HyCiM: A Hybrid Computing-in-Memory QUBO Solver for General Combinatorial Optimization Problems with Inequality Constraints

Yu Qian[1], Zeyu Yang[1], Kai Ni[2], Alptekin Vardar[3], Thomas Kämpfe[3] and Xunzhao Yin[1,4*]
[1]Zhejiang University, Hangzhou, China; [2]University of Notre Dame, Notre Dame, USA; [3]Fraunhofer IPMS, Dresden, Germany
[4]Key Laboratory of CS&AUS of Zhejiang Province, Hangzhou, China; *Corresponding author email: xzyin1@zju.edu.cn

ABSTRACT

Computationally challenging combinatorial optimization problems (COPs) play a fundamental role in various applications. To tackle COPs, many Ising machines and Quadratic Unconstrained Binary Optimization (QUBO) solvers have been proposed, which typically involve direct transformation of COPs into Ising models or equivalent QUBO forms (D-QUBO). However, when addressing COPs with inequality constraints, this D-QUBO approach introduces numerous extra auxiliary variables, resulting in a substantially larger search space, increased hardware costs, and reduced solving efficiency. In this work, we propose HyCiM, a novel hybrid computing-in-memory (CiM) based QUBO solver framework, designed to overcome aforementioned challenges. The proposed framework consists of (i) an innovative transformation method (first to our known) that converts COPs with inequality constraints into an inequality-QUBO form, thus eliminating the need of expensive auxiliary variables and associated calculations; (ii) "inequality filter", a ferroelectric FET (FeFET)-based CiM circuit that accelerates the inequality evaluation, and filters out infeasible input configurations; (iii) a FeFET-based CiM annealer that is capable of approaching global solutions of COPs via iterative QUBO computations within a simulated annealing process. The evaluation results show that HyCiM drastically narrows down the search space, eliminating 2^{100} to 2^{2536} infeasible input configurations compared to the conventional D-QUBO approach. Consequently, the narrowed search space, reduced to 2^{100} feasible input configurations, leads to a substantial hardware area overhead reduction, ranging from 88.06% to 99.96%. Additionally, HyCiM consistently exhibits a high solving efficiency, achieving a remarkable average success rate of 98.54%, whereas D-QUBO implementatoin shows only 10.75%.

1 INTRODUCTION

Combinatorial optimization problems (COPs) find applications in various domains, including logistics, resource allocation, communication network design, and transportation systems, among others [1, 2]. Many of these problems are classified as non-deterministic polynomial-time hard (NP-hard), indicating their complexity within the NP realm. A wide range of COPs, including Max-Cut, graph coloring, and knapsack problems, can be effectively reformulated into Ising models or Quadratic Unconstrained Binary Optimization (QUBO) forms, which can be addressed by Ising machines and QUBO solvers [3, 4]. For example, digital ASIC annealers [5, 6] implement different annealing algorithms within digital circuits, while dynamical system Ising machines [7, 8] utilize the intrinsic system dynamics and their tendency to settle at ground state to solve COPs. These solvers are specifically designed to solve COPs much more quickly and efficiently than von Neumann hardware.

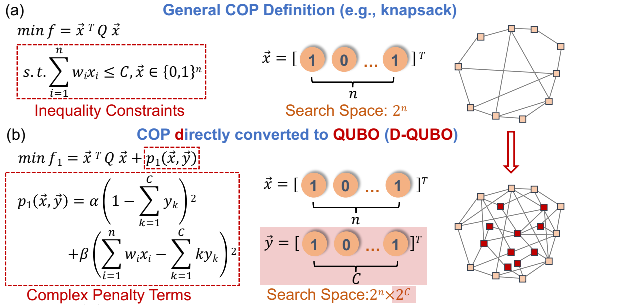
(a)
General COP Definition (e.g., knapsack)
$min\, f = \vec{x}^{\,T} Q\, \vec{x}$
$s.t. \sum_{i=1}^{n} w_i x_i \leq C, \vec{x} \in \{0,1\}^n$
Inequality Constraints
$\vec{x} = [\,1\ 0\ \ldots\ 1\,]^T$
n
Search Space: 2^n
(b)
COP directly converted to QUBO (D-QUBO)
$min\, f_1 = \vec{x}^{\,T} Q\, \vec{x} + p_1(\vec{x}, \vec{y})$
$p_1(\vec{x},\vec{y}) = \alpha\left(1 - \sum_{k=1}^{C} y_k\right)^2 + \beta\left(\sum_{i=1}^{n} w_i x_i - \sum_{k=1}^{C} k y_k\right)^2$
Complex Penalty Terms
$\vec{x} = [\,1\ 0\ \ldots\ 1\,]^T$
n
$\vec{y} = [\,1\ 0\ \ldots\ 1\,]^T$
C
Search Space: $2^n \times 2^C$

Figure 1: (a) A general COP with an inequality constraint involves a search space of 2^n variable configurations; (b) D-QUBO introduces extra scaled auxiliary variable vector $\vec{y}$, expanding the search space to 2^{n+C}.

It's worth noting that existing Ising machines and QUBO solvers cannot directly address COPs. The transformation of COPs into Ising models or their equivalent QUBO forms [9] is a prerequisite. However, existing solvers exhibit certain limitations when transforming general COPs with inequality constraints into QUBO forms, such as (quadratic) knapsack and bin packing [10]. As Fig. 1 shows, traditional QUBO transformation directly embeds the inequality constraints as penalty terms in the objective function, referred to as "D-QUBO", introducing extra auxiliary variables and associated calculations. Such direct transformation significantly expands the search space, and increases both the number and size of parameters in the objective function, leading to much slower solution convergence and larger hardware resources. Due to the limited capacity of existing QUBO solvers and the complexity introduced by D-QUBO, many COPs with inequality constraints remain unsolvable [11].

To address aforementioned challenges, we introduce HyCiM, a hybrid computing-in-memory (CiM) QUBO solver framework that efficiently solves general COPs while minimizing hardware overhead and optimizing solving efficiency. The key contributions of this work are summarized as follows:

- A novel transformation method is proposed to transform a general COP with inequality constraints into an inequality-QUBO form. This conversion avoids introducing auxiliary variables and significantly reduces the complexity of the search space and objective function (Sec. 3.2).
- We proposed "inequality filter", a ferroelectric FET (FeFET)-based CiM architecture that accelerates inequality evaluations and filters out infeasible input configurations by leveraging the multi-level characteristics of FeFETs (Sec. 3.3).

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06…$15.00
https://doi.org/10.1145/3649329.3655989

- We proposed a FeFET-based CiM annealer that approaches global solutions of COPs through iterative QUBO vector-matrix-vector (VMV) multiplication computations within a simulated annealing (SA) process (Sec. 3.4).
- HyCiM achieved significant reduction in both search space and object function complexity, resulting in a higher COP-solving efficiency compared to other QUBO solvers (Sec. 4).

Our evaluation clearly demonstrate that in solving 40 quadratic knapsack problems with 100 items, HyCiM offers a substantial reduction in the search space (eliminating 2^{100} to 2^{2536} infeasible input configurations) compared to the conventional D-QUBO approach. This reduction leads to significant hardware overhead savings, ranging from 88.06% to 99.96%. Furthermore, HyCiM demonstrates high solving efficiency, achieving an impressive success rate of 98.54%, while D-QUBO implementation reaches only 10.75%.

2 BACKGROUND

In this section, we first introduce the QUBO related concepts and works, then review FeFET based CiM basics.

2.1 QUBO basics and existing solvers

Ising models offer a potent framework for modeling and solving COPs [4], where problem variables are represented as spins, and constraints are captured by spin couplings. The objective function of the problem translates into the Hamiltonian energy function of the Ising model. Problem solving entails finding the spin configuration that minimizes the corresponding Ising Hamiltonian H_{P}:

$$\min H_{\text{P}} = \sum_{i,j=1}^{N} J_{ij}\sigma_i\sigma_j + \sum_{i=1}^{N} h_i\sigma_i \tag{1}$$

where N denotes the number of spins and $\sigma_i \in \pm 1$ represents the state of spin i. J_{ij} and h_i stand for the coupling between spin i and j and the self-coupling of spin i, respectively.

The Ising Hamiltonian (Eq. (1)) is equivalent to the QUBO form [9] through a simple variable transformation $\sigma_i = 1-2x_i, x_i \in \{0,1\}$. The QUBO form can be elegantly expressed as:

$$\min y = \vec{x}^{\,T} Q\vec{x} \tag{2}$$

where $\vec{x} \in \{0,1\}^n$, and Q is an $n \times n$ matrix.

While certain COPs, e.g., Max-Cut, graph coloring, can be seamlessly converted into QUBO, more general COPs with inequality constraints, e.g., (quadratic) knapsack, bin packing, etc [10], pose challenges for direct QUBO transformation.

Fig. 1(a) depicts a general COP with an inequality constraint. The inherent search space of the problem is defined by the dimension of variable $\vec{x}$, specifically 2^n. Fig. 1(b) illustrates the D-QUBO transformation of the COP in (a) [10, 12], which introduces an auxiliary variable $\vec{y}$ and an extra penalty function $p_1(\vec{x},\vec{y})$ to represent the impact of inequality constraint. The first term in $p_1(\vec{x},\vec{y})$ tends to guarantee that $\vec{y}$ has only one element equal to 1 and $\sum_{k=1}^{C} ky_k$ lies within the range of $\{1,2,...,C\}$, while the second term tends to ensure that $\sum_{i=1}^{n} w_i x_i = \sum_{k=1}^{C} ky_k$, thus satisfying the inequality constraint. The auxiliary variable $\vec{y}$ and penalty function $p_1(\vec{x},\vec{y})$ significantly expand the search space from 2^n to 2^{n+C} and increase the number and size of parameters. Addressing such D-QUBO form requires large-scale Ising machine and QUBO solver, resulting in higher hardware overhead and degraded solution convergence.

Numerous Ising machines and QUBO solvers have been developed to tackle D-QUBO forms. Digital ASIC annealers, implementing diverse annealing algorithms to solve Ising models [5, 6], may consume substantial energy and time due to the exponential increase in explicit computations during each annealing iteration. Dynamic system Ising machines leverages the inherent nature of physical systems to explore optimal solutions [13, 14], but they are highly sensitive to spin coupling implementations, that even slight deviations in coupling strength can disrupt convergence [8, 13]. Moreover, these Ising solvers are limited to COPs without self-interaction terms, such as Max-Cut and Sherrington-Kirkpatrick models, as integrating self-interaction into these solvers is non-trivial [7, 15]. In this work, we propose an alternative inequality-QUBO implementation to enhance the solving efficiency while avoiding the challenges associated with D-QUBO forms.

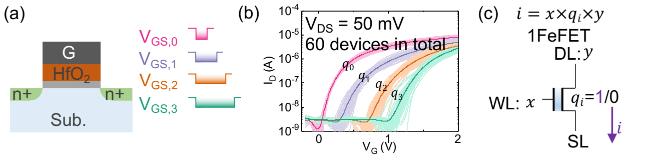

Figure 2: (a) By applying different write pulses, (b) multi-level I_D-V_G characteristics of FeFETs can be programmed storing q_0 to q_3. (c) FeFET storing a binary bit can naturally perform consecutive multiplications $i = x \times q_i \times y$ by simultaneously applying the inputs x and y to gate and drain, respectively.

2.2 CiM Preliminaries and FeFET Basics

Prior CiM implementations employs non-volatile memories (NVMs) such as resistive random access memory (ReRAM) and magnetic tunneling junction (MTJ) to build crossbars that store weight matrix, and perform parallel vector-matrix multiplication (VMM) operations for neural network accelerations [16, 17]. However, these NVMs still face challenges for enhancing energy and area efficiency due to the current-driven mechanism, low ON/OFF ratio and large driving transistors. FeFET, which integrates HfO$_2$ as the ferroelectric dielectric within the gate stack of MOSFET, has emerged as a promising candidate for embedded memory and CiM. Compared with ReRAM [18] and MTJ [19], FeFET offers better energy and area efficiency due to the CMOS compatibility, voltage-driven mechanism, high ON/OFF ratio and three-terminal structure [20–23].

When applied with different gate pulses as shown in Fig. 2(a) at gate, a FeFET can store multi-level $I_D - V_G$ curves which are experimentally measured from 60 devices as shown in Fig. 2(b). Such multi-level characteristic can be utilized for our proposed FeFET-based CiM inequality filter. When storing binary bit q_i, a FeFET can inherently perform multiplication $i = x \times q_i \times y$ through drain-source current i by applying input signals x and y to gate WL and drain DL, respectively, as shown in Fig. 2(c). Such single transistor multiplication is well-suited for the proposed FeFET-based CiM crossbar supporting QUBO computation.

3 HYCIM FRAMEWORK

In this section, we introduce our proposed HyCiM, including core components and operational flow.

3.1 HyCiM Overview

Fig. 3 illustrates the overview of the proposed HyCiM framework. To tackle COPs with inequality constraints, e.g., knapsack, HyCiM first transforms these problems into an inequality-QUBO formulation (Sec. 3.2). The inequality-QUBO forms are then mapped onto a FeFET-based CiM inequality filter (Sec. 3.3) and a FeFET-based CiM

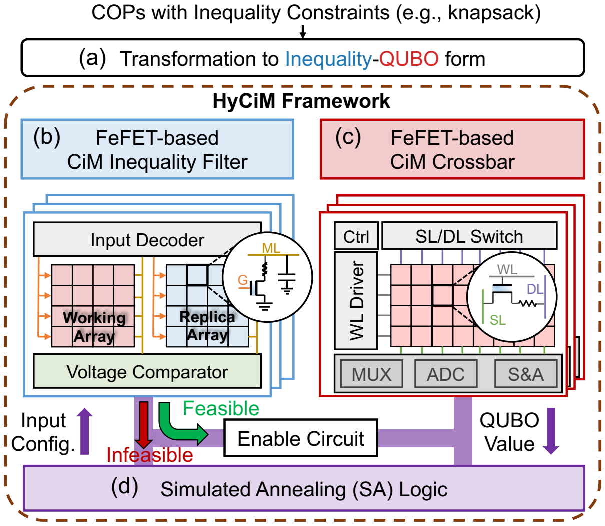
COPs with Inequality Constraints (e.g., knapsack)
(a) Transformation to Inequality-QUBO form
HyCiM Framework
(b) FeFET-based CiM Inequality Filter
(c) FeFET-based CiM Crossbar
Input Decoder
ML
G
Working Array
Replica Array
Voltage Comparator
Ctrl
SL/DL Switch
WL Driver
WL
DL
SL
MUX
ADC
S&A
Feasible
Input Config.
Infeasible
Enable Circuit
QUBO Value
(d) Simulated Annealing (SA) Logic

Figure 3: Overview of HyCiM. (a) Inequality-QUBO transformation method; (b) FeFET-based CiM inequality filter for input configuration filtering; (c)(d) FeFET-based CiM annealer for approaching optimal solutions.

Crossbar (Sec. 3.4), respectively. A SA logic is connected to these components to process the annealing (Sec. 3.4).

During each SA iteration, the SA logic generates an input variable configuration, which is then routed to the FeFET-based inequality filter. The inequality filter directly evaluates whether the configuration violates any constraints, and forwards feasible configuration to the FeFET-based CiM crossbar for further QUBO computations. Infeasible configurations are returned to SA logic to generate next input variable configuration.

3.2 Inequality-QUBO Transformation

COPs with inequality constraints represent a generalized category of COPs, as COPs without constraints or with equality constraints can be considered as special cases of COPS with inequality. In this work, we focus on the general COPs with inequality constraints, and take quadratic knapsack problem (QKP) as the representative. QKP is a more general form of knapsack problem, encompassing additional quadratic terms in the objective function [10, 12]: Given a set of items with weight and profit measurements, along with a knapsack limited in capacity, the objective is to select a subset of items that maximizes the total profit within the knapsack's capacity. Each item selected in the knapsack contributes an individual profit, and there are additional profits for pairs of selected items. Mathematically, the problem can be stated as follows:

$$\max \sum_{i,j=1}^{n} p_{ij} x_i x_j \tag{3}$$

$$st. \sum_{i=1}^{n} w_i x_i \leq C, x_i \in \{0, 1\} \tag{4}$$

where n is the number of items, $C \in Z^+$ is the capacity of knapsack, $w_i \in Z^+$ is the weight of item i, $p_{ij}(i \neq j)$ denotes the additional profit when x_i and x_j are selected, and $p_{ij}(i = j)$ denotes the individual profit of x_i. $p_{ij} = p_{ji}$. When item i is selected, $x_i = 1$, otherwise $x_i = 0$.

In our proposed transformation, Eq. (3) is transformed to

$$\min f = \sum_{i,j=1}^{n} q_{ij} x_i x_j = \vec{x}^{\,T} Q \vec{x} \tag{5}$$

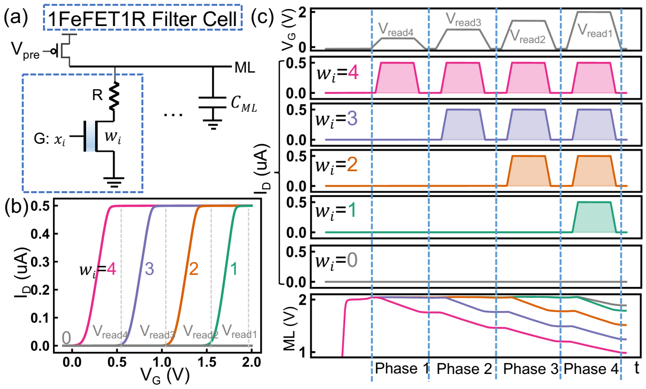

Figure 4: (a) Cell schematic of inequality filter array; (b) Transfer curves of cells storing five distinct weights and corresponding V_{read}'s; (c) Transient waveforms of filter cells storing five weights during the evaluation operation.

assuming $p_{ij} = -q_{ij}$. Subsequently, rather than embedding the constraint Eq. (4) directly into f as D-QUBO (Fig. 1(b)), we reformulate the objective function as follows:

$$\min E = \left(\sum_{i=1}^{n} w_i x_i \leq C\right) \times \vec{x}^{\,T} Q \vec{x} \tag{6}$$

The constraint Eq. (4) is transformed to a logical function, and E is non-positive. When Eq. (4) is satisfied, $E = \vec{x}^{\,T} Q \vec{x}$, assuming a negative value to be minimized, otherwise, $E = 0$. This inequality-QUBO transformation separates the constraints from QUBO form, resulting in significant search space reduction in QUBO variable configurations, and offering the potential for reduced hardware overhead and improved problem solving efficiency.

3.3 FeFET-based Inequality Filter

Fig. 4(a) depicts the cell schematic of the FeFET-based CiM inequality filter array, comprising a single FeFET storing w_i and a series resistor R. The input variable x_i is applied at the gate. The cells within a word are connected to matchline (ML), with an additional capacitance C_{ML} connected to ML. A precharging PMOS is connected to ML. As depicted in Fig. 4(b), the cell's ON current is regulated by 1FeFET1R structure to reduce current variability [24, 25] shown in Fig. 2(b). A cell can store five different weights $w_i = 0$ to 4, with four corresponding V_{read} voltages. When a cell stores $w_i = k$, applying V_{readj} $(j \leq k)$ turns the cell ON, conducting I_{D}. Otherwise, the cell is OFF.

Fig. 4(c) illustrates the cell operation of a filter iteration. Firstly, ML is precharged to VDD (2V), followed by four phases. During phases 1 to 4, if $x_i = 0$, V_{G} is set to 0, and ML remains VDD. If $x_i = 1$, a staircase pulse is applied at V_{G}, with amplitude ranging from V_{read4} to V_{read1}. Only when V_{G} is applied with V_{readj} that $k \geq j$, can the cell turns ON, discharging ML. As a result, ML voltage V_{ML} after a filter iteration can be expressed as follows:

$$V_{\text{ML}} = VDD - k \times \frac{\int I_{\text{D}} t \, dt}{C_{\text{ML}}} \times x_i \tag{7}$$

By appropriately choosing C_{ML} and VDD, $\frac{\int I_{\text{D}} t \, dt}{C_{\text{ML}}}$ can be approximately constant. Consequently, when $x_i = 1$, V_{ML} exhibits a linear relationship with k, which is also validated by simulation as shown in Fig. 4(c). Notably, Eq. (7) is equivalent as below:

$$\text{ML} \propto -w_i x_i \tag{8}$$

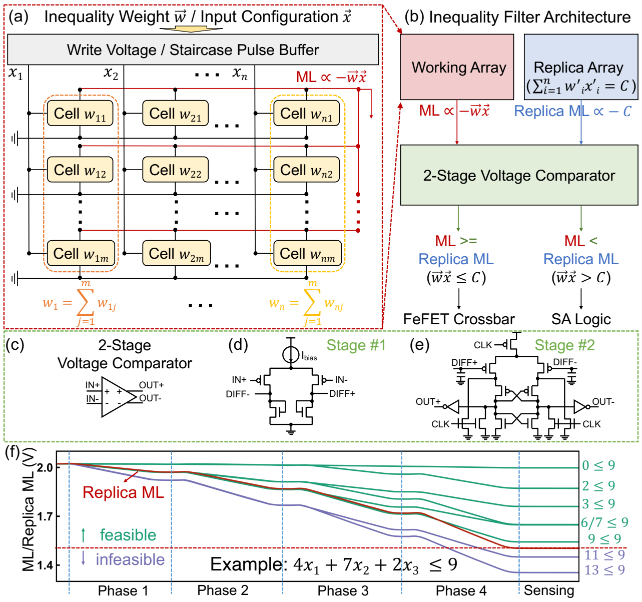

Figure 5: (a) Schematic of an $m \times n$ working array. The output ML ∝ $-\vec{w}\vec{x}$; (b) The inequality filter architecture, containing one working array, one replica array and a voltage comparator; (c) Symbol of 2-stage voltage comparator; (d) First-stage differential pre-amplifier; (e) second-stage dynamic latched voltage comparator; (f) Transient waveforms of an exemplary inequality evaluation. Infeasible configurations are filtered out.

Building upon Eq. (8), we introduce an $m \times n$ working array for inequality evaluation, as depicted in Fig. 5(a). In the array, cells within a column (e.g., column i) share an input signal x_i, and all MLs are interconnected. The precharge circuit and C_{ML} are not shown for simplicity. The item weight vector $\vec{w} = [w_1, w_2, ..., w_n]$ is stored horizontally across n columns. Suppose that each cell can store 0, 1, ..., k distinct weights. Accordingly, each item weight w_i is decomposed into multiple w_{ij} values, such that $w_i = \sum_{j=1}^{m} w_{ij}$, $w_{ij} \in \{0, 1, ..., k\}$, subsequently stored in the cells within the i^{th} column. Eq. (8) describes ML of a single column, and the summed ML of the $m \times n$ working array can be expressed as:

$$\text{ML} \propto -\sum_{i}^{n} w_i x_i \tag{9}$$

Fig. 5(b) shows the inequality filter, comprising a working array, a replica array, and a 2-stage voltage comparator. The replica array stores a precomputed weight vector $\vec{w'} = [w'_1, w'_2, ..., w'_n]$, with a fixed input configuration $\vec{x'}$ that satisfies $\sum_{i}^{n} w'_i x'_i = C$. The ML value of the replica can be expressed as follows:

$$\text{Replica ML} \propto -\sum_{i}^{n} w'_i x'_i = -\text{C} \tag{10}$$

The two array ML outputs are fed to the 2-stage voltage comparator [19] as shown in 5(c-e) to compare $\sum_{i}^{n} w_i x_i$ and C. If $\sum_{i}^{n} w_i x_i \le C$, indicating a feasible input variable configuration, the FeFET-based crossbar is activated to perform QUBO computation per Fig. 3. If $\sum_{i}^{n} w_i x_i > C$, indicating an infeasible configuration, a signal is transmitted to SA logic, initiating next SA iteration.

Fig. 5(f) illustrates an example where input variable configurations are evaluated and filtered for the inequality $4x_1 + 7x_2 + 2x_3 \le 9$. The waveforms validate all possible input configurations ($2^3 = 8$ cases). Following four phases, six MLs corresponding to feasible input configurations exhibit higher voltage than the Replica ML, while the other two MLs corresponding to infeasible input configurations drop below the Replica ML.

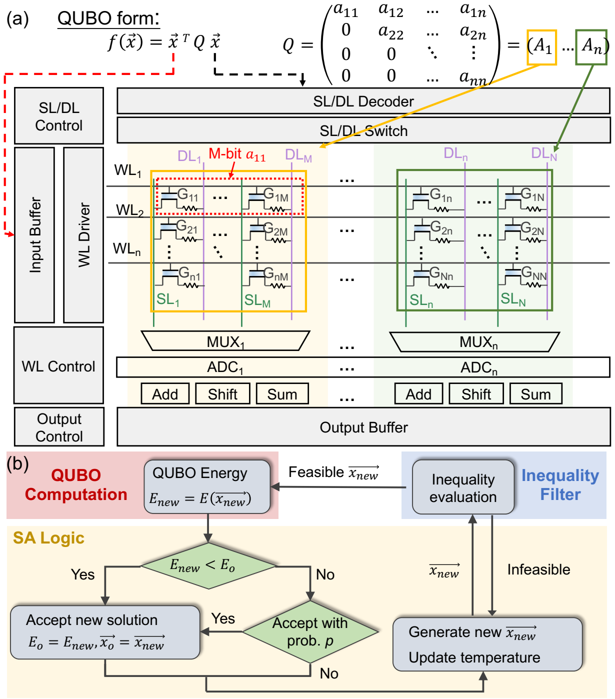

Figure 6: (a) The FeFET-based CiM crossbar array mapping the QUBO form; (b) The simulated annealing flow, including Inequality filtering, QUBO computation and SA logic.

3.4 FeFET-based Crossbar and SA

Fig. 6(a) shows our proposed FeFET-based CiM crossbar array that map a general QUBO form $\vec{x}^T Q \vec{x}$ (Eq. (2)) by storing matrix Q and applying input vector $\vec{x}$. Each column of the matrix Q, denoted as A_i, is mapped onto an $n \times M$ subarray, assuming M-bit matrix element quantization. Each 1FeFET1R cell stores 1 bit of the element. The feasible inputs $\vec{x}^T$ and $\vec{x}$ are simultaneously directed to the input buffer and SL/DL decoder, respectively. During each QUBO computation, the drains and gates of FeFET devices are applied with corresponding binary bits of input vector $\vec{x}$, and the column currents representing the partial vector-matrix-vector multiplications are sensed and accumulated together to get the QUBO form result.

Fig. 6(b) illustrates the SA flow, where the SA logic interacts with the crossbar and inequality filter. During each SA iteration, the SA logic generates a new input configuration $\overrightarrow{x_{\text{new}}}$, which is then forwarded to the inequality filter for inequality evaluation. If $\overrightarrow{x_{\text{new}}}$ is determined to be infeasible, a signal is sent back to the SA logic to initiate next SA iteration. If $\overrightarrow{x_{\text{new}}}$ is determined to be feasible, the inequality filter forwards $\overrightarrow{x_{\text{new}}}$ to the crossbar for QUBO computation. The resulting QUBO form output E_{new} is then directed back to the SA logic, which compares E_{new} with E_{o} (the reserved output of the previous iteration) and updates the optimal input configuration x_{o} and corresponding QUBO form output E_{o} per the comparison result and probability associated with annealing temperature.

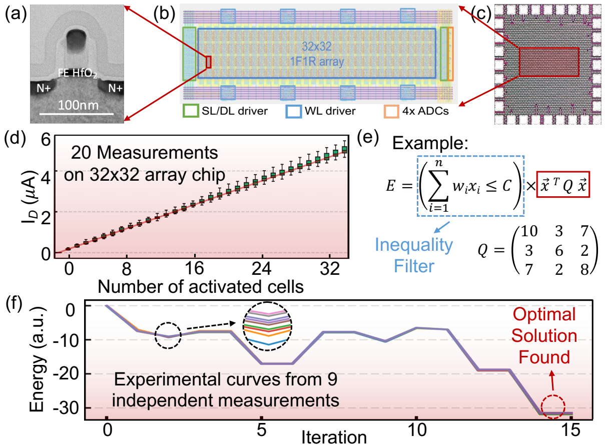

Figure 7: (a) TEM cross section of a FeFET; (b) Layout of the 32×32 FeFET array chip; (c) Die photo; (d) Experimental measurements validating a good linearity of the crossbar; (e) The inequality-QUBO form of a QKP example; (f) Experimental energy evolution demonstrating successful problem-solving.

Based on the structure in Fig. 6(a), we fabricated a 32×32 FeFET-based CiM array chip at 28nm high-k metal gate (HKMG) technology that implements the iterative QUBO computations. Fig. 7(a-c) demonstrate the transmission electron microscopy (TEM) cross section of a FeFET device, chip layout and the die photo, respectively. Fig. 7(d) experimentally validates the current linearity with the activated cell number within the 32×32 array chip. A QKP example shown in Fig. 7(e) is solved by HyCiM, with annealing process validated on the chip. Fig. 7(f) shows the energy evolution with SA iterations under 9 independent experimental measurements. For each measurement, the FeFET CiM array chip is erased and programmed again with the same QUBO matrix, and SA is executed. The curve results demonstrate that HyCiM successfully finds optimal solutions, affirming its capability and robustness in solving general COPs with inequality constraints.

4 VALIDATION AND EVALUATION

In this section, we first validate the functionality of our proposed FeFET-based inequality filter and evaluate the hardware overhead reduction achieved by HyCiM in comparison to D-QUBO based annealer. We then evaluate the efficiency improvements in solving COPs with HyCiM compared to others. All simulations were conducted using SPECTRE. The Preisach FeFET model [26] is adopted for FeFET, the 28nm TSMC model is used for MOSFETs with TT process corner at a temperature of 27°C. The wiring parasitics are extracted from DESTINY [27]. We choose QKP as the representative COP and select 40 instances from [28], each containing 100 items.

4.1 Validation of Inequality Filter

In our evaluation, both the working and replica arrays are of 16×100 size. Each array column stores an item weight, ranging from 0 to 64. As a result, the inequality filter effectively represents inequalities in the form of $\vec{w}\vec{x} \leq C$, $w_i \in \{0, 1, ..., 64\}$. For each inequality of 40 QKP instances, we generated 20 unique input configurations using Monte Carlo sampling method, including 10 feasible ones and 10 infeasible ones, totaling 800 cases for evaluation.

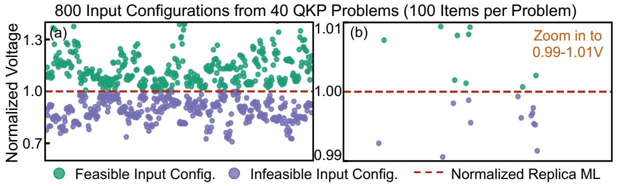

Figure 8: (a) Normalized ML outputs of the inequality filter classifying 800 input configurations for 40 QKP instances; (b) Closer view near the Normalized Replica ML.

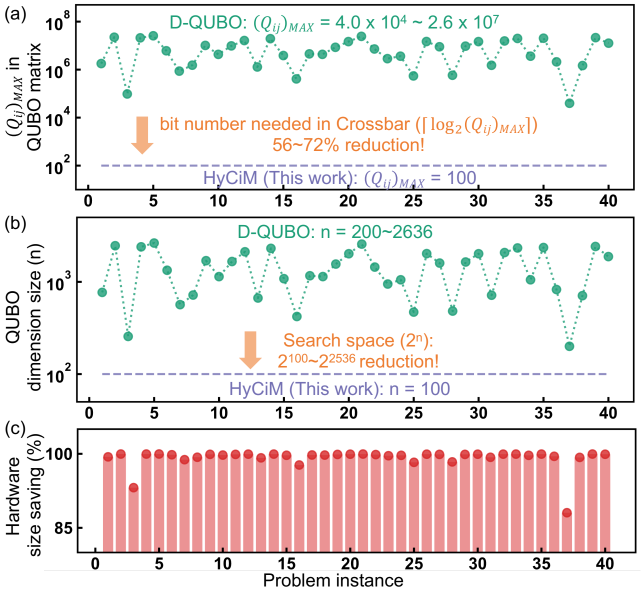

Figure 9: (a) The largest QUBO matrix element $(Q_{ij})_{MAX}$ in D-QUBO and HyCiM; (b) QUBO dimension size in D-QUBO and HyCiM; (c) Hardware size saving of HyCiM over D-QUBO.

Fig. 8(a) displays the normalized ML output of working array over replica array for the 800 cases, where green circles represent feasible configurations, and purple circles infeasible ones. Fig. 8(b) provides a closer view near the normalized replica ML. As can be seen, the inequality filter can clearly distinguish between feasible and infeasible configurations, whose corresponding ML outputs are distributed on the two sides of replica ML output, thus validating the effectiveness of the inequality filter.

4.2 Hardware Overhead

We compare our proposed HyCiM with D-QUBO approach in terms of hardware overhead. Both are implemented using the FeFET-based crossbar for fair comparison. Coefficients α and β in Fig. 1(b) are set to 2. Each cell in the crossbar stores 1-bit.

When mapping the QUBO matrix Q to crossbar, the quantization precision is determined by the largest matrix element, denoted as $(Q_{ij})_{\text{MAX}}$. Therefore, the crossbar needs $\lceil \log_2 (Q_{ij})_{\text{MAX}} \rceil$ bits to represent each element. In D-QUBO, the large-scaled auxiliary variable vector $\vec{y}$ introduces a large $(Q_{ij})_{\text{MAX}}$ value. As shown in Fig. 9(a), the largest matrix element in D-QUBO approach can reach up to 4-7 orders of magnitude, which corresponds to 16-25-bit quantization in the crossbar. On the contrary, HyCiM addresses the constraints using inequality filter, thus $(Q_{ij})_{\text{MAX}} = 100$, corresponding to 7-bit. Therefore, the required quantization bits are reduced by 56-72% in HyCiM compared to D-QUBO.

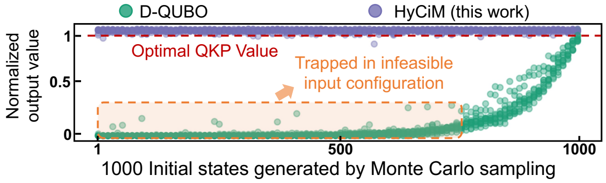

Figure 10: Normalized QKP values solved by D-QUBO based implementation and HyCiM.

Table 1: Summary of QUBO Solvers

Reference	[29]	[30]	[31]	[3]	[32]	This work
COP	Max-Cut	Spin Glass	Traveling Salesman	Graph Coloring	Knapsack	Quadratic Knapsack
Constraint	-	-	Equality	Equality	Inequality	Inequality
Search Space Reduction	No	No	No	No	No	Yes
COP to QUBO Transformation	D-QUBO	D-QUBO	D-QUBO	D-QUBO	D-QUBO	Inequality -QUBO
Hardware Imp.* for Crossbar	Memristor	RRAM	RRAM	FeFET	RRAM	FeFET
Problem Size	60 node	15 node	100 node	21 node	10 node	100 node
Average Success Rate (%)	$65^{\dagger}$	-	$31^{\dagger}$	-	$92.4^{\dagger}$	98.54

*: Imp. denotes for implementation. †: Extracted from literature.

In QUBO, the size of search space is 2^n, where n refers to the size of QUBO matrix dimension, i.e., the size of binary input variable configuration. Fig. 9(b) illustrates the sizes of QUBO matrix dimension n for both D-QUBO and HyCiM when processing 40 QKP problem instances. In D-QUBO shown in Fig. 1(b), n equals to the total size of $\vec{x}$ and $\vec{y}$, ranging from 200 to 2636, corresponding to a huge search space (2^{200} to 2^{2636}). On the contrary, in HyCiM, n is 100 since all instances contain 100 items, corresponding to a search space of 2^{100}. When compared to D-QUBO, HyCiM reduces the search space by 2^{100} to 2^{2536}, effectively preventing unnecessary QUBO computations and speeding up search space exploration.

When compared to D-QUBO, HyCiM not only reduces the required quantization precision to store the QUBO matrix, but also significantly reduces the size of QUBO matrix dimensions, thus saving huge hardware overhead. Fig. 9(c) illustrates the overall hardware size savings achieved by HyCiM (inequality filter and crossbar)over D-QUBO (crossbar only). The results show that HyCiM achieves hardware overhead reductions ranging from 88.06% to 99.96%, indicating improved energy efficiency and performance, making HyCiM an appealing solution for COP solving COPs.

4.3 Problem Solving Efficiency

We evaluate the QKP solving efficiency of D-QUBO based implementation and HyCiM using FeFET based CiM. We generate 1000 initial input configurations for each QKP instance using Monte Carlo sampling, and conduct 100 SA runs per initial configuration on both implementations to record the QKP values they can obtain. Each SA run executes 1000 iterations. Fig .10 shows the normalized QKP values found by both frameworks. The optimal QKP value is set as 95% of the true optimal value. It clearly shows that HyCiM consistently finds solutions around optimal QKP value, achieving a remarkable 98.54% success rate in average. Conversely, D-QUBO mostly gets trapped with infeasible input configurations during SA, resulting in poor QKP values and a 10.75% average success rate.

The solver summary in Table. 1 demonstrates that the proposed HyCiM can solve more complex and larger general COPs with high efficiency than prior works by leveraging the proposed inequality-QUBO transformation to reduce the search space.

5 CONCLUSION

In this paper, we introduce HyCiM, a hybrid CiM QUBO solver framework designed to efficiently solve COPs with inequality constraints. We present a novel transformation method that converts COPs with inequality constraints into an inequality-QUBO form. We propose a FeFET based inequality filter that evaluates inequalities and filters out infeasible inputs by leveraging the multi-level characteristic of FeFETs. We fabricate a FeFET based CiM crossbar that accelerates the QUBO computations by exploiting the single transistor multiplication property of FeFET devices as well as SA process. Evaluation results show that HyCiM offers substantial search space reduction, hardware overhead saving, and improved problem solving efficiency compared to prior QUBO solvers.

ACKNOWLEDGMENT

This work was partially supported by NSFC (62104213, 92164203) and SGC Cooperation Project (Grant No. M-0612).

REFERENCES

[1] V. T. Paschos, *Applications of combinatorial optimization.* John Wiley & Sons, 2014, vol. 3.
[2] M. Chang *et al.*, "An analog clock-free compute fabric base on continuous-time dynamical system for solving combinatorial optimization problems," in *IEEE CICC*, 2022, pp. 1–2.
[3] X. Yin *et al.*, "Ferroelectric compute-in-memory annealer for combinatorial optimization problems," *Nature Communications*, vol. 15, no. 1, p. 2419, 2024.
[4] N. Mohseni *et al.*, "Ising machines as hardware solvers of combinatorial optimization problems," *Nature Reviews Physics*, vol. 4, pp. 363–379, 2022.
[5] K. Katsuki *et al.*, "Fast solving complete 2000-node optimization using stochastic-computing simulated annealing," in *2022 ICECS.* IEEE, 2022, pp. 1–4.
[6] T. Takemoto *et al.*, "A 144kb annealing system composed of 9× 16kb annealing processor chips with scalable chip-to-chip connections for large-scale combinatorial optimization problems," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64. IEEE, 2021, pp. 64–66.
[7] W. Moy *et al.*, "A 1,968-node coupled ring oscillator circuit for combinatorial optimization problem solving," *Nature Electronics*, vol. 5, pp. 310–317, 2022.
[8] I. Ahmed *et al.*, "A probabilistic compute fabric based on coupled ring oscillators for solving combinatorial optimization problems," *IEEE JSSC*, vol. 56, pp. 2870–2880, 2021.
[9] F. Glover *et al.*, "A tutorial on formulating and using qubo models," *arXiv preprint arXiv:1811.11538*, 2018.
[10] R. A. Quintero *et al.*, "Characterizing and benchmarking qubo reformulations of the knapsack problem," Technical Report. Department of ISE, Lehigh University, Tech. Rep., 2021.
[11] L. Pusey-Nazzaro *et al.*, "Adiabatic quantum optimization fails to solve the knapsack problem," *arXiv preprint arXiv:2008.07456*, 2020.
[12] T. Bontekoe *et al.*, "Translating constraints into qubos for the quadratic knapsack problem," in *International Conference on Computational Science.* Springer, 2023, pp. 90–107.
[13] A. Mallick *et al.*, "Cmos-compatible ising machines built using bistable latches coupled through ferroelectric transistor arrays," *Scientific reports*, vol. 13, p. 1515, 2023.
[14] R. Afoakwa *et al.*, "Brim: bistable resistively-coupled ising machine," in *2021 IEEE International Symposium on High-Performance Computer Architecture (HPCA).* IEEE, 2021, pp. 749–760.
[15] 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.
[16] L. Zhao *et al.*, "Convfifo: A crossbar memory pim architecture for convnets featuring first-in-first-out dataflow," in *29th ASP-DAC.* IEEE, 2024, pp. 824–829.
[17] A. Mondal *et al.*, "In-situ stochastic training of mtj crossbar based neural networks," in *Proceedings of the International Symposium on Low Power Electronics and Design*, 2018, pp. 1–6.
[18] S. Salahuddin *et al.*, "The era of hyper-scaling in electronics," *Nature Electronics*, vol. 1, no. 8, pp. 442–450, 2018.
[19] C. Zhuo *et al.*, "Design of ultra-compact content addressable memory exploiting 1t-1mtj cell," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022.
[20] K. Ni *et al.*, "Ferroelectric ternary content-addressable memory for one-shot learning," *Nature Electronics*, vol. 2, no. 11, pp. 521–529, 2019.
[21] X. Yin *et al.*, "An ultracompact single-ferroelectric field-effect transistor binary and multibit associative search engine," *Advanced Intelligent Systems*, vol. 5, no. 7, p. 2200428, 2023.
[22] X. S. Hu *et al.*, "In-memory computing with associative memories: a cross-layer perspective," in *2021 IEEE IEDM*, 2021, pp. 25–2.
[23] X. Yin *et al.*, "Ferroelectric ternary content addressable memories for energy-efficient associative search," *IEEE TCAD*, vol. 42, no. 4, pp. 1099–1112, 2022.
[24] T. Soliman *et al.*, "Ultra-low power flexible precision fefet based analog in-memory computing," in *IEDM.* IEEE, 2020, pp. 29–2.
[25] D. Saito *et al.*, "Analog in-memory computing in fefet-based 1t1r array for edge ai applications," in *2021 Symposium on VLSI Technology.* IEEE, 2021, pp. 1–2.
[26] K. Ni *et al.*, "A circuit compatible accurate compact model for ferroelectric-fets," in *IEEE VLSI*, 2018, pp. 131–132.
[27] M. Poremba *et al.*, "Destiny: A tool for modeling emerging 3d nvm and edram caches," in *DATE.* EDA Consortium, 2015, pp. 1543–1546.
[28] "Quadratic Knapsack Problem dataset," http://cedric.cnam.fr/ soutif/QKP/QKP.html.
[29] F. Cai *et al.*, "Power-efficient combinatorial optimization using intrinsic noise in memristor hopfield neural networks," *Nature Electronics*, vol. 3, pp. 409–418, 2020.
[30] J. H. Shin *et al.*, "Hardware acceleration of simulated annealing of spin glass by rram crossbar array," in *2018 IEEE International Electron Devices Meeting (IEDM).* IEEE, 2018, pp. 3–3.
[31] M.-C. Hong *et al.*, "In-memory annealing unit (imau): Energy-efficient (2000 tops/w) combinatorial optimizer for solving travelling salesman problem," in *IEDM.* IEEE, 2021, pp. 21–3.
[32] K. Taoka *et al.*, "Simulated annealing algorithm & reram device co-optimization for computation-in-memory," in *2021 IEEE International Memory Workshop (IMW).* IEEE, 2021, pp. 1–4.