Enhanced Open-Source Scatter Search Algorithm for Solving QUBO Problems

Donghao Liu [a,†], Wei Yang [b,e,†], Hongpu Wang [b], Yu Du [c], Yang Wang [b,*], Zhipeng Lü [d], Jin-Kao Hao [e]

[a] *Cockrell School of Engineering, The University of Texas at Austin, 301 E Dean Keeton St, 78712 Austin, United States*

[b] *School of Management, Northwestern Polytechnical University,127 Youyi West Road, 710072 Xi'an, China*

[c] *Business School, University of Colorado Denver, 1201 Larimer St, 80204 Denver, United States*

[d] *Department of Computer Science, Huazhong University of Science & Technology, 1037 Luoyu Road, 430074 Wuhan, China*

[e] *LERIA, Université d'Angers, 2 Boulevard Lavoisier, 49045 Angers, France*

Computers and Operations Research, 2025

Abstract

In recent years, quantum computing has driven significant excitement and innovation, with the Quadratic Unconstrained Binary Optimization (QUBO) model at its core. This paper introduces SATPR, a new open-source metaheuristic algorithm that combines scatter search, adaptive tenure tabu search, and path-relinking. The adaptive nature of the tabu tenure, achieved by integrating different heuristic components, enables SATPR to effectively solve different types of QUBO problem instances. Additionally, SATPR utilizes parallelism to fully leverage multi-threading capabilities, further enhancing its computational efficiency. We conducted extensive evaluations on large and challenging problem instances from four benchmark sets, including well-known QUBO and Max-Cut instances, as well as less explored random graph structures. Our results show that SATPR is highly competitive in both solution quality and computational efficiency when compared to leading metaheuristic QUBO solvers and the quantum-inspired Fixstars Amplify Annealing Engine.

Keywords: QUBO; quantum computing; path relinking; tabu search; scatter search

† These authors contributed equally to this work.

* Corresponding author.

Email addresses: donghaoliu@utexas.edu (Donghao Liu), yangwei123@mail.nwpu.edu.cn (Wei Yang), yu.du@ucdenver.edu (Yu Du),

1 Introduction

In recent years, there has been a great deal of excitement about quantum computing, primarily driven by its ability to optimize the Ising spin glass model and its broader potential for addressing general combinatorial optimization problems, as outlined in Parekh (2023) and Abbas et al. (2023). At the heart of the excitement is the QUBO model, because of its equivalence to the spin glass model and its established capacity to function as a unified framework for combinatorial optimization, as documented in studies such as Kochenberger et al. (2004), Kochenberger et al. (2014), Lucas (2014) and Anthony et al. (2017), Aramon et al. (2019), Glover et al. (2022a) and Glover et al. (2022b). In this context, quantum computers distinguish themselves as one of the primary platforms for running QUBO formulations.

Mathematically, the QUBO model is represented by

$$\max_{x\in\{0,1\}^n} x^T Q x = \max_{x\in\{0,1\}^n} (\sum_{i=1}^{n} c_i \cdot x_i + \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij} \cdot x_i \cdot x_j)$$

where x is a vector of binary variables $x \in \{0,1\}^n$ and Q is a symmetric matrix of constants. Let c_i be the coefficients on the matrix Q diagonal, Let d_{ij} be the coefficients off the matrix Q diagonal. The equivalent Ising problems replace x vector by $x' \in \{-1,1\}^n$ and can be put in the QUBO form by defining $x_{j'} = (x_j + 1)/2$ and then redefining x_j to be $x_{j'}$. To put it simply, QUBO represents a quantum-ready modeling framework.

The QUBO solver is a powerful computational tool that can be used to address complex optimization problems in various domains. It belongs to the family of optimization techniques that deal with discrete variables, specifically binary variables, to find the optimal solutions for a given objective function. QUBO solvers have gained prominence due to their versatility and applicability in diverse fields and applications, ranging from operations research to machine learning and quantum computing. In particular, QUBO finds utility in classical combinatorial optimization problems, including but not limited to max cut, graph coloring, capital budgeting, task allocation, product distribution, number partitioning, max sat, clique partitioning, clustering, sat packing, assignment, and knapsack problems. Additionally, emerging applications extend into diverse fields such as machine learning, deep learning, biotechnology, supply chains, logistics, portfolio analysis, and various other domains, holding the potential to drive future advancements (see details in Glover et al. (2022a) and Glover et al. (2022b)).

To solve the QUBO model or the Ising spin glass model, there are well-known quantum annealing, quantum gate-circuit optimization algorithms, coherent ising machines,

`yangw@nwpu.edu.cn` (Yang Wang), `zhipeng.lv@hust.edu.cn` (Zhipeng Lü),
`jin-kao.hao@univ-angers.fr` (Jin-Kao Hao).

and other quantum inspired algorithms. Quantum annealing is a computational technique that leverages principles of quantum physics to solve optimization problems. It is different from universal quantum computing in that it's specifically designed to solve optimization problems by finding the lowest energy state (minimizing a cost function) of a given system. D-Wave Systems stands out as one of the prominent companies in the development of quantum annealers (Bunyk et al. , 2014). Their machines are designed to solve specific types of problems known as quadratic unconstrained binary optimization (QUBO) problems. These problems can be mapped onto the hardware of a quantum annealer, which then explores the potential optimal configurations, according to Boixo et al. (2014). Moreover, Fujitsu offers cloud services for solving QUBO models through Digital Annealers (Fujitsu , 2021). Fixstars Amplify Anealer Engine is a digital annealer running on a cluster of Graphics Processing Units (GPUs), with a capacity of 65,536 bits connected by a complete graph. It can run more complex models than D-Wave and Fujitsu because it has a complete connection graph topology between bits (Fisxtar Amplify , 2021) . Quantum Approximation Optimization Algorithm (QAOA) is a type of hybrid quantum-classical variational algorithm for solving combinatorial optimization problems, seeking approximate solutions. QAOA constructs an approximate solution by combining classical and quantum computation techniques proposed by Barak et al. (2015) and Farhi et al. (2014). But the hyperparameter search is difficult.

Metaheuristic algorithms are practically used to produce approximate solutions for large QUBO problem instances that cannot be solved exactly due to the high computational complexity (Kochenberger et al. , 2014; Punnen , 2022). The adaptive memory tabu search algorithm (AMTS) was designed by Glover et al. (1998), which innovatively uses recency and frequency information to affect the move selection. It is the first tabu search algorithm that outperforms the best exact and heuristic methods previously reported. Glover et al. (2010) presented a diversification-driven tabu search algorithm that alternates between a basic tabu search procedure and a memory-guided perturbation procedure. The perturbation operates on a set of elite solutions and favors moves with low flipping frequency and high consistency. The path relinking (PR) algorithm was proposed by Wang et al. (2012), which is composed of a reference set initialization method, a solution improvement method, a reference set update method, a relinking method and a path solution selection method. The PR algorithm showed a high performance when solving extensive instance sets including QUBO, Max-Cut and minimum sum coloring problems. Lately, An automatic algorithmic selection approach was proposed by Dunning et al. (2018). It generates a random forest model to predict the probability of a heuristic among 37 Max-Cut and QUBO heuristics that will perform the best for a given problem instance and constructs an algorithm portfolio to select a set of heuristics with the highest predicted probability. Another automatic algorithm by combining problem-specific heuristic components was designed by Souza and Ritt (2018). This automatic algorithm employs a grammar in Backus-Naur form to model the space of heuristic strategies categorized into construction methods, search methods and recombination methods and their parameters. Recently, Hanafi et al. (2023) introduced a novel alternating ascent search framework. This framework incorporates an innovative adaptive memory, employing exponential extrapolation to prevent selected moves from reaching a specified number of recently encountered local optima. The emphasis is on the basic version of the Tabu Search metaheuristic, leveraging local optimality in binary optimization.

In this paper, we introduce a novel metaheuristic algorithm named SATPR, designed to serve as a bridge between operations research specialists and quantum or quantum-inspired optimization. Our objective is to develop an enhanced tabu search algorithm capable of effectively addressing challenging problem instances with diverse structures. In contrast to recent high-performance QUBO solvers that achieve outstanding results through the construction of a portfolio comprising state-of-the-art algorithms from the literature (Dunning et al., 2018; Souza and Ritt , 2018), our approach benefits from a novel scatter search structure, that successfully combines adaptive tenure tabu search and path-relinking (Glover et al. , 2003, 2004; Marti et al. , 2006; Molina et al. , 2007) in parallelism. Beyond tailored designs for the QUBO problem, we also develop new scatter search strategies that are applicable to solve various combinatorial optimization problems. Our main contributions are summarized as follows:

- We introduce a two-phase reference set initialization method that combines scatter search and tabu search to balance solution quality and diversity. The first phase involves a large population of initial solutions, each with a randomized tabu tenure, performing a short run of tabu search. In the second phase, a small population of initial solutions, each with a dispersed tabu tenure around the best-found solution, undergoes a longer run of tabu search.
- We propose a novel method for updating the reference set E when a pool of solutions P is obtained after executing tabu search in parallel. This method utilizes a weighted measure of solution quality and diversity, sequentially evaluating each newly generated solution. By adopting this approach, the method avoids expensive calculations on the distance between each member within the set $E+P$ and achieves a well-balanced trade-off between intensification and diversification compared to classical implementations.
- An innovative tabu tenure adaptation strategy is introduced to address the challenges associated with the time-consuming fine-tuning procedure and the typical tuning approach that is tailored to fit specific instances but may not generalize well to new instances. In essence, our tabu search method determines the best tenure by effectively utilizing and learning information gathered from the paralleled scatter search framework.
- We conduct systematic experiments to evaluate both solution quality and computational time across four sets of instances, comparing our algorithm with five best performing algorithms in the literature. These experiments include QUBO instances of varying properties and sizes, with a particular emphasis on challenging cases to demonstrate the robustness and efficiency of our approach. Additionally, we perform comparative tests to analyze the novel components of SATPR, validating the effectiveness and improvements introduced by these new features.

The remainder of this paper is structured as follows: Section 2 delves into the details of our SATPR algorithm. In Section 3, we conduct comprehensive assessments on large-scale problem instances to evaluate the performance of SATPR, comparing the results with state-of-the-art metaheuristic algorithms and Fixstars Amplify Digital Annealer. The effectiveness of the major novel features in the algorithm is also analyzed. Finally, Section 4 summarizes our findings, draws conclusions, and outlines potential avenues for future research.

2 SATPR Algorithm

Our proposed SATPR algorithm could be seen as an enhanced scatter search framework by combining adaptive tenure tabu search and path relinking. In particular, we integrate the diversification generation component with adaptive tabu search to produce an initial reference set with good quality and diversity. In the following subsections, we present the main scheme of SATPR and the involved search components.

2.1 Main Scheme of SATPR Algorithm

We show the main scheme of SATPR in Algorithm 1, including the initial reference set generation, reference set update, subset generation, solution combination, and solution improvement methods.

We design a two-phase method to obtain the initial reference set equipped with high-quality and well-diversified solutions. In the first phase (Lines 3 to 5), p^0 tabu tenures (or simply tenures) are initiated to perform tabu search for a large population of randomized solutions P^r. The random tenures are bounded by an upper limit u^r. Line 6 selects the tenure of the current best solution t^*. In the second phase, shown in Lines 7 to 10, we generate a new small-sized population P^f by using p^1 tenures scattered around the best tabu tenure t^* with radius d. These centralized tenures further improve the best random tenure and settle down some good solutions to construct the reference set. We note that the first phase uses fewer tabu search iterations but a much larger population size compared to the second phase. Hence, the first phase pays more attention to diversification, while the second phase focuses more on intensification.

After producing the initial reference set, SATPR performs scatter search iterations as described in Lines 12 to 19. During each iteration, pair-finding in the reference set is first used to generate subsets. Then, the solution combination, tabu search-based solution improvement, and reference set update methods are performed sequentially. The reference set reconstruction is a simple procedure that retains information about the best solution and regenerates other members of the reference set. This procedure is

Algorithm 1 Main Scheme

Input: Random solutions $P^r = \{x^1, ..., x^{p^0}\}$ with objective values V^r and random tenures $T^r = \{t^1, ..., t^{p^0}\}$, reference set $E = \emptyset$ with objective values V^E and tenure T^E of size $|E|$, 2^{nd} phase tenure diameter d
Output: The best solution x^*
for $i = 1$ to p^0 **in parallel do**
  $(P^r_i, V^r_i) \leftarrow \textsc{TabuSearch}(P^r_i, T^r_i)$
$\textsc{RefSetUpdate}(P^r, V^r, T^r, E, V^E, T^E)$
Set $t^* \leftarrow T^E_{e^*}; e^* : E_{e^*} = \max_e E_e$
Generate tenures $T^f = \{t^1, ..., t^{p^1}\}$ where $\forall t^k \in T^f : t^k = t^* - \frac{d}{2} + k \cdot \frac{d}{p^1}$, random solutions $P^f = \{x^1, ..., x^{p^1}\}$ and objective values V^f
for $i = 1$ to p^1 **in parallel do**
  $(P^f_i, V^f_i) \leftarrow \textsc{TabuSearch}(P^f_i, T^f_i)$
$\textsc{RefSetUpdate}(P^f, V^f, T^f, E, V^E, T^E)$
$S \leftarrow \{1\}^{|E|}$
while stop criterion is not satisfied **do**
  $A \leftarrow \textsc{PairFind}(E, S)$; $P \leftarrow \emptyset$; $V \leftarrow \emptyset$; $T \leftarrow \emptyset$
  for $i = 1$ to $|A|$ **in parallel do**
    $(x, t) \leftarrow \textsc{PathRelink}(E, V^E, T^E, A_i)$
    $(P_i, V_i) \leftarrow \textsc{TabuSearch}(x, t)$; $T_i \leftarrow t$
  $S \leftarrow \textsc{RefSetUpdate}(P, V, T, E, V^E, T^E)$
  if $S = \{0\}^{|E|}$ **then**
    $S \leftarrow \{1\}^{|E|}$; $\textsc{RefSetRecontruct}(E, V^E, T^E)$
return $x^* \leftarrow \max_e E_e$

invoked when the scatter search fails to improve the reference set in one iteration. It is important to note that the tenure is updated through the combination method in each round of scatter search.

The design of this adaptive tenure addresses a significant issue with tabu search—its sensitivity to tenure, which necessitates tuning and can vary significantly across different problem instances. Specifically, "adaptive tenure" refers to the mechanism that dynamically adjusts the duration for which moves remain in the tabu list based on the evolving state of the search process. This adaptation is refined through the combination method in each round of scatter search, which evaluates the performance of various tenures and selects the most effective one. When solving QUBO, tenures can differ based on the problem structure and matrix density. The adaptive tenure will automatically adjust the tabu search to its optimal configuration for a given instance, thus improving the robustness and efficiency of the algorithm.

Moreover, parallelism is implemented during all the SATPR search procedures using OpenMP (OpenMP ARB , 2018). In the reference set initialization phase, the individual solutions in the reference set are mutually exclusive and naturally lead themselves to parallel processing. Similarly, in the subsequent reference set update phase, all candidates can be treated simultaneously. OpenMP creates individual threads to handle each substructure independently, without interfering with each other. Each thread takes on an initial solution and performs tabu search in parallel. Afterward, all threads are paused until they complete their processing because the unified update procedure is then invoked before the next round of parallelism. The number of threads used and the size of the reference set can significantly impact the algorithm's performance.

2.2 Tabu Search based Solution Improvement

Algorithm 2 Solution Improvement

function TabuSearch(x, t)
    Input: Inital solution $x \in \{0,1\}^{|N|}$, tabu tenure t
    Output: Improved solution $x^* \in \{0,1\}^{|N|}$, improved value v^B
    $it \leftarrow 0$; $tlist \leftarrow \{0\}^{|N|}$; $v^B \leftarrow x$'s objective value
    for $it^n = 0$ to IT^{max} **do**
        for $j \in N$ **do**
            if $tlist_j < iter$ **then**
                if Flipping j updates the best non-tabued solution **then**
                    Select j as the best non-tabued solution
            else
                if Flipping j updates the best solution v^B **then**
                    Select j as the best aspiration solution
            Pick the best operation flipping j^* from aspiration or non-tabued
            Set $x_{j^*} \leftarrow 1 - x_{j^*}$; $tlist_{j^*} \leftarrow iter + t + rand()\%10$; $iter \leftarrow iter + 1$
            $x^* \leftarrow x$; $it^n \leftarrow 0$ **if** v^B has been updated

Tabu search is one of the most important procedures for handling large scale QUBO problem instances (Wang et al. , 2012; Kochenberger et al. , 2013; Dunning et al., 2018). Hence, our SATPR algorithm also adopts tabu search to perform solution improvement. However, the performance of tabu search depends on fine-tuned tabu tenures. For the unified QUBO model, different tenures are actually used for solving the QUBO and Max-Cut instances due to distinct problem structures. An adaptive setting of tabu tenure is especially important for the unified QUBO model since a variety of optimization problems can be reformulated as QUBO. For this purpose, our tabu search method attains the best tenure by utilizing and learning information collected from the scatter search framework. In this way, SATPR is able to work effectively without spending effort on preliminary experiments as in previous studies.

Recall that tabu search is applied for both randomized solutions in the initial reference set generation and the solutions generated from the solution combination. For the former case, we first use a large number of random tenures to perform a sequence of short tabu search trials to coarsely fetch a reasonable tenure from the best-found solution. Then, we set a fewer number of fine-grained tenures around this referred tenure to perform a sequence of long tabu search trials. Meantime, each solution in the reference set along with the used tenure is recorded. For the latter case, the tabu tenure is simply set as the average of the tabu tenures of the selected solutions in the solution combination.

The tabu search based improvement method employs the classical 1-flip move operator for solving QUBO problems, which consists of flipping a single binary variable at each iteration. The tabu list is constructed based on the position of this flipping operation. By maintaining a list that records the objective difference when a flip is selected, Lines 5 to 15 in Algorithm 2 can be executed efficiently. This list is incrementally updated once the best operation is performed. It's important to note that the aspiration criterion, meaning that a tabu operation will still be taken if it updates the best recorded solution, is also implemented in tabu search. It's noteworthy that by using an adjacency list data structure to store matrix information and perform incremental maintenance, the complexity for each tabu search iteration will be kept within the number of non-zero indices of the Q matrix. The described tabu search yields satisfactory results when the tenure is adaptively converged to a reasonable range for a specific instance. The simple structure and quadratic nature of QUBO facilitate the implementation of the flipping operation and the construction of the straightforward tabu list. IT^{max}, the maximum number of iterations for non-improving rounds, also known as improvement cutoff, will determine when to terminate the tabu search procedure. To speed up the initial phase, a smaller IT^{max} will be used for randomly generated tenures to quickly scout those desirable values.

2.3 Path-Relinking inspired Solution Combination

The subset generation method follows the traditional pair-finding procedure that essentially selects all pairs of candidates from the reference set for solution combination. Following the reference set update procedure, a list S indicating the update state of the reference set members is constructed. In each round, the pairs to be combined are selected from all the newly generated reference set members and all possible combinations between new members and old members. The pair-finding procedure leads to a thorough inspection of all new combinations in parallel without revisiting any of the old solutions.

For each generated subset, the solution combination method employs the pair of reference solutions to create new solutions. Given that path linking has been found to be quite effective for generating promising solutions when solving challenging QUBO problem instances (Wang et al. , 2012), we propose a path relinking inspired solution combination method, as outlined in Algorithm 3. Unlike the previous path relinking method that starts with one of the two selected solutions, we employ the simultaneous relinking approach starting with both solutions. Specifically, it initially identifies the common index set, denoted as j^s, from the two selected solutions and alternately inherits the

Algorithm 3 Solution Combination

function PATHRELINKING($E, V^E, T^E, \langle e, e' \rangle$)
  Input: Reference set E, elite tenure T^E, elite value V^E, combination index $\langle e, e' \rangle$
  Output: Result solution x^*, result tenure t^*
  $x^1 \leftarrow E_e$; $x^2 \leftarrow E_{e'}$;$v^1 \leftarrow V^E_e$; $v^2 \leftarrow V^E_{e'}$; $t^1 \leftarrow T^E_e$; $t^2 \leftarrow T^E_{e'}$; $J \leftarrow N$; $x^* \leftarrow \{0\}^N$; $v^* \leftarrow 0$;
  $\forall j^s \in N : x^1_{j^s} = x^2_{j^s}$; $J \leftarrow J \backslash \{j^s\}$
  $b \leftarrow rand(0, \alpha|J|)$
  while $|J| \geq b$ **do**
    if $|J| \% 2 = 0$ **then**
      Pick the best remaining index j^r for x^1 to flip, $x^1_{j^r} \leftarrow 1 - x^1_{j^r}$; Update v^1
    else
      Pick the best remaining index j^r for x^2 to flip, $x^2_{j^r} \leftarrow 1 - x^2_{j^r}$; Update v^2
    if $|J| \leq \alpha|J|$ **then**
      Update x^* to x^1 if $v^* > v^1$; Update x^* to x^2 if $v^* > v^2$
    $J \leftarrow J \backslash \{j^r\}$
  $t^* \leftarrow (t^1 + t^2)/2$

indices that differ in the two solutions to gradually converge to a new solution. In particular, starting from the reference solutions x^1 and x^2, the uncommon index set $J = N - \{j^s\}$ of them is analyzed iteratively. For solution x^1, an index j^r is selected such that $x^1_{j^r} \neq x^2_{j^r}$, and flipping j^r will create the maximum objective increment. Subsequently, j^r is removed from J, and another j^r is selected based on the maximum objective increment if $x^2_{j^r}$ is updated to $1 - x^2_{j^r}$. The combination method will also produce a new tabu tenure for this offspring solution by taking the average of those of the reference solutions.

This path-relinking inspired combination method not only incorporates the favorable aspects of the pair of reference solutions but also introduces diversification based on the uncommon parts. The alternating adoption from each reference solution is terminated by a randomly generated parameter b, ranging from 0 to α times the uncommon indices. α is a distance scale parameter that controls the minimum distance of the newly generated solution to each reference solution. This ensures that the resulting solution will be at least α distance away from either of the reference solutions. Additionally, it guarantees a high-quality solution by consistently choosing the index with the maximum objective increment and selecting the best solution along the relinking path.

2.4 Reference Set Update Method

The reference set update method, as shown in Algorithm 4, aims to maintain a set of elite solutions with a good objective value and high diversity. It is triggered each time a population of solutions is generated, which occurs during the reference set initialization

Algorithm 4 Reference Set Update

function RefSetUpdate(P, V, T, E, V^E, T^E)
  Input: New population P with value V and tenure T, reference set E with value V^E and tenure T^E
  Output: Update state $S \in \{0,1\}^{|E|}$
  Calculate the distance between each member e in E and the reference set E, i.e., $d_e \leftarrow \min_{e' \neq e} \|E_e - E_{e'}\|$
  for $k \in P$ **do**
    Calculate the distance between the solution P_k and the reference set E, i.e., $d^*_{E+1} \leftarrow \min_{e} \|E_e - P_k\|$; $d^*_e \leftarrow min(d_e, \|E_e - P_k\|)$
    $d^{min} \leftarrow \min_{e \in E+1} d^*_e$; $d^{max} \leftarrow \max_{e \in E+1} d^*_e$
    $v^{min} \leftarrow min(V_k, \min_{e} V^E_e)$; $v^{max} \leftarrow max(V_k, \max_{e} V^E_e)$
    for $e \in E$ **do**
      $s_e \leftarrow \beta \frac{d^{min}_e}{d^{max} - d^{min}} + (1-\beta) \frac{V^E_e}{v^{min} - v^{max}}$
    $s_{E+1} \leftarrow \beta \frac{d^{min}_{E+1}}{d^{max} - d^{min}} + (1-\beta) \frac{V_k}{v^{min} - v^{max}}$
    if $s_{E+1} > \min_{e} s_e$ **then**
      $e' \leftarrow e^* : s_{e^*} = \min_{e} s_e$; $E_{e'} \leftarrow P_k$; $V^E_{e'} \leftarrow V_k$; $T^E_{e'} \leftarrow T_k$; $S_{e'} \leftarrow 1$
      $\forall e, d_e \leftarrow d^*_e$; $d_{e'} = d^*_{E+1}$

and in each iteration of the scatter search. Unlike the update strategy covered in past literature (Marti et al. , 2009; Sanchez et al. , 2015), this new strategy strives to maintain a reference set E when inserting new solutions into population P. This is different from other studies where only a single solution is inserted typically. By examining the new solutions iteratively, rather than collectively, the algorithm avoids expensive calculations on the distance between each member within the combined set $E + P$. The proposed strategy involves evaluating and potentially inserting each new solution from P into the reference set E one by one. This ensures that each solution is assessed individually for its contribution to the overall objective value and diversity, maintaining the quality of the reference set without the computational burden of handling the entire population simultaneously.

The specific procedure is described as follows. After obtaining a new population of solutions through the combination and improvement methods for each candidate pair of elite solutions, each solution in this population is sequentially assessed based on its objective value and its distance from the current reference set. The measures for a specific solution are initially standardized and then combined using a linear combination with a weight parameter β to derive a score. If the score of the new solution is higher than that of the member with the lowest score in the reference set, the reference set member is replaced. The standardized scoring rules, considering solution quality and diversity, contribute to maintaining a favorable reference set for the algorithm to enhance solution improvement. In coordination with the pair-finding procedure, each modified index is recorded and output upon evaluating each of the new solutions. It's worth noting that when the set remains unchanged afterward, a reconstruction of the reference set becomes important. The reconstruction procedure simply flips half of the variables randomly across all reference set members to explore local optima that are far from the current search area.

3 Numerical experiments

In this section, we first present the experimental protocols and provide computational comparisons with the leading metaheuristic solvers in the literature, as well as Quantum Digital Annealer Fixstars Amplify Engine. Then we perform experimental analysis to reveal the effectiveness of three novel features in the proposed algorithm, including the adaptive tabu tenure strategy, the new reference set update method, and the two-way path-relinking strategy.

3.1 Experimental protocols

To provide a comprehensive evaluation of our algorithms, we use the following four benchmark sets:

- Palubeckis Instances: This set includes 21 of the largest instances from Palubeckis, named p3000.1 through p7000.3, with problem sizes ranging from $n = 3000$ to 7000. These randomly generated instances are particularly challenging when the number of variables exceeds 5000.
- Max-Cut Instances: This set comprises the 17 largest instances derived from the Max-Cut problem, named G55 through G81, with variable sizes ranging from $n = 5000$ to 20000. These instances are generated using a machine-independent graph generator and include toroidal, planar, and random weighted graphs with weights of 1, 0, or -1. Both the Palubeckis and Max-Cut instances are widely used as QUBO benchmarks in the literature for testing algorithm performance.
- QPLIB Instances: This set contains 15 nonconvex quadratic binary instances from QPLIB, with sizes ranging from $n = 231$ to 1225.
- Culberson Instances: This set consists of 10 of the largest instances from the "culberson" type, selected from 3,296 instances generated by Dunning et al. (2018), with variable sizes ranging from $n = 4000$ to 5000.

In total, these 63 instances provide a comprehensive evaluation of algorithm performance.

Our proposed algorithm is programmed in C++ and compiled using GNU GCC 10.2.0 with a $-O3$ flag. We run experiments on a server cluster equipped with an Intel(R) Xeon(R) Gold 6226R (2.90GHz) processor and a Linux operating system. Each task is allocated with 10 threads, allowing a maximum of 10 parallel threads. For each instance, our SATPR algorithm was executed in 10 independent runs, with each run having a time limit of 1800 seconds. The source code of our SATPR solver will be publicly available upon the publication of the paper. We run the reference algorithms in the same environment and under the same stopping condition for a fair comparison.

A comprehensive description of all the parameters used in the proposed algorithm is provided in Table 1. The reference set size $|E|$, improvement cutoff IT^{max}, and distance scale α are acquired from Wang et al. (2012). The specific values of p^0, u^r, p^1, and d are determined through separate experiments conducted during the initial two phases of the main scheme. These experiments contribute to tailoring these parameters to the characteristics of the two phases. Additionally, the parameter β undergoes tuning across the entire algorithm, ensuring its adaptability and optimization for various instances. The comprehensive set of parameters and their individualized adjustments reflect the hybrid and adaptive nature of the proposed algorithm, making it well-suited for a diverse range of problem instances.

Table 1
Parameter settings

Parameters	Description	Section	Value
$\vert E\vert$	Reference set size	2.1	10
p^0	Population size of the first phase	2.1	$12 \cdot \vert E\vert$
u^r	Upper bound of random tenures	2.1	$0.2 \cdot n$
p^1	Population size of the second phase	2.1	$3 \cdot \vert E\vert$
d	Radius of scattering tenure	2.1	$0.05 \cdot n$
IT^{max}	Improvement cutoff	2.2	$5 \cdot n$ ($0.6 \cdot n$ for random tenures)
α	Distance scale	2.3	$\frac{1}{3}$
β	Scoring weight	2.4	0.9

3.2 Experimental comparison with leading metaheuristic solvers

We compare our SATPR algorithm against the following leading reference methods:

- CLPR19: The clustering-driven evolutionary algorithm for solving QUBO problems proposed by Samorani et al. (2019).
- Dunning: The open-source algorithm portfolio developed by Dunning et al. (2018), which uses a random forest model to select the best algorithm from 37 state-of-the-art heuristics based on the characteristics of each problem instance (Dunning et al.,

2018). This method intelligently integrates various heuristics published in the literature, making it a powerful QUBO solver.
- LU2010: The path relinking algorithm proposed by Lü et al. (2010), one of the top-ranked population-based evolutionary algorithms among the 37 heuristics.
- MERZ04: The algorithm proposed by Merz and Katayama (2004), another top-ranked population-based evolutionary algorithm selected from the 37 heuristics.

We will use these standard names consistently throughout the rest of this paper.

The summarized comparison results for the four sets of benchmarks are presented in Table 2, while Table 3 to 6 provide detailed results, including the best and average objective values and the average CPU running time. In Table 2, the first column lists the benchmarks. Columns *BestOvj* and *AvgObj* summarize the best and average objective values achieved across 10 independent runs. The columns "#Wins", "#Ties", and "#Losses" indicate the number of instances where our proposed algorithm performed better than, equal to, and worse than the reference algorithm, respectively.

The results in Table 2 clearly show that our proposed algorithm is highly competitive compared to the reference algorithms in terms of both best results and average results. Overall, our algorithm found better solutions for 21 out of 63 instances and matched the best solutions for 36 instances, and found worse solutions for 6 instances. Specifically, for the Palubeckis instances set, our algorithm matched the best-known solutions for all 21 instances. For the Max-Cut instances set, our algorithm found the better solutions for 13 out of 17 instances, and for the culberson instances set, it found the new best solutions for 8 out of 10 instances. In the QPLib set, all algorithms excepted for Dunning et al. (2018) matched the optimal solutions reported. It should be noted that this well-known QUBO dataset was not included in the training set of Dunning et al. (2018). This result suggests that the hyper-heuristic algorithm of Dunning et al. (2018) struggles with these untrained instance types, as both LU2010 and MERZ04, included in its repository, found the optimal solution for 15 instances. Similar performance trends were observed regarding the average objective value.

Tables 3 to 6 present the detailed comparison results obtained by the five algorithms. The best-known solution values (*BKS*) were reported in the second column. For the Palubeckis and Max-Cut instances sets, the *BKS* were directly taken from Goudet et al. (2024), in which the current *BKS* were reported. Goudet et al. (2024) proposed an evolutionary algorithm with a very large population organized in different islands with a population size of up to 64,000. The algorithm reported some new *BKS* results on Max-cut instances, but the solving time for large-scale instances reached as high as 66,600 seconds. Therefore, this algorithm was not included as a comparison in our study; only the *BKS* results they provided for the max-cut instances were used. For the QPLib instances set, the *BKS* were taken from Furini et al. (2019) which has been proved to be optimal. For the Culberson instances set, the *BKS* was first reported in this study since Dunning et al. (2018) does not include these instances in their paper.

The best objective value (column *Best*), the average objective value (column *AvgObj*), and the average time to best solution (column *AvT*) for each instance corresponding to each algorithm were reported respectively. From Tables 3 to 6, it can be seen that on the three instances sets Max-Cut, QPLib, and Culberson, our proposed algorithm can obtain better results in a shorter time. For the Palubeckis instances set, there is no obvious difference in computation time.

Table 2
Summarized comparison results of SATPR against reference algorithms on four instance sets.

Instances	Pair algorithms	*BestObj*			*AvgObj*		
		#Wins	#Ties	#Losses	#Wins	#Ties	#Losses
Palubeckis	SATPR vs. CLPR19	0	21	0	6	12	3
	SATPR vs. Duning	0	21	0	3	12	6
	SATPR vs. LU2010	1	20	0	6	12	3
	SATPR vs. MERZ04	3	18	0	14	7	0
	SATPR vs. Total	0	21	0	1	13	7
Max-Cut	SATPR vs. CLPR19	17	0	0	17	0	0
	SATPR vs. Duning	13	0	4	15	1	1
	SATPR vs. LU2010	17	0	0	17	0	0
	SATPR vs. MERZ04	17	0	0	17	0	0
	SATPR vs. Total	13	0	4	15	1	1
QPLib	SATPR vs. CLPR19	0	15	0	0	15	0
	SATPR vs. Duning	7	8	0	7	10	0
	SATPR vs. LU2010	0	15	0	2	13	0
	SATPR vs. MERZ04	0	15	0	3	12	0
	SATPR vs. Total	0	15	0	0	15	0
Culberson	SATPR vs. CLPR19	10	0	0	10	0	0
	SATPR vs. Duning	9	0	1	8	0	2
	SATPR vs. LU2010	9	0	1	9	0	1
	SATPR vs. MERZ04	10	0	0	10	0	0
	SATPR vs. Total	8	0	2	7	0	3

Table 3
Detailed comparison results on Palubeckis instances set

Instances	BKS	SATPR			CLPR19			Dunning			LU2010			MERZ04		
		Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT
3000.1	3931583	**3931583**	**3931583**	13.2	**3931583**	**3931583**	9.8	**3931583**	**3931583**	12.3	**3931583**	**3931583**	10.7	**3931583**	**3931583**	86.7
3000.2	5193073	**5193073**	**5193073**	15.5	**5193073**	**5193073**	7.7	**5193073**	**5193073**	17.7	**5193073**	**5193073**	7.8	**5193073**	**5193073**	18.3
3000.3	5111533	**5111533**	**5111533**	32.1	**5111533**	**5111533**	35.2	**5111533**	**5111533**	86.8	**5111533**	**5111533**	12	**5111533**	**5111533**	318.3
3000.4	5761822	**5761822**	**5761822**	26.7	**5761822**	**5761822**	16.5	**5761822**	**5761822**	32.6	**5761822**	**5761822**	15.8	**5761822**	**5761822**	117
3000.5	5675625	**5675625**	**5675625**	58.1	**5675625**	**5675625**	107	**5675625**	**5675625**	52.4	**5675625**	**5675625**	49.7	**5675625**	5675386	328.6
4000.1	6181830	**6181830**	**6181830**	16.2	**6181830**	**6181830**	18.6	**6181830**	**6181830**	19.8	**6181830**	**6181830**	10.8	**6181830**	**6181830**	39
4000.2	7801355	**7801355**	**7801355**	49.2	**7801355**	**7801355**	165.9	**7801355**	**7801355**	155.8	**7801355**	**7801355**	168.8	**7801355**	7801146	518.1
4000.3	7741685	**7741685**	**7741685**	45.6	**7741685**	**7741685**	71.5	**7741685**	**7741685**	57.8	**7741685**	**7741685**	27	**7741685**	7741652.6	120.2
4000.4	8711822	**8711822**	**8711822**	87.6	**8711822**	**8711822**	79.2	**8711822**	**8711822**	59.5	**8711822**	**8711822**	27.6	**8711822**	**8711822**	45.7
4000.5	8908979	**8908979**	**8908979**	81.2	**8908979**	**8908979**	154.3	**8908979**	**8908979**	194.2	**8908979**	**8908979**	199.5	**8908979**	**8908979**	191.8
5000.1	8559680	**8559680**	8559582.5	905	**8559680**	8559600.5	776.8	**8559680**	8559647.5	562.8	**8559680**	8559490.1	792.3	**8559680**	8559280.8	496.9
5000.2	10836019	**10836019**	**10836019**	759.1	**10836019**	**10836019**	334.3	**10836019**	**10836019**	151.9	**10836019**	**10836019**	157.1	**10836019**	10835346.2	213.1
5000.3	10489137	**10489137**	**10489137**	312.9	**10489137**	10489127.6	768.2	**10489137**	10489132.3	526.6	**10489137**	10489101.6	845	**10489137**	10488708.9	462.6
5000.4	12252318	**12252318**	12252147.2	725.7	**12252318**	12252193.5	850.1	**12252318**	12251938.1	899	**12252318**	12252091.2	1016	12251710	12250704.4	815.6
5000.5	12731803	**12731803**	**12731803**	248.3	**12731803**	**12731803**	234.3	**12731803**	12731790.4	463	**12731803**	**12731803**	186.5	**12731803**	12731085.5	147.7
6000.1	11384976	**11384976**	11384916.1	578.6	**11384976**	11384859.8	913.1	**11384976**	11384956.3	879.7	**11384976**	11384957.9	565.6	**11384976**	11384598.5	414.4
6000.2	14333855	**14333855**	14333842.2	665.9	**14333855**	14333792.2	1291	**14333855**	14333846.2	843.8	**14333855**	14333846.2	638.4	14333767	14332998.6	477.4
6000.3	16132915	**16132915**	16132792.8	836.8	**16132915**	16132778.5	1220.7	**16132915**	**16132915**	559	**16132915**	16132904.6	1271.8	**16132915**	16130553	727.2
7000.1	14478676	**14478676**	14478289.4	854	**14478676**	14478242.4	947.1	**14478676**	14478590.2	729.6	14478655	14478161.2	1244	**14478676**	14475963.5	702.7
7000.2	18249948	**18249948**	18249238.2	984.6	**18249948**	18249239.2	1230.1	**18249948**	18249380.5	944.1	**18249948**	18249113.8	1152.7	18249844	18246439.3	851.4
7000.3	20446407	**20446407**	**20446407**	319.3	**20446407**	20446253.4	832.8	**20446407**	**20446407**	373	**20446407**	20445397.6	1030.9	**20446407**	20443759.7	635.6

Table 4
Detailed comparison results on Gset instances set

Instances	BKS	SATPR			CLPR19			Dunning			LU2010			MERZ04		
		Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT
G55	10299	**10290**	**10284**	81.2	10269	10249.7	1613.2	10273	10260.7	1557.8	10213	10201.9	422.1	10235	10214.9	1477.7
G56	4017	**4013**	**4002.3**	282.5	3988	3966.9	1335.8	3989	3976.4	1451.1	3942	3922.6	365.1	3950	3936.1	1245.5
G57	3494	**3490**	**3484.2**	554.8	3462	3458	1380.4	3480	3474	668.2	3480	3474.4	1062.9	3446	3437.8	1270.2
G58	19293	19248	19235.4	162.6	19210	19191.5	1465.7	**19251**	**19237.2**	1218.2	19200	19189.9	540	19193	19172.4	1623.6
G59	6087	6051	**6038.6**	590.8	6046	6010.5	1516.8	**6053**	**6038.6**	1273.5	5999	5974.1	398.9	6012	5977	1456
G60	14190	**14176**	**14167.6**	277.5	14129	14106.3	1360.8	14145	14137.3	572	14090	14064.9	728.8	14096	14054.7	1655.2
G61	5798	**5778**	**5772.1**	271.6	5743	5717.7	1287.5	5755	5746.3	802.9	5684	5671.7	712.6	5678	5661.9	1439.5
G62	4870	**4866**	**4860.8**	595.1	4816	4809	1550.4	4846	4838.4	1702.4	4842	4833.8	1708.8	4794	4783.8	1370.3
G63	27045	26969	**26957.5**	122.7	26889	26869.9	1316	**26970**	26956	1161.6	26917	26900.9	1091.8	26900	26870.1	1477.7
G64	8751	8703	**8674.5**	123.2	8648	8613.4	1670.1	**8711**	8666.4	1423.7	8612	8577.5	848.7	8600	8575.5	1669.4
G65	5562	**5552**	**5547**	592.5	5494	5485.2	1505.4	5528	5516.6	1667.3	5518	5500.2	1739.5	5466	5456.8	1417.5
G66	6364	**6350**	**6341.2**	341.7	6274	6266.4	1596.9	6250	6237	667.9	6294	6273.2	1670.4	6240	6227	1232.1
G67	6950	**6936**	**6929.4**	735.2	6854	6844.2	1799.3	6824	6800.2	1441.4	6854	6839.6	1645.6	6830	6808	1277.4
G70	9591	**9556**	**9544.5**	792.8	9343	9323.5	1663.7	9539	9527.8	748.8	9504	9480.1	1758.5	9422	9398.8	1673.3
G72	7006	**6986**	**6980**	579.9	6896	6891.6	1621.6	6878	6858.2	1545.3	6906	6890	1686	6878	6865.6	1323.1
G77	9938	**9910**	**9899.8**	917.6	9780	9768.6	1465.4	9762	9750	803.9	9756	9741.2	1560.5	9742	9725.2	1405.1
G81	14048	**14014**	**13999.6**	1104.7	13824	13816.2	2047.6	13782	13772	730.3	13756	13724.8	1327.4	13776	13736.8	1655.5

Table 5
Detailed comparison results on QPLib instances set

Instances	BKS	SATPR			CLPR19			Dunning			LU2010			MERZ04		
		Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT
QPLIB_3506	478	**478**	**478**	0	**478**	**478**	0.1	**478**	**478**	0.9	**478**	**478**	0.3	**478**	**478**	0.2
QPLIB_3565	282	**282**	**282**	0	**282**	**282**	0	**282**	**282**	0.7	**282**	**282**	0.1	**282**	**282**	0
QPLIB_3642	1034	**1034**	**1034**	0.2	**1034**	**1034**	1.7	916	907.2	33.9	**1034**	**1034**	16.6	**1034**	1033.8	12.2
QPLIB_3650	922	**922**	**922**	0.3	**922**	**922**	1.5	828	815.6	31.3	**922**	**922**	28.5	**922**	**922**	9.1
QPLIB_3693	1154	**1154**	**1153.8**	4.6	**1154**	**1153.8**	15.2	1014	1005	33.2	**1154**	1152.6	23.5	**1154**	1151.2	9.1
QPLIB_3705	384	**384**	**384**	0	**384**	**384**	0	**384**	**384**	0.7	**384**	**384**	0.1	**384**	**384**	0
QPLIB_3706	682	**682**	**682**	0.1	**682**	**682**	0.4	632	625.6	24.1	**682**	**682**	1.6	**682**	**682**	2
QPLIB_3738	422	**422**	**422**	0	**422**	**422**	0.1	**422**	**422**	1.9	**422**	**422**	0.7	**422**	**422**	1.9
QPLIB_3745	334	**334**	**334**	0	**334**	**334**	0	**334**	**334**	0.8	**334**	**334**	0.1	**334**	**334**	0.1
QPLIB_3822	850	**850**	**850**	0.1	**850**	**850**	0.2	776	767.4	38.3	**850**	**850**	2.1	**850**	**850**	1.9
QPLIB_3832	554	**554**	**554**	0.1	**554**	**554**	0.2	**554**	**554**	2	**554**	**554**	7.7	**554**	**554**	1.7
QPLIB_3838	746	**746**	**746**	0.2	**746**	**746**	1.3	678	676.2	21.8	**746**	**746**	19.7	**746**	**746**	6.4
QPLIB_3850	1198	**1198**	**1198**	0.4	**1198**	**1198**	3.2	1040	1026.6	32.5	**1198**	1197.2	25.8	**1198**	1196.4	22.1
QPLIB_3852	234	**234**	**234**	0	**234**	**234**	0	**234**	**234**	0.7	**234**	**234**	0	**234**	**234**	0
QPLIB_3877	602	**602**	**602**	0.1	**602**	**602**	0.3	**602**	**602**	2.3	**602**	**602**	3.6	**602**	**602**	2.7

Table 6
Detailed comparison results on culberson instances set

Instances	BKS	SATPR			CLPR19			Dunning			LU2010			MERZ04		
		Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT	Best	AvgObj	AvgT
culberson26	14049.776	**14049.776**	**14013.967**	277.7	13989.41	13950.531	1504.8	13237.449	13191.333	1033.2	13949.27	13924.783	437.8	14020.74	13972.329	1663.1
culberson107	4281.57	**4281.57**	4273.494	794.6	4269.14	4265.094	1503.3	4278.51	**4275.141**	877	4250.511	4241.332	681.4	4268.476	4259.242	1713.9
culberson15	60570.791	**60570.791**	**60505.236**	258.3	60460.63	60374.362	1310	58406.789	58342.968	963	60142.641	60054.136	393.6	60326.749	60273.895	1657.4
culberson85	12479180.12	12478573.96	12476446.98	1106.2	12474012.77	12471101.19	1595.1	12477814.2	12474935.47	987.5	**12479180.12**	**12477044.96**	1264.3	12477793.51	12473275.84	1063.7
culberson93	444592	**444592**	**444522.9**	563	444361	444235.6	1708	444569	444450.9	1125.4	444518	444468.7	799.9	444498	444365.3	908
culberson74	6637.796	6631.927	6618.997	521.3	6615.47	6605.327	1282.3	**6637.796**	**6632.715**	747.4	6579.478	6566.951	461.4	6617.221	6604.728	1684.4
culberson23	195433.075	**195433.075**	**195317.88**	1308.6	195292.5	195092.085	1381.2	194090.243	193845.212	842	194749.242	194548.574	898.9	195198.133	194960.2	1421.2
culberson30	141397	**141397**	**141184.8**	276.2	141031	140839.7	1478.2	141179	141019.6	934.7	140490	140317.5	387.5	141019	140798.4	1377.9
culberson91	654236.086	**654236.086**	**653713.372**	112.5	653121.62	652341.357	1493.6	653786.109	653336.424	797.8	650432.547	649670.678	989.5	653272.995	652002.222	1139.1
culberson68	1043	**1043**	**1039.8**	80.5	1006	994.9	1439.3	1025	968.6	1587.4	963	932.6	174	994	934.3	1353.5

Our initial aim was to compare our algorithm with the D-Wave quantum computer. However, due to the large size of the tested instances, the D-Wave Advantage System's architecture cannot handle these instances. As a result, we turned to the Fixstars Amplify Annealing Engine (Fisxtar Amplify , 2021), which can run more complex models and more variables than D-Wave because it has a complete connection graph topology between bits (i.e., QUBO variables). Fixstars Amplify offers a free software development kit (SDK) for evaluation and testing, with a QUBO scale limit of $n = 16,000$ and a time limit of 100 seconds on a V100 GPU. Since Fixstars Amplify can only deal with integer inputs and cannot return results for the Culberson datasets, we compared the two algorithms using the Palubeckis, Max-Cut and QPLib instance sets. For large-scale Palubeckis and Max-Cut instance sets, the time limit was set to 100 seconds for Fixstars Amplify. For the QPLib instance set, the time limit was reduced to 5 seconds, as our SATPR consistently found the best objective value with an average runtime of no more than 5 seconds.

Table 7 presents the detailed comparison results of our algorithm and Fixstars Amplify on the Palubeckis, Max-Cut and QPLib instance sets. Since Fixstars Amplify returns a solution without specifying the time taken to find the best solution, we do not report its computation time. For Palubeckis instances, we observe that our proposed algorithm yields highly competitive results compared to Fixstars Amplify in terms of the best and average objective values. Specifically, both algorithms achieve the best-known solutions. In terms of the average solutions, our algorithm matches the best solutions with Fixstars on 13 instances, performs better on 3 instances, and worse on 5 instances.

A similar performance pattern is observed for Max-Cut instances. Our proposed algorithm achieves better results for 9 out of 17 instances, while Fixstars Amplify performs better on the remaining 8 instances in terms of the best objective values. In terms of the average objective values, our algorithm achieves better results for 8 instances and worse results for 9 instances. Regarding QPLib instances, both algorithms reach the best-known solutions and produce identical average objective values across all 15 instances.

The effectiveness of our proposed algorithm is clearly demonstrated by the comparisons with quantum computing algorithm on these three instance sets. It's worth noting that Fixstars Amplify is running on superior hardware with a V100 GPU, while our algorithm runs only on a CPU. Despite this, the performance is similar.

Table 7
Experimental comparison between SATPR and Fixstars Amplify on three instance sets

Instances	SATPR		Fixstars Amplify	
	Best	AvgObj	Best	AvgObj
3000.1	**3931583**	**3931583**	**3931583**	**3931583**
3000.2	**5193073**	**5193073**	**5193073**	**5193073**
3000.3	**5111533**	**5111533**	**5111533**	**5111533**
3000.4	**5761822**	**5761822**	**5761822**	**5761822**
3000.5	**5675625**	**5675625**	**5675625**	**5675625**
4000.1	**6181830**	**6181830**	**6181830**	**6181830**
4000.2	**7801355**	**7801355**	**7801355**	**7801355**
4000.3	**7741685**	**7741685**	**7741685**	**7741685**
4000.4	**8711822**	**8711822**	**8711822**	**8711822**
4000.5	**8908979**	**8908979**	**8908979**	**8908979**
5000.1	**8559680**	**8559582.5**	**8559680**	8559411.4
5000.2	**10836019**	**10836019**	**10836019**	10835930
5000.3	**10489137**	**10489137**	**10489137**	**10489137**
5000.4	**12252318**	**12252147.2**	**12252318**	12251978.8
5000.5	**12731803**	**12731803**	**12731803**	**12731803**
6000.1	**11384976**	11384916.1	**11384976**	**11384976**
6000.2	**14333855**	14333842.2	**14333855**	**14333846.2**
6000.3	**16132915**	16132792.8	**16132915**	**16132915**
7000.1	**14478676**	14478289.4	**14478676**	**14478676**
7000.2	**18249948**	18249238.2	**18249948**	**18249941.5**
7000.3	**20446407**	**20446407**	**20446407**	**20446407**
G55	10290	10284	**10294**	**10291.8**
G56	4013	4002.3	**4016**	**4011.5**
G57	**3490**	**3484.2**	3486	3482.4
G58	19248	19235.4	**19251**	**19243.3**
G59	6051	6038.6	**6073**	**6060.8**
G60	**14176**	14167.6	14175	**14169.9**
G61	5778	5772.1	**5796**	**5783.9**
G62	**4866**	**4860.8**	4852	4845.6
G63	26969	26957.5	**26975**	**26964.8**
G64	8703	8674.5	**8744**	**8698.2**
G65	**5552**	**5547**	5538	5531.2
G66	**6350**	**6341.2**	6334	6322.4
G67	**6936**	**6929.4**	6914	6906.8
G70	9556	9544.5	**9582**	**9573.6**
G72	**6986**	**6980**	6970	6961.6
G77	**9910**	**9899.8**	9856	9841.6
G81	**14014**	**13999.6**	-	-
QPLIB_3506	**478**	**478**	**478**	**478**
QPLIB_3565	**282**	**282**	**282**	**282**
QPLIB_3642	**1034**	**1034**	**1034**	**1034**
QPLIB_3650	**922**	**922**	**922**	**922**
QPLIB_3693	**1154**	**1153.8**	**1154**	**1153.8**
QPLIB_3705	**384**	**384**	**384**	**384**
QPLIB_3706	**682**	**682**	**682**	**682**
QPLIB_3738	**422**	**422**	**422**	**422**
QPLIB_3745	**334**	**334**	**334**	**334**
QPLIB_3822	**850**	**850**	**850**	**850**
QPLIB_3832	**554**	**554**	**554**	**554**
QPLIB_3838	**746**	**746**	**746**	**746**
QPLIB_3850	**1198**	**1198**	**1198**	**1198**
QPLIB_3852	**234**	**234**	**234**	**234**
QPLIB_3877	**602**	**602**	**602**	**602**

3.4 Effectiveness of the Adaptive Tabu Tenure Strategy

Tabu tenure is a crucial parameter in tabu search algorithms and is typically tuned to fit specific instances. However, due to the nature of QUBO problems, where instances may vary significantly, the tuning procedure is time-consuming and may not be suitable for each individual instance. To verify the effectiveness of the tabu tenure strategy in our proposed algorithm, we refer to the tabu tenure settings in the algorithm designed by Wang et al. (2012), which are $0.01n$ for the Palubeckis instances set and $0.1n$ for the Max-Cut instances set. We generate a variant, Fixed Tenure, for comparison by fixing the tabu tenure. For fairness, this variation skips the first two phases probing the tabu tenure and dives into the scatter search directly, as it already presumably has the best tenure to perform tabu search. Representative instances were selected from the previous two instance sets: 11 instances with sizes $n \geq 5000$ from the Palubeckis instance set and the 9 largest Max-Cut instances.

Table 8
Experimental comparison between Adaptive Tenure and Fixed Tenure

Instances	Adaptive Tenure			Fixed Tenure		
	Best	AvgObj	AvgT	Best	AvgObj	AvgT
5000.1	**8559680**	**8559582.5**	905.0	**8559680**	8559452.5	288.1
5000.2	**10836019**	**10836019**	759.1	**10836019**	10835902.6	454.2
5000.3	**10489137**	**10489137**	312.9	**10489137**	10489101.6	582.5
5000.4	**12252318**	**12252147.2**	725.7	**12252318**	12252108.8	1098.6
5000.5	**12731803**	**12731803**	248.3	**12731803**	**12731803**	153.6
6000.1	**11384976**	11384916.1	578.6	**11384976**	**11384959.1**	688.0
6000.2	**14333855**	14333842.2	665.9	**14333855**	**14333855**	529.5
6000.3	**16132915**	**16132792.8**	836.8	**16132915**	16132696.6	461.6
7000.1	**14478676**	**14478289.4**	854	**14478676**	14478243.1	963.8
7000.2	**18249948**	18249238.2	984.6	**18249948**	**18249655.4**	502.8
7000.3	**20446407**	**20446407**	319.3	**20446407**	20445673.1	1023.6
G63	**26969**	**26957.5**	122.7	26930	26912.2	634.1
G64	**8703**	**8674.5**	123.2	8693	8659.4	412.9
G65	**5552**	**5547**	592.5	5546	5537.8	1057.7
G66	**6350**	**6341.2**	341.7	6338	6330	915.2
G67	**6936**	**6929.4**	735.2	6930	6914.8	849.3
G70	**9556**	**9544.5**	792.8	9488	9465.5	222.2
G72	**6986**	**6980**	579.9	6974	6964.6	1236.5
G77	**9910**	**9899.8**	917.6	9894	9884.2	1147.3
G81	**14014**	**13999.6**	1104.7	13978	13965	1153.4

Table 8 shows detailed comparison results obtained by the two algorithms for solving the 20 representative instances. Column 1 again gives the instance name, and the following columns list the best and average objective values, and average time to the best. From the table, we can see that our proposed algorithm with the adaptive tabu tenure performs significantly better than the variant with the fixed tabu tenure. Specifically, regarding the best objective values, our algorithm performs better for 9 out of 20 instances and no worse instance. In terms of the average objective values, our algorithm performs better for 16 instances and worse for 3 instances. Among them, the Fixed Tenure variant performs as good as SATPR on the Palubeckis instances in terms of the best and average objectives and is significantly worse on the Max-cut instances set.

In conclusion, the adaptive tenure will be at least as effective as the previously tuned tenure and can surpass it when instances are not perfectly tuned or exhibit high sensitivity. Tuning on the entire set may not be sufficient in such cases.

3.5 Effectiveness of the RefSet Update Method

The new reference set update method evaluates the weighted value based on both solution quality and variation, maintaining the best $|E|$ solutions. This is in contrast to many scatter search implementations, which split the whole set into two portions: $\frac{|E|}{2}$ solutions with the best objective values and the other $\frac{|E|}{2}$ with the largest distances from all reference set members. The classical implementations will tend to create more diverse offspring, but their poor solution value and tabu tenure can also pose challenges in finding the best solution when doing solution combination. Therefore, we replaced our Weighted Update method with the half-half updating strategy and created Splitted Update variation to distinguish between those two strategies.

Table 9
Experimental comparison between Weighted Update and Splitted Update

Instances	Weighted Update			Splitted Update		
	Best	AvgObj	AvgT	Best	AvgObj	AvgT
5000.1	**8559680**	8559582.5	905.0	**8559680**	**8559647.5**	886.6
5000.2	**10836019**	**10836019**	759.1	**10836019**	**10836019**	387.7
5000.3	**10489137**	**10489137**	312.9	**10489137**	10489101.6	914.6
5000.4	**12252318**	12252147.2	725.7	**12252318**	**12252171.8**	893.1
5000.5	**12731803**	**12731803**	248.3	**12731803**	**12731803**	336.6
6000.1	**11384976**	11384916.1	578.6	**11384976**	**11384976**	237
6000.2	**14333855**	**14333842.2**	665.9	**14333855**	14333704.2	1095.8
6000.3	**16132915**	**16132792.8**	836.8	**16132915**	16130771.3	1301
7000.1	**14478676**	14478289.4	854.0	**14478676**	**14478384.4**	1125.2
7000.2	**18249948**	**18249238.2**	984.6	**18249948**	18249183.1	1134.5
7000.3	**20446407**	**20446407**	319.3	**20446407**	**20446407**	813.2
G63	**26969**	**26957.5**	122.7	26959	26944.9	98.3
G64	**8703**	**8674.5**	123.2	8684	8656	85.1
G65	**5552**	**5547**	592.5	5540	5533	419.7
G66	**6350**	**6341.2**	341.7	6342	6323.8	324.2
G67	**6936**	**6929.4**	735.2	6920	6914.8	137.5
G70	**9556**	**9544.5**	792.8	9539	9525.4	235.7
G72	**6986**	**6980**	579.9	6978	6967.8	319.3
G77	**9910**	**9899.8**	917.6	9894	9889.6	346.7
G81	**14014**	**13999.6**	1104.7	13984	13970.6	1341.4

The detailed comparison results of these update strategies are given in Table 9. It can be seen that, in terms of the best objective values, compared with the Splitted Update variant, our algorithm performs better for 9 instances. In terms of the average objective values, our algorithm performs better for 13 instances and worse for 4 instances. From the instance set perspective, SATPR ties with the variant on the Palubeckis set and outperforms the variant on the Max-Cut set where higher demands on tenure and reference set quality are required. This aligns with our primary evaluations of the old reference set maintenance strategy. As a result, the new weighted update strategy is preferred, as it performs better across all kinds of instances by maintaining a relatively quality-focused reference set and better tabu tenure.

3.6 Effectiveness of the Two-way Path-relinking Strategy

Table 10
Experimental comparison between Two-way and One-way Path-relinking

Instances	Two-way Path-relinking			One-way Path-relinking		
	Best	AvgObj	AvgT	Best	AvgObj	AvgT
5000.1	**8559680**	**8559582.5**	905.0	**8559680**	8559517.5	826.1
5000.2	**10836019**	**10836019**	759.1	**10836019**	**10836019**	722.7
5000.3	**10489137**	**10489137**	312.9	**10489137**	10489106.2	434.2
5000.4	**12252318**	12252147.2	725.7	**12252318**	**12252182.2**	678.9
5000.5	**12731803**	**12731803**	248.3	**12731803**	**12731803**	308.4
6000.1	**11384976**	11384916.1	578.6	**11384976**	**11384976**	855.1
6000.2	**14333855**	**14333842.2**	665.9	**14333855**	14333802.2	525.5
6000.3	**16132915**	**16132792.8**	836.8	**16132915**	16132450.6	771.1
7000.1	**14478676**	14478289.4	854.0	**14478676**	**14478305.1**	901.6
7000.2	**18249948**	**18249238.2**	984.6	**18249948**	18249116.3	1032.6
7000.3	**20446407**	**20446407**	319.3	**20446407**	**20446407**	708.1
G63	26969	**26957.5**	122.7	**26973**	26942.1	723.1
G64	**8703**	**8674.5**	123.2	8689	8667.6	793.4
G65	**5552**	**5547**	592.5	5548	5538.2	258.8
G66	**6350**	**6341.2**	341.7	6342	6332.8	983.7
G67	**6936**	**6929.4**	735.2	6930	6919.2	403.4
G70	**9556**	**9544.5**	792.8	9541	9525.5	996.3
G72	**6986**	**6980**	579.9	6984	6974.2	269.3
G77	**9910**	**9899.8**	917.6	9902	9889	382.1
G81	**14014**	**13999.6**	1104.7	13998	13981.4	1254.2

The last experiment concentrates on the path-relinking-inspired solution combination method. In this method, the variables to flip are alternately selected from both reference solutions, as opposed to choosing from only one solution. This Two-way Path-relinking strategy generally creates better new solutions as it evaluates the properties of the other reference solution while branching on one side. This prevents the path from flipping some of the crucial variables by elementary local search. A new algorithmic variant, which replaces the alternative path-relinking with the previously employed One-way Path-relinking strategy, was implemented. The detailed results of these two strategies are provided in Table 10. It is evident that, in terms of the best objective values, our algorithm outperforms the One-way Path-relinking variation for 8 instances and performs worse for only 1 instance. Regarding the average objective values, our algorithm exhibits superior performance for 13 instances and inferior performance for 3 instances. The consistently better performance across all three sets strongly emphasizes the superiority of the proposed alternative path-relinking strategy.

4 Conclusions and Future Work

The QUBO model has proven to be very successful for many important problems in the classical computing field. We demonstrate in this paper that our new open-source solver SATPR is capable of competing favorably with the leading metaheuristics and quantum solvers on very large-scale QUBO instances of different structures. Specifically, compared to other metaheuristics, our algorithm found better solutions for 21 out of 63 instances, matched the best solutions for 36 instances, and performed worse on 6 instances. Additionally, compared to the Fixstars Amplify digital annealer, our algorithm found better solutions for 9 out of 38 instances, matched the best solutions for 21 instances, and found worse solutions for 8 instances. Moreover, as our solver is open-source, individuals have the opportunity to contribute to its development, enhancing the algorithm's robustness for benchmarking different QUBO applications.

For future research, we intend to incorporate preprocessing procedures to enhance algorithm's efficiency. These procedures aim to substantially reduce the size of the QUBO matrix by identifying variables whose optimal values can be predetermined. Another forthcoming project is to run benchmarks with exact solvers such as Gurobi (2023) and SCIP (Rehfeldt et al. , 2023) and to compare our solvers on various applications such as support vector machines, feature selections, clique partitioning, graph coloring, scheduling problems and so on. Finally, it is important to develop new solvers to solve other QUBO related models such as PUBO and QUBO-Plus models (Du et al. , 2024, 2025).

Acknowledgement

Special thanks to Fred Glover and Gary Kochenberger for the motivation, support and many discussions that led to this work. This work was partially supported by the National Natural Science Foundation of China (No. 72371200, 71971172).

References

Abbas, A., A. Ambainis, B. Augustino, A. Bärtschi, H. Buhrman, C. Coffrin, G. Cortiana, et al. (2023), Quantum Optimization: Potential, Challenges, and the Path Forward. *arXiv preprint*, arXiv:2312.02279.

Anthony, M., E. Boros, Y. Crama, A. Gruber (2017) Quadratic Reformulations of Nonlinear Binary Optimization Problems. *Math. Program.* 162(1-2):115–144.

Aramon, M., G. Rosenberger, E. Valiante, T. Miyazawa, H. Tamura, H.G. Katzgraber (2019) Physics-Inspired Optimization for Quadratic Unconstrained Problems Using a Digital Annealer. *Front. Phys.* 7:48.

Barak, B., A. Moitra, R. O'Donnell, P. Raghavendra, O. Regev, D. Steurer, et al. (2015) Beating the Random Assignment on Constraint Satisfaction Problems of Bounded Degree. *arXiv preprint*, arXiv:1505.03424.

Boixo, S., T.F. Rønnow, S.V. Isakov, Z. Wang, D. Wecker, D.A. Lidar, J.M. Martinis, M. Troyer (2014) Evidence for Quantum Annealing with More Than One Hundred Qubits. *Nat. Phys.* 10:218–224.

Bunyk, P.I, E.M. Hoskinson, M.W. Johnson, E. Tolkacheva, F. Altomare, A.J. Berkley, R. Harris, J.P. Hilton, T. Lanting, A.J. Przybysz, and J. Whittaker (2014) Architectural considerations in the design of a superconducting quantum annealing processor. *IEEE Transactions on Applied Superconductivity*, 24(4): 1–10.

Du, Y., F. Glover, G. Kochenberger, R. Hennig, H. Wang, A. Hulandageri (2024) Solving the Minimum Sum Coloring Problem: Alternative Models, Exact Solvers, and Metaheuristics. *INFORMS J. Comput.*, forthcoming.

Du, Y., H. Wang, R. Hennig, A. Hulandageri, G. Kochenberger, F. Glover (2025) New Advances for Quantum-Inspired Optimization. *Int. Trans. Oper. Res.*, 32(1): 6–17.

Dunning, I., S. Gupta, J. Silberholz (2018) What Works Best When? A Systematic Evaluation of Heuristics for Max-Cut and QUBO. *INFORMS J. Comput.* 30:608–624.

Farhi, E., F. Goldstone, S. Gutmann (2014) A Quantum Approximate Optimization Algorithm. *arXiv preprint*, arXiv:1411.4028.

Fisxtar Amplify Digital Annealer (2021) *https://amplify.fixstars.com/en/*.

Fujitsu Digital Annealer (2021) *https://www.fujitsu.com/global/services/business-services/digital-annealer*.

Furini, F., E. Traversi, P. Belotti (2019) QPLIB: A Library of Quadratic Programming Instances. *Math. Program. Comp.* 11:237–265.

Glover, F., G. Kochenberger, B. Alidaee (1998) Adaptive Memory Tabu Search for Binary Quadratic Programs. *Manag. Sci.* 44:336–345.

Glover, F., M. Laguna, R. Marti (2003) Scatter Search and Path Relinking: Advances and Applications. *Handb. Metaheuristics* 57:1–35.

Glover, F., M. Laguna, R. Marti (2004) Scatter Search and Path Relinking: Foundations and Advanced Designs. *New Optim. Technol. Eng.* 141:87–100.

Glover, F., Z. Lu, J.K. Hao (2010) Diversification-Driven Tabu Search for Unconstrained Binary Quadratic Problems. *4OR Q. J. Oper. Res.* 8:239–253.

Glover, F., J-K. Hao, G. Kochenberger (2011) Polynomial Unconstrained Binary Optimization – Part 1. *Int. J. Metaheuristics* 1(3):232–256.

Glover, F., G. Kochenberger, R. Hennig, Y. Du (2022a) Quantum Bridge Analytics I: A Tutorial on Formulating and Using QUBO Models. *Ann. Oper. Res.* 314:141–183.

Glover, F., G. Kochenberger, Y. Du (2022b) Applications and Computational Advances for Solving the QUBO Model. In: Punnen, A.P. (ed.), *The Quadratic Unconstrained Binary Optimization Problem*, Springer.

Goudet, O., A. Goëffon, J.K. Hao (2024) A Large Population Island Framework for the Unconstrained Binary Quadratic Problem. *Comput. Oper. Res.* 168:106684.

Grover, L.K. (1996) A Fast Quantum Mechanical Algorithm for Database Search. *Proc. 28th ACM Symp. Theory Comput. (STOC '96)*, Philadelphia, PA: ACM, 212–219.

Gurobi Optimization, LLC (2023) Gurobi Optimizer Reference Manual.

Hanafi, S., Y. Wang, F. Glover, W. Yang, R. Hennig (2023) Tabu Search Exploiting Local Optimality in Binary Optimization. *Eur. J. Oper. Res.* 308(3):1037–1055.

Harrow, A.W., A. Hassidim, S. Lloyd (2008) Quantum Algorithm for Solving Linear Systems of Equations. *Phys. Rev. Lett.* 103(15):150502.

Kochenberger, G., F. Glover, B. Alidaee, C. Rego (2004) A Unified Modeling and Solution Framework for Combinatorial Optimization Problems. *OR Spectrum* 26(2):237–

250.

Kochenberger, G., J-K. Hao, F. Glover, M. Lewis, Z. Lu, H. Wang, Y. Wang (2014) The Unconstrained Binary Quadratic Programming Problem: A Survey. *J. Comb. Optim.* 28(1):58–81.

Kochenberger, G., J-K. Hao, Z. Lu, H. Wang, F. Glover (2013) Solving Large-Scale Max Cut Problems via Tabu Search. *J. Heuristics* 19(4):565–571.

Lü, Z., F. Glover, J-K. Hao (2010) A Hybrid Metaheuristic Approach to Solving the UBQP Problem. *Eur. J. Oper. Res.* 207(3):1254–1262.

Lucas, A. (2014) Ising Formulations of Many NP Problems. *Front. Phys.* 2(5). Available at: *https://arxiv.org/abs/1302.5843*.

Martí, R., A. Duarte, M. Laguna (2009) Advanced Scatter Search for the Max-Cut Problem. *INFORMS J. Comput.* 11(1):26–38.

Martí, R., M. Laguna, F. Glover (2006) Principles of Scatter Search. *Eur. J. Oper. Res.* 169(2):359–372.

Merz, P., K. Katayama (2004) Memetic Algorithms for the Unconstrained Binary Quadratic Programming Problem. *BioSystems* 78:99–118.

Molina, J., M. Laguna, R. Martí, R. Caballero (2007) SSPMO: A Scatter Search Tabu Search Procedure for Non-Linear Multiobjective Optimization. *INFORMS J. Comput.* 19(1):91–100.

Parekh, O. (2023) Synergies Between Operations Research and Quantum Information Science. *INFORMS J. Comput.* 35(2):266–273.

Peruzzo, A., J. McClean, P. Shadbolt, M.H. Yung, X.Q. Zhou, P.J. Love, A. Aspuru-Guzik, J.L. O'Brien (2014) A Variational Eigenvalue Solver on a Photonic Quantum Processor. *Nat. Commun.* 5(1):4213.

Punnen, A.P. (2022) *The Quadratic Unconstrained Binary Optimization Problem: Theory, Algorithms, and Applications*. Springer, Cham.

Rehfeldt, D., T. Koch, Y. Shinano (2023) Faster Exact Solution of Sparse Max-Cut and QUBO Problems. *Math. Program. Comp.* 15:445–470.

Samorani, M., Y. Wang, Z. Lv, F. Glover (2019) Clustering-Driven Evolutionary Algorithms: An Application of Path Relinking to the Quadratic Unconstrained Binary Optimization Problem. *J. Heuristics* 25:629–642.

Sánchez-Oro, J., M. Laguna, A. Duarte, R. Martí (2015) Scatter Search for the Profile Minimization Problem. *Networks* 65(1):10–21.

Shor, P.W. (1994) Algorithms for Quantum Computation: Discrete Logarithms and Factoring. *Proc. 35th Annu. Symp. Found. Comput. Sci. (FOCS '94)*, IEEE, 124–134.

Souza, M., M. Ritt (2018) Automatic Grammar-Based Design of Heuristic Algorithms for Unconstrained Binary Quadratic Programming. In: Liefooghe, A., M. Lopez-Ibanez (eds.), *EvoCOP 2018*, LNCS 10782, Springer, pp. 67–84.

Wang, Y., Z. Lu, F. Glover, J.K. Hao (2012) Path Relinking for Unconstrained Binary Quadratic Programming. *Eur. J. Oper. Res.* 223:595–604.

OpenMP Architecture Review Board (2018) OpenMP Application Program Interface Version 5.0. *OpenMP*. Available at: *https://www.openmp.org/wp-content/uploads/OpenMP-API-Specification-5.0.pdf*.