a 5,000-qubit programmable spin glass," *Nature*, vol. 617, no. 7959, pp. 61–66, May 2023, publisher: Springer Science and Business Media LLC.

[12] A. Lucas, "Ising formulations of many NP problems," *Frontiers in Physics*, vol. 2, 2014.

[13] L. A. Wolsey, *Integer programming*. John Wiley & Sons, 2020.

[14] C. Wilbaut, S. Hanafi, and S. Salhi, "A survey of effective heuristics and their application to a variety of knapsack problems," *IMA Journal of Management Mathematics*, vol. 19, no. 3, pp. 227–244, Mar. 2007. [Online]. Available: https://academic.oup.com/imaman/article-lookup/doi/10.1093/imaman/dpn004

[15] M. R. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, no. 5, pp. 303–320, Nov. 1969.

[16] T. Krauss and J. McCollum, "Solving the Network Shortest Path Problem on a Quantum Annealer," *IEEE Transactions on Quantum Engineering*, vol. 1, pp. 1–12, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9186612/

[17] K. Ikeda, Y. Nakamura, and T. S. Humble, "Application of Quantum Annealing to Nurse Scheduling Problem," *Scientific Reports*, vol. 9, no. 1, p. 12837, Sep. 2019. [Online]. Available: https://www.nature.com/articles/s41598-019-49172-3

[18] C. Carugno, M. Ferrari Dacrema, and P. Cremonesi, "Evaluating the job shop scheduling problem on a D-wave quantum annealer," *Scientific Reports*, vol. 12, no. 1, p. 6539, Apr. 2022. [Online]. Available: https://www.nature.com/articles/s41598-022-10169-0

[19] M. Parizy and N. Togawa, "Analysis and Acceleration of the Quadratic Knapsack Problem on an Ising Machine," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E104.A, no. 11, pp. 1526–1535, Nov. 2021.

[20] K. Ohno, T. Shirai, and N. Togawa, "Toward practical benchmarks of ising machines: A case study on the quadratic knapsack problem," *IEEE Access*, vol. 12, pp. 97 678–97 690, 2024.

[21] T. Bontekoe, F. Phillipson, and W. v. d. Schoot, "Translating constraints into qubos for the quadratic knapsack problem." Berlin, Heidelberg: Springer-Verlag, 2023, p. 90–107.

[22] M. L. Fisher, "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, vol. 50, no. 12,, pp. 1861–1871, 2004.

[23] C. Bybee, D. Kleyko, D. E. Nikonov, A. Khosrowshahi, B. A. Olshausen, and F. T. Sommer, "Efficient optimization with higher-order ising machines," *Nature Communications*, vol. 14, no. 1, p. 6033, Sep. 2023.

[24] S. Jimbo, D. Okonogi, K. Ando, T. V. Chu, J. Yu, M. Motomura, and K. Kawamura, "A Hybrid Integer Encoding Method for Obtaining High-Quality Solutions of Quadratic Knapsack Problems on Solid-State Annealers," *IEICE Transactions on Information and Systems*, vol. E105.D, no. 12, pp. 2019–2031, Dec. 2022.

[25] Xing Zhao, P. Luh, and Jihua Wang, "The surrogate gradient algorithm for Lagrangian relaxation method," in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 1. San Diego, CA, USA: IEEE, 1997, pp. 305–310.

[26] S. P. Boyd and L. Vandenberghe, *Convex optimization*, version 29 ed. Cambridge New York Melbourne New Delhi Singapore: Cambridge University Press, 2023.

[27] K. Y. Camsari, R. Faria, B. M. Sutton, and S. Datta, "Stochastic p -Bits for Invertible Logic," *Physical Review X*, vol. 7, no. 3, p. 031014, Jul. 2017.

[28] J. Kaiser, W. A. Borders, K. Y. Camsari, S. Fukami, H. Ohno, and S. Datta, "Hardware-Aware *In Situ* Learning Based on Stochastic Magnetic Tunnel Junctions," *Physical Review Applied*, vol. 17, no. 1, p. 014016, Jan. 2022.

[29] N. S. Singh, K. Kobayashi, Q. Cao, K. Selcuk, T. Hu, S. Niazi, N. A. Aadit, S. Kanai, H. Ohno, S. Fukami, and K. Y. Camsari, "CMOS plus stochastic nanomagnets enabling heterogeneous computers for probabilistic inference and learning," *Nature Communications*, vol. 15, no. 1, p. 2685, Mar. 2024.

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

[31] A. Billionnet and Éric Soutif, "An exact method based on lagrangian decomposition for the 0–1 quadratic knapsack problem," *European Journal of Operational Research*, vol. 157, no. 3, pp. 565–575, 2004.

[32] P. C. Chu and J. E. Beasley, "A Genetic Algorithm for the Multidimensional Knapsack Problem," *Journal of Heuristics*, 1998.

[33] M. J. D. Powell, "Algorithms for nonlinear constraints that use lagrangian functions," *Mathematical Programming*, vol. 14, no. 1, pp. 224–248, Dec. 1978.