QiSA: A Parameter-free Quantum-inspired Search Algorithm – A Preliminary Study on Combinatorial Optimization

Kuo-Chun Tseng*, Wei-Chieh Lai*, Chen-Hsin Lu†, Wei-Chun Huang‡, Jen-Shin Hong§, Hsin-Hung Cho*
*Department of Computer Science and Information Engineering, National Ilan University, Yilan, Taiwan
†Department of Materials Science and Engineering, National Tsing Hua University, Hsinchu, Taiwan
‡National Chung-Shan Institute of Science & Technology, Taoyuan, Taiwan
§Department of Computer Science and Information Engineering, National Chi Nan University, Nantou, Taiwan
Corresponding author: hhcho@niu.edu.tw

***Abstract*—This study is inspired by quantum algorithms and proposes the Quantum-inspired Search Algorithm (QiSA), an entropy-driven quantum-inspired metaheuristic for parameter-free combinatorial optimization. Guided by Occam's razor, QiSA simplifies the search process and reduces parameter dependence as much as possible in this preliminary study. It uses entropy as the core signal to unify parameter control, overcoming the traditional dilemma of balancing exploration and exploitation, and maps it to a confidence-like level to define an interpretable closed-loop termination criterion. QiSA also exhibits several observable quantum-like behaviors during the search, reflecting characteristic phenomena associated with quantum algorithms. Experiments on classic combinatorial optimization benchmarks show that QiSA delivers robust, consistent, and competitive performance, with the potential to stand alongside Differential Evolution in combinatorial optimization.**

***Index Terms*—Quantum-inspired metaheuristic, Grover's algorithm, Entropy-driven metaheuristic, Parameter-free metaheuristic, Combinatorial optimization.**

I. Introduction

Researchers have long wanted an optimization method that works well on many problems without tuning control parameters. However, the No-Free-Lunch (NFL) theorem [1] says that if objective functions are uniformly random over the search space, then all algorithms have the same average performance. This was once seen as a barrier to building "general-purpose" algorithms. In practice, the NFL assumption is very strict. As noted by Igel [2] and McDermott [3], real optimization problems are shaped by physical laws and problem structures, not by uniform randomness. Because real problems contain exploitable structure, we can still design adaptive and efficient algorithms for real-world problem sets without being limited by NFL [1].

To achieve this goal on real-world problems, a common approach is parameter control, where an algorithm adjusts its internal parameters during evolution [4]. The Differential Evolution (DE) family is one of the most successful examples [5], [6]. Variants such as SHADE, L-SHADE, and jSO [7], [8] use historical memory and adaptive rules to capture problem features, and have performed very well in both continuous and discrete domains. However, even these strong DE variants [6] are still mainly built around the classic trade-off between exploration and exploitation [9]. To balance these two forces, designers often add more repair operators and intricate adaptive mechanisms, makes the algorithms complex and harder to implement. This suggests that the "two-force balance" mindset has become another bottleneck in algorithm development.

The idea of quantum computing offers a new direction for search theory. This paradigm may guide us toward parameter-free algorithms and avoid the burden of balancing two forces. Many studies add "quantum" features to classical metaheuristics [10], creating many variants. Although these features can improve performance in some cases, they often increase complexity and introduce more parameters without truly capturing real quantum effects. Among these studies, some researchers have focused on Grover's algorithm [11], which marks target solutions and adjusts amplitudes to gradually maximize their probabilities. This process is simple, parameter-free, and coherent in design. Therefore, if this idea can be transferred to optimization, it may lead to a parameter-free and easy-to-implement algorithm without many variants or extra tuning.

Inspired by these ideas, quantum-inspired algorithms based on pure quantum concepts use simulated qubits to imitate amplitude updates and parallel search. The Quantum-inspired Evolutionary Algorithm (QEA) [12] introduced solution marking and amplitude adjustment. The Quantum-inspired Tabu Search algorithm (QTS) [13] addressed the lack of true parallelism in simulated qubits by expanding a subset of candidate solutions each generation to update probability amplitudes. The Amplitude-Ensemble QTS (AE-QTS) [14] further addressed a key limitation of QTS [13]: it uses information from the entire population during amplitude adjustment, bringing it closer to quantum-style updating. Further analysis shows that QEA [12] and QTS [13] often do not converge, while AE-QTS [14] converges consistently. Quantum-like effects are also observed in AE-QTS [14], such as probability maximization of the best solution, entanglement-like behavior, and probability oscillation among top solutions. Therefore, AE-QTS [14] stands out from other quantum-inspired methods [10] and is closest to quantum algorithm design thinking.

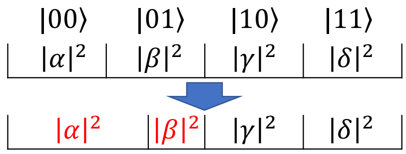


Fig. 1: Probability representation with multi-qubit binding.

AE-QTS [14] still has three parameters: $\Delta\theta$, population size, and the termination criterion (maximum generations). If we ignore the common settings (population size and termination) and focus on the core design, AE-QTS already removes explicit operations and parameters for exploration and exploitation, because these two forces are merged into the update rule itself. Following this idea, we develop the Quantum-inspired Search Algorithm (QiSA). We continue to use key ideas from Grover's algorithm [11], including mean-based amplitude adjustment and fixed-angle scaling. **Specifically, our contributions in this preliminary study are: (i) a Grover-like, mean-based update principle that consolidates exploration and exploitation into the update rule through a unified control of $\Delta\theta$; (ii) an entropy-driven $\Delta\theta$ adaptation mechanism, where Shannon entropy [15] measures global uncertainty and is combined with Grover's scaling principle [11] to reduce manual tuning; and (iii) an interpretable termination criterion that maps entropy to a confidence-like level, aligns with the probabilistic nature of quantum systems and moves QiSA closer to a closed-loop, parameter-free design.** Thus, QiSA is easy to implement, variant-free, and stable.

II. Background

This section defines the notations used in this study, introduces key concepts, and describes four combinatorial optimization problems [16]–[18]. These problems are used to demonstrate QiSA's behavior and performance and to support later design discussions.

A. Quantum Computing Notation Definitions

We use bra-ket notation for a qubit state $\alpha|0\rangle + \beta|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$, $|0\rangle = \begin{bmatrix}1 & 0\end{bmatrix}^T$, and $|1\rangle = \begin{bmatrix}0 & 1\end{bmatrix}^T$. If each qubit can be represented independently, we update it with the rotation matrix $R(\Delta\theta)$ (Eq. 1). If multiple qubits are bound together, we update the system in probability form by directly adjusting state probabilities. For example, two qubits represent $|00\rangle, |01\rangle, |10\rangle, |11\rangle$ (Fig. 1); the four cells show their probabilities, which we modify directly.

$$R(\Delta\theta) = \begin{bmatrix}\cos\Delta\theta & -\sin\Delta\theta \\ \sin\Delta\theta & \cos\Delta\theta\end{bmatrix}. \tag{1}$$

B. Entropy-to-confidence-like mapping

Unlike traditional metaheuristics that explicitly maintain population diversity, QiSA focuses on the uncertainty of simulated qubits to gauge remaining unexplored regions. Because these qubits are represented by classical probabilities, we use Shannon entropy [15] to naturally quantify this uncertainty, avoiding the prohibitive overhead of quantum-mechanical alternatives. If qubits are independent, we compute each qubit's entropy using Eq. 2. If qubits are bound in groups, we compute the entropy of each fixed-size group using Eq. 3.

$$H_{\text{bind}}^{(j)} = -\left(p_{j,0}\log_2 p_{j,0} + p_{j,1}\log_2 p_{j,1}\right), \tag{2}$$

where $p_{j,0} = |\alpha_j|^2$ and $p_{j,1} = |\beta_j|^2$ denote the measurement probabilities of the j-th qubit.

$$H_{\text{bind}}^{(g)} = \frac{-\sum_{x\in\{0,1\}^m} p_g(x)\log_2 p_g(x)}{m}, \tag{3}$$

where $g \in \{1,\dots,G\}$ indexes the binding groups, each group binds m qubits, $x \in \{0,1\}^m$ denotes an m-bit basis state, and $p_g(x)$ is the joint probability of observing state x for the g-th group, satisfying $\sum_{x\in\{0,1\}^m} p_g(x) = 1$ and $H_{\text{bind}}^{(g)} \in [0,1]$.

Shannon entropy [15] measures overall uncertainty during the search. A confidence-like level measures how sure we are, under a given rule, that the search has converged enough. Both describe the current search state, but they are not generally equivalent in theory. In practice, under finite samples and incomplete information, neither quantity usually reaches its extreme value. Therefore, we do not treat them as equivalent. Instead, we use a simple monotonic linear mapping to place them on the same scale and define an interpretable termination criterion. This is also consistent with the idea in quantum algorithms of stopping with an acceptable success probability (or failure rate). We define the linear relationship as follows. Let $CL \in [0,1]$ denote the confidence-like level (CL), where larger values mean higher certainty. The corresponding normalized entropy $H \in [0,1]$ is defined as

$$H \triangleq 1 - CL. \tag{4}$$

Therefore, this equation satisfies the endpoints: $CL = 0$ gives $H = 1$ (maximum uncertainty), and $CL = 1$ gives $H = 0$ (minimum uncertainty under this mapping). It is a monotonically decreasing mapping and can be used as a termination criterion or a convergence indicator.

C. Combinatorial Problems

We use four problems to evaluate the algorithm's behavior and performance. We also use results from AE-QTS [14] on these problems to reflect on the QiSA design, compare it with Grover's algorithm [11], and refine our ideas. The four classic combinatorial optimization problems [16]–[18] are the 0/1 Knapsack Problem (KP), Multidimensional Knapsack Problem (MKP), Uncapacitated Warehouse Location Problem (UWLP), and Traveling Salesman Problem (TSP). Many later discussions focus on knapsack problems, because their difficulty can be scaled easily within the same setting, making the algorithm's behavior easier to observe.

1) 0/1 Knapsack Problem (KP) [16]: The KP seeks the maximum total profit under a capacity limit. We use four KP types (CASE1-CASE4), where the number of items k can be chosen to control difficulty. Let w_i and p_i be the weight and profit of item i; the goal is to maximize total profit while keeping total weight within capacity C.

All later comparisons use these four cases. CASE1 and CASE2 are standard benchmarks in the literature [12], [13], while CASE3 and CASE4 are designed to ease analysis and increase difficulty. For analyzing algorithm behavior, we mainly use CASE3 because CASE1 and CASE2 are random and require many runs for statistical reliability. CASE3 has a deterministic structure, which makes the algorithm's behavior easier to observe.

CASE1: $w_i \sim \mathcal{U}[1, 10],\ \ p_i = w_i + 5.$
CASE2: $w_i \sim \mathcal{U}[1, 10],\ \ p_i = w_i + \ell_i,\ \ \ell_i \sim \mathcal{U}[0, 5].$
CASE3: $w_i = 1 + ((i-1) \bmod 10),\ \ p_i = w_i + 5.$
CASE4: $w_i = 1 + \big((i-1) \bmod 10\big),\ \ p_i = w_i + 100.$

2) Multidimensional Knapsack Problem (MKP) [17]: MKP extends the standard knapsack problem. It uses the same binary variables $x_i \in \{0, 1\}$ and still maximizes profit over k items, but adds m resource constraints. Each dimension j has capacity C_j, and item i has weight $w_{j,i}$ in that dimension. The objective function and constraints follow the standard OR-Library definition [17].

3) Uncapacitated Warehouse Location Problem (UWLP) [17]: UWLP selects warehouses to open from a candidate set $\mathcal{J}$ to minimize the total cost, including opening costs and transportation costs for serving customers $\mathcal{I}$. Each customer is assigned to the opened warehouse with the lowest service cost. We use the standard OR-Library definition [17].

4) Traveling Salesman Problem (TSP) [18]: TSP seeks a minimum-cost Hamiltonian cycle that visits a given set of cities $\mathcal{V}$ exactly once and returns to the start. Let d_{ij} be the distance between cities; the goal is to minimize the tour length $L(\pi)$ for a permutation π. We evaluate the algorithm using standard instances from TSPLIB95 [18].

III. Empirical Motivation

This section reports key AE-QTS [14] behaviors and quantum-like effects motivating QiSA, and clarifies its gradual integration with Grover's algorithm [11].

A. AE-QTS Parameters Study for QiSA

This section summarizes key properties of AE-QTS [14] that motivate the QiSA design. Fig. 2a shows that AE-QTS uses population information more efficiently than QTS [13]: with only 10 individuals, AE-QTS [14] achieves almost the same results as QTS [13] with 50. This supports building QiSA on the AE-QTS [14] concept. Fig. 2b shows AE-QTS [14] performance under different population sizes and $\Delta\theta$. Increasing the population size greatly improves performance, while the impact of $\Delta\theta$ is limited when it varies only slightly. This suggests that AE-QTS [14] is stable and robust to parameter variation, which is important for QiSA's dynamic angle design. Finally, Fig. 2c indicates that enlarging the population in AE-QTS [14] is more cost-effective than DE [5]. However, the benefit is not unlimited: the cost-effectiveness drops as the population becomes larger. This supports using a fixed $N = 200$ in QiSA, which may be sufficient across different difficulty levels.

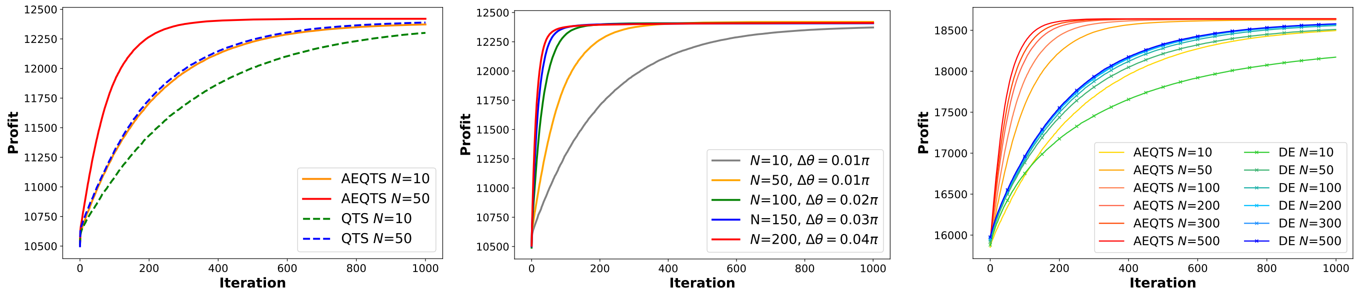
B. Quantum-Like Effects in Simulated Qubits

AE-QTS [14] also exhibits several quantum-like properties. These properties can be inherited by QiSA and may help it move closer to Grover's algorithm [11], completing a key missing piece of quantum-inspired metaheuristics. We test these effects on KP [16] CASE3, which is stable and easy to scale in difficulty. The observed effects are:

1. **Unconstrained maximization of simulated qubit probabilities:** Quantum algorithms try to maximize the probability of the target answer before measurement. QiSA and AE-QTS [14] show the same behavior. If we limit the maximum probability of a solution to keep population diversity, the final result becomes worse. In Fig. 3a, constraining each qubit's maximum probability to 90% for the first 500 iterations prevents convergence to the best solution. Once we remove the constraint after 500 iterations, the algorithm quickly converges. This suggests that, in the late stage, QiSA and AE-QTS [14] naturally produce nearly identical solutions, so population diversity is not a key concern.
2. **Entanglement-like behavior among simulated qubits:** We plot the probability trajectories of all qubits during evolution and observe that some qubits are positively or negatively correlated. In some cases, the correlation even reverses over time (Fig. 3b). This behavior resembles quantum entanglement, where qubit states evolve together or in opposite directions. It is also consistent with quantum algorithms: entanglement-like effects can emerge naturally without an explicit entanglement design.
3. **Oscillating probability changes among solutions with the same value:** In quantum algorithms, if multiple solutions share the same value, their probabilities can increase together, reach a maximum, then decrease and peak again periodically. QiSA and AE-QTS [14] behave differently. When they encounter equal-value solutions, the probabilities oscillate and compete with each other (Fig. 3c), showing a stronger "tug-of-war" among these solutions. This can stop QiSA's average entropy from decreasing smoothly, making automatic termination difficult. If many solutions have the same fitness value, the oscillation can also hinder convergence.

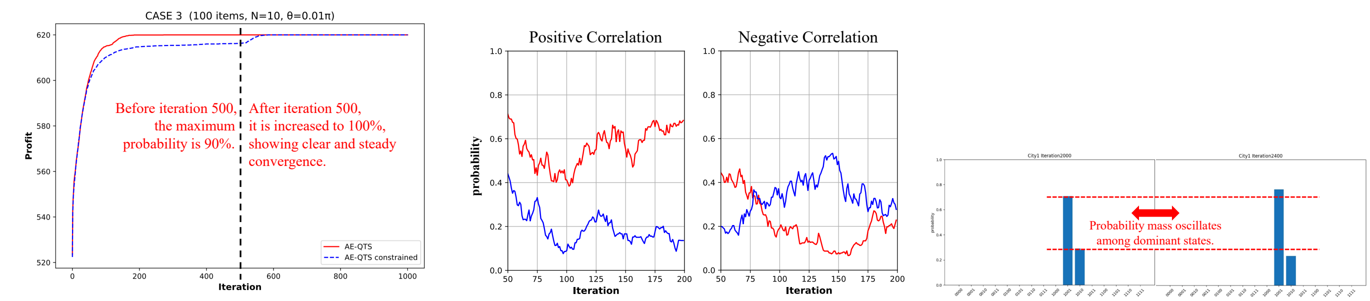
IV. Quantum-inspired Search Algorithm (QiSA)

AE-QTS [14] is close to Grover's algorithm [11], but it still has three parameters: the rotation angle $\Delta\theta$, the population size N, and the termination criterion. To move toward a parameter-free design in the spirit of Grover [11], QiSA uses Shannon entropy H [15] to quantify the search state and map it to a confidence-like level for an interpretable termination criterion. At each iteration t, we measure k qubits $Q(t)$ initialized as $\alpha\,|0\rangle + \beta\,|1\rangle$ with $\alpha = \beta = 1/\sqrt{2}$ (Eq. 5) to sample N solutions as a population $P(t)$. We evaluate and rank these solutions based on their fitness, where $f(\cdot)$ denotes the fitness value of a given candidate solution and $\bar{f}_t$ represents the average fitness of the population at iteration t. Then, we compute the pair-wise rotation $\Delta\theta_{t,x}$ by Eq. 8 (with



(a) AE-QTS vs QTS with 2000 items (b) Different N with 2000 items (c) AE-QTS vs DE with 3000 items

Fig. 2: AE-QTS [14] parameters study in KP CASE3.



(a) Convergence comparison with constrained vs. unconstrained probability values/settings (b) Entanglement-like probability correlations among quantum states (c) Back-and-forth oscillations of state probabilities

Fig. 3: Quantum-like effects of AE-QTS and QiSA.

variables from Eq. 6, Eq. 7, and Eq. 9), and update each $|q_i\rangle$ accordingly. Let $\bar{H}_t$ be the average entropy of the qubits at iteration t; it drives both $\Delta\theta_{t,x}$ and the termination criterion. We convert $\bar{H}_t$ to a confidence-like level using Eq. 4 and stop when $\bar{H}_t < 0.02$ (i.e., a 98% confidence-like level); users may choose an acceptable confidence-like level. In this preliminary study, N is kept fixed because our initial attempt to bind N with $\bar{H}_t$ was not satisfactory; thus, we use a relatively large N to help $\bar{H}_t$ reach the threshold. The full procedure is given in Algo. 1 and Algo. 2.

$$\begin{aligned} Q(0) &= \{\, |q_i\rangle \,\}_{i=1}^{k}, \\ |q_i\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \forall i \in \{1,\ldots,k\}. \end{aligned} \tag{5}$$

$$\Delta\theta_t = \Delta\theta_{t-1} + a \times \Delta\theta \left(\frac{\bar{f}_t - \bar{f}_1}{\bar{f}_{t-1} - \bar{f}_1}\right)^a \tag{6}$$

$$a = \begin{cases} 1, & \text{if } \bar{f}_t \text{ is better than } \bar{f}_{t-1}, \\ -1, & \text{if } \bar{f}_t \text{ is worse than } \bar{f}_{t-1}, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

$$\Delta\theta_{t,x} = \Delta\theta_t \times weight_{t,x}, \forall x \in \{1,\ldots,\frac{N}{2}\} \tag{8}$$

$$weight_{t,x} = \bar{H}_t^2 \times \frac{f_t(x_{\text{best}}^{\text{th}}) - f_t(x_{\text{worst}}^{\text{th}})}{f_t(1_{\text{best}}^{\text{st}}) - f_t(1_{\text{worst}}^{\text{st}})} + \frac{1-\bar{H}_t^2}{x} \tag{9}$$

Line 1-5: We initialize constants $\Delta\theta$ and N (Section III-A). Following Grover's algorithm [11], the initially small $\Delta\theta_t$ grows as a multiple of $\Delta\theta$. As certainty rises, $\bar{H}_t$ drops below 0.02 (corresponding to a 98% confidence-like level; Eq. 4), enabling cost-saving early termination.

Line 8: We measure $Q(t-1)$ to generate N binary strings as the population $P(t)$. For each bit i, we draw a random number $r_i \in [0,1]$ and compare it with $|\beta_i|^2$: output '1' if $r_i \leq |\beta_i|^2$, otherwise output '0'.

Line 13-15: If the iteration-best fitness remains unchanged, insert the global best s^b at the population's front to guide the search and reduce convergence-hindering oscillations.

Line 16-18: Except for the first iteration, we recompute $\Delta\theta_t$ each iteration. It scales the minimum angle $\Delta\theta$ multiplicatively based on whether the population improves and by how much. To measure relative improvement, we subtract the first-iteration average fitness $\bar{f}_1$, as used in Eq. 9 and Eq. 7. This is similar to Grover's angle adjustment [11] and mean-guided amplitude updates. The variable a controls the update direction and step size: if the population improves, the angle increases; if it worsens, the angle decreases; if there is no change, the angle stays the same.

Line 19: We update the quantum states using Algo. 2. We pair solutions from the whole population and compute a rotation angle for each pair. This lets the update use information from all solutions, similar to quantum algorithms where amplitude amplification acts on the entire system.

Next, we explain another important subroutine for updating the quantum states, Algo. 2. We focus on the key steps and explain our design thinking as follows:

Algorithm 1 Quantum-inspired Search Algorithm (QiSA)

1: $\Delta\theta \leftarrow 0.0001\pi$
2: $N \leftarrow 200$ ▷ fixed constant in this preliminary study
3: $t \leftarrow 0$
4: initialize $Q(t)$
5: calculate $\bar{H}_t$ by Eq. 2 and Eq. 3
6: **while** $\bar{H}_t \geq 0.02$ **do**
7: $\quad t \leftarrow t+1$
8: $\quad$ make $P(t)$ with N solutions by measuring $Q(t-1)$
9: $\quad$ repair $P(t)$ ▷ apply specific repair if required
10: $\quad$ evaluate $P(t)$
11: $\quad \bar{f}_t =$ average fitness of $P(t)$
12: $\quad P_{\text{best}}(t) =$ the best solution in $P(t)$
13: $\quad$ **if** $t > 1$ **and** $f(P_{\text{best}}(t)) = f(P_{\text{best}}(t-1))$ **then**
14: $\qquad$ insert s^b at the beginning of $P(t)$
15: $\quad$ **end if**
16: $\quad$ **if** $t > 2$ **then**
17: $\qquad$ update $\Delta\theta_t$ by Eqs. 6 and 7
18: $\quad$ **end if**
19: $\quad$ update $Q(t)$ by $P(t)$ ▷ see Algo. 2
20: $\quad$ update $\bar{H}_t$
21: $\quad$ update s^b if $P_{\text{best}}(t)$ is better than s^b
22: **end while**

Note: Here, "best" and "worst" follow the problem objective (max/min); no formulas need to be changed.

Algorithm 2 Update $Q(t)$ of QiSA

1: $x \leftarrow 0$
2: $index \leftarrow$ sort $P(t)$ from best to worst
3: $maximum = |f_t(1^{\text{st}}_{\text{best}}) - f_t(1^{\text{st}}_{\text{worst}})|$
4: **while** $x < N/2$ **do**
5: $\quad p^b \leftarrow index\,[x]$
6: $\quad p^w \leftarrow index\,[N-1-x]$
7: $\quad x \leftarrow x+1$
8: $\quad$ **if** $maximum = 0$ **then**
9: $\qquad weight_{t,x} \leftarrow \dfrac{(1-\bar{H}_t^2)}{x}$
10: $\quad$ **else**
11: $\qquad$ calculate $weight_{t,x}$ by Eq. 9
12: $\quad$ **end if**
13: $\quad \Delta\theta_{t,x} = \Delta\theta_t \times weight_{t,x}$
14: $\quad i \leftarrow 0$
// for bound qubits, the algorithm directly adjusts the joint state probabilities, bypassing individual rotation gates
15: $\quad$ **while** $i < k$ **do**
16: $\qquad$ Set the phase of $\Delta\theta_{t,x}$ by p_i^b and p_i^w ([12], [13])
17: $\qquad |q_i\rangle = \begin{bmatrix} \cos(\Delta\theta_{t,x}) & -\sin(\Delta\theta_{t,x}) \\ \sin(\Delta\theta_{t,x}) & \cos(\Delta\theta_{t,x}) \end{bmatrix} \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$
18: $\qquad i \leftarrow i+1$
19: $\quad$ **end while**
20: **end while**

Note: Here, "best" and "worst" follow the problem objective (max/min); no formulas need to be changed.

Line 3: $maximum$ is the fitness gap between the best and worst solutions in the current population. We compute it first to avoid division by zero in Eq. 9.

Line 5-7: p^b and p^w are the paired solutions. We build pairs by matching solutions from the best half and the worst half of the population. The index x denotes the x-th pair: the x-th best is paired with the x-th worst.

Line 8-12: We compute $weight_{t,x}$ based on $maximum$, where $f(\cdot)$ simply denotes the evaluated fitness value of a given candidate solution. If the population shows no fitness difference, then $\bar{H}_t \approx 0$ and $weight_{t,x}$ becomes almost the same as in AE-QTS [14]. Otherwise, the search is still in an early stage with larger $\bar{H}_t$, so we use Eq. 9. In this case, pairs with a larger fitness gap relative to the population best-worst gap receive a larger weight, because their update direction is more certain. We square $\bar{H}_t$ to smooth the angle change and avoid overly large updates.

Line 13: We then compute the rotation angle for the x-th pair using Eq. 8. Early in the search, pairs with larger fitness gaps receive larger rotations. Later, the behavior becomes close to the AE-QTS weighting scheme [14].

Line 16: Following the rotation strategy described in [12], [13], we identify the quadrant of the current quantum state to choose the correct direction toward $|0\rangle$ or $|1\rangle$. This determines the phase (sign) of $\Delta\theta_{t,x}$.

Line 17: We update $|q_i\rangle$ using the rotation angle. For bound multi-qubits, we directly add or subtract the computed probability instead.

V. Experimental Results

QiSA is studied as a fundamental algorithm, so we avoid complex problem-specific adaptive mechanisms. To ensure fairness, we compare canonical versions and exclude DE variants such as SHADE, L-SHADE, and jSO [7], [8]. The compared methods are AE-QTS [14], QTS [13], QEA [12], Genetic Algorithm (GA) [19], DE [5], Tabu Search (TS) [20], Particle Swarm Optimization (PSO) [21], and Whale Optimization Algorithm (WOA) [22]. In this section, we first define unified metrics and parameter settings, then present convergence curves for selected hard instances to compare QiSA with other methods, especially quantum-inspired metaheuristics [12]–[14]. Finally, we discuss the effects and possible issues of entropy-based termination.

A. Overall Comparisons with Other Algorithms

This section compares all algorithms on all problems. For fairness, all algorithms use the same population size, 200. QiSA stops when $\bar{H} < 0.02$, while other algorithms stop at a fixed maximum iteration. For KP [16], MKP [17], and UWLP [17], the maximum is 500 iterations, so other algorithms run $200 \times 500 = 100000$ evaluations. TSP [18] is harder, so its maximum iterations vary by benchmark: 1000 (burma14, ulysses16), 2000 (ulysses22), 5000 (eil51, berlin52), 7000 (eil76, pr76), 9000 (gr96), 10000 (eil101), and 13000 (gr137). We repeat each experiment 100 times, and all convergence curves in the figures represent the average results of these 100 independent runs.

Before the main comparison, we tuned each algorithm's parameters using an easier instance from each problem class and applied this configuration to all instances within that class (Table I). In total, we evaluated 85 instances across four problem classes: KP (16 instances) [16], MKP (24 instances) [17], UWLP (35 instances) [17], and TSP (10 instances) [18]. Detailed instance names are provided in Table III.

Because these benchmarks include many instances, we cannot report all detailed results. Instead, we use the average error in Eq. 10 to summarize the overall performance, where error_j denotes the gap between the algorithm's final result and the published best-known value for the j-th instance, and B is the total number of instances. For most instances, the best-known value is available. If it is not reported, we use the best solution found by all compared algorithms as the reference value. This allows us to compute each method's relative distance to the best value and summarize overall performance.

$$\mathrm{avg.\,error} = \frac{1}{B}\sum_{j=1}^{B}\mathrm{error}_j \tag{10}$$

Table II (the average-error results) shows that AE-QTS [14] and DE [5] win on different problems, though AE-QTS [14] is more stable overall. Inheriting these strengths, QiSA performs similarly to AE-QTS [14] across problems while remaining competitive with DE [5]. Overall, QiSA consistently ranks in the top three, yielding solutions near the best-known values.

We select one hard instance per class to show convergence (Fig. 4). In KP (Fig. 4a), QiSA starts slower due to a small $\Delta\theta$, but decisively increases the angle as the population improves, converging faster than AE-QTS [14]. In MKP (Fig. 4b), where quantum-inspired metaheuristics perform poorly, QiSA steadily converges. In UWLP (Fig. 4c), QiSA exhibits stable convergence. In TSP [18] (Fig. 4d), QEA [12] and QTS [13] fail, while AE-QTS [14] and QiSA succeed. This TSP result confirms AE-QTS [14] and QiSA fundamentally differ from QEA [12] and QTS [13] in their underlying search dynamics, with QiSA slightly outperforming AE-QTS [14] here.

B. Computational Efficiency and Robustness of QiSA

Table III reports the mean evaluations, mean, SD, CV, and $err.$ of QiSA across four problem classes [16]–[18]. In terms of robustness, KP, MKP, and UWLP show low average CV values of 0.616%, 0.183%, and 0.647%, respectively, and their generally small $err.$ values further indicate stable solution quality. By contrast, TSP has a much higher average CV of 3.45%, showing that QiSA is much more robust on KP, MKP, and UWLP than on TSP. In terms of evaluation efficiency, QiSA achieves average evaluation improvements of 8.8%, 56.33%, and 76.92% on KP, MKP, and UWLP, respectively. This efficiency likely comes from the distinct objective values of local optima, which help QiSA detect progress and converge more easily. Although KP needs more evaluations on average because small CASE1 instances contain many similar-fitness solutions that slow the convergence of $\bar{H}$, this effect becomes weaker on harder instances.

TSP [18] behaves differently and thus needs further discussion. The success rate of QiSA on TSP, defined as the probability that $\bar{H}$ reaches 0.02, is generally high but not guaranteed in every run: 100% (eil51, eil76), 97% (eil101), 96% (burma14, ulysses16, gr96), 93% (ulysses22), 82% (berlin52), 79% (gr137), and 75% (pr76). Although all rates exceed 75%, failed runs can still require many more evaluations and greatly raise the average. A possible reason is that different encoding values may decode to the same permutation and thus yield the same fitness. As a result, the probabilities of these equivalent states oscillate and compete with each other, as shown in Fig. 3c, preventing $\bar{H}$ from decreasing further. This issue will be a key topic in our future work.

VI. Conclusion and Future Work

QiSA builds on AE-QTS [14] and moves toward a parameter-free metaheuristic. Inspired by Grover's algorithm [11], it uses an average solution and adaptive rotation angles to update $\Delta\theta$, making the framework closer to a quantum algorithm with near closed-loop logic. This design targets the long-standing parameter-tuning problem by moving beyond the 'exploration vs. exploitation" balance requiring separate controls. It minimizes the need to maintain diversity and avoids complex mechanisms. AE-QTS [14] and QiSA show quantum-like effects (Section III-B), supporting our design direction and providing quantum and probabilistic interpretations for QiSA's behavior, improving physical interpretability.

Experiments (Section V) show that QiSA performs close to AE-QTS [14] and DE [5] across problem types, and that the entropy-based stopping rule works well in most cases to greatly reduce evaluations. For some TSP instances [18], however, many different tours can share the same fitness, triggering probability oscillations that prevent $\bar{H}_t$ from converging, even with the global-best insertion strategy (Algo. 1, Line 13). Moreover, as a preliminary study, QiSA has not yet achieved complete parameter independence. It still relies on a fixed constant, namely an empirically tuned population size N, because our initial attempt to bind N directly to entropy was not fully satisfactory. Future work will therefore focus on mitigating oscillations and removing this remaining constant. Specifically, we aim to design a dynamic population size mechanism to keep $\bar{H}_t$ decreasing, further reduce evaluations, and strengthen the overall closed-loop logic.

Acknowledgment

This work was supported by the National Science and Technology Council (NSTC), Taiwan, R.O.C., under Grants 114-2221-E-197-010, 114-2221-E-197-021-, and 112-2221-E-197-016-MY3.

References

[1] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

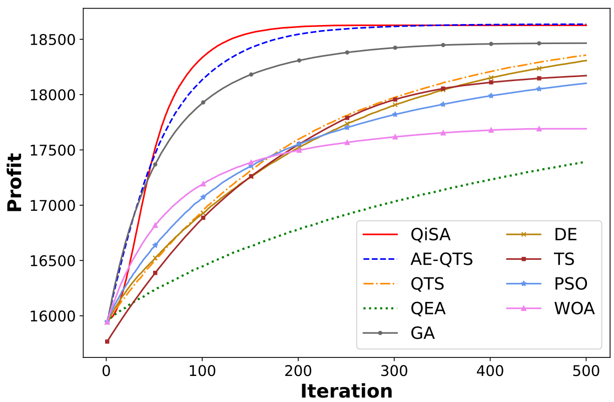
[2] C. Igel, "No free lunch theorems: Limitations and perspectives of metaheuristics," in *Theory and Principled Methods for the Design of Metaheuristics*. Springer, 2013, pp. 1–23.

[3] J. McDermott, "When and why metaheuristics researchers can ignore "No Free Lunch" theorems," *SN Computer Science*, vol. 1, no. 60, pp. 1–18, 2020.

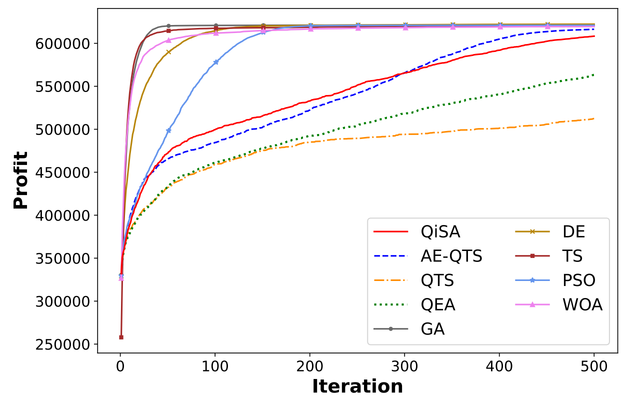
TABLE I: Parameter settings of all algorithms on four combinatorial optimization problems.

Problem	QiSA	AE-QTS	QTS	QEA	GA	DE	TS	PSO	WOA
KP	$\Delta\theta = 10^{-4}\pi$	$\Delta\theta = 10^{-2}\pi$	$\theta = 10^{-2}\pi$	$\Delta\theta = 10^{-2}\pi$	$CR = 0.6$ $MR = 0.45$	$CR = 0.3$	$tabu\ list = 10$	$c_1 = 1$ $c_2 = 1$	$a = 6$
MKP	$\Delta\theta = 10^{-4}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$CR = 0.6$ $MR = 0.45$	$CR = 0.3$	$tabu\ list = 10$	$c_1 = 1.5$ $c_2 = 1$	$a = 4$
UWLP	$\Delta\theta = 10^{-4}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$\Delta\theta = 5 \times 10^{-3}\pi$	$CR = 0.6$ $MR = 0.45$	$CR = 0.2$	$tabu\ list = 10$	$c_1 = 1$ $c_2 = 1$	$a = 2$
TSP	$\Delta\theta = 10^{-4}$	$\Delta\theta = 0.002$	$\Delta\theta = 0.005$	$\Delta\theta = 0.005$	$CR = 0.5$ $MR = 0.6$	$CR = 0.3$	$tabu\ list = 30$	$c_1 = 1.5$ $c_2 = 1$	$a = 8$

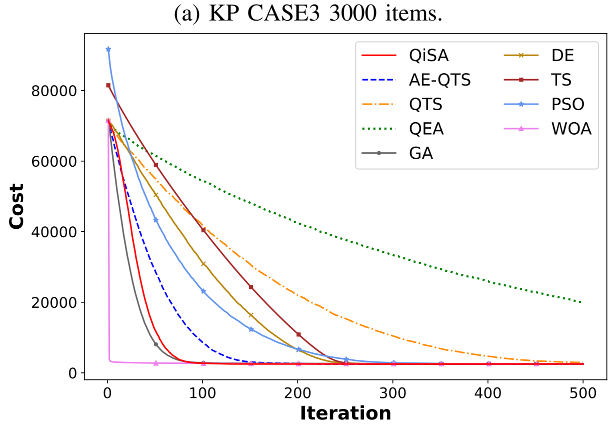
Note: *CR* is crossover rate. *MR* is mutation rate.



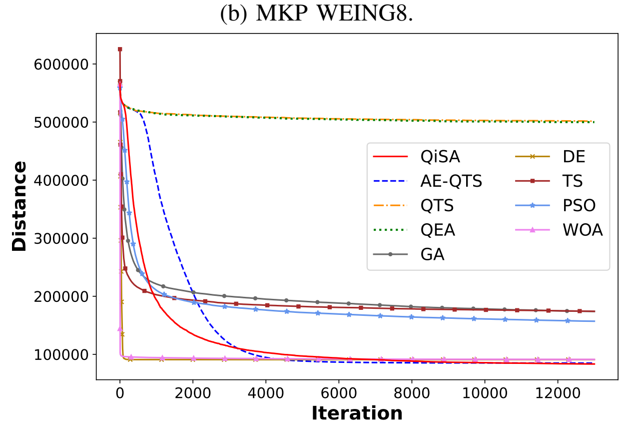
(a) KP CASE3 3000 items.



(b) MKP WEING8.



(c) UWLP MR5.



(d) TSP Gr137.

Fig. 4: QiSA and AE-QTS on selected combinatorial optimization benchmarks.

[4] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter control in evolutionary algorithms: Trends and challenges," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 167–187, 2015.

[5] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[6] M. Pant, H. Zaheer, L. Garcia-Hernandez, A. Abraham *et al.*, "Differential evolution: A review of more than two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 90, 103479, 2020.

[7] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *IEEE Congress on Eevolutionary Computation*, 2013, pp. 71–78.

[8] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *IEEE Congress on Evolutionary Computation*, 2017, pp. 1311–1318.

[9] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, 2003.

[10] F. S. Gharehchopogh, "Quantum-inspired metaheuristic algorithms: Comprehensive survey and classification," *Artificial Intelligence Review*, vol. 56, no. 6, pp. 5479–5543, 2023.

[11] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *ACM Symposium on Theory of Computing*, 1996, pp. 212–219.

[12] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 580–593, 2002.

[13] H.-P. Chiang, Y.-H. Chou, C.-H. Chiu, S.-Y. Kuo, and Y.-M. Huang, "A quantum-inspired tabu search algorithm for solving combinatorial optimization problems," *Soft Computing*, vol. 18, no. 9, pp. 1771–1781, 2014.

[14] K.-C. Tseng, W.-C. Lai, I.-C. Chen, Y.-H. Hsiao, J.-Y. Chiue, and W.-C. Huang, "Amplitude-ensemble quantum-inspired tabu search algorithm for solving 0/1 knapsack problems," in *IEEE International Conference on Systems, Man, and Cybernetics*, 2024, pp. 718–725.

[15] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[16] R. M. Karp, "Reducibility among combinatorial problems," in *50 Years of Integer Programming 1958-2008: From the Early Years to the State-*

TABLE II: Ranking of all algorithms on four combinatorial optimization problems.

Problem		QiSA	AE-QTS	QTS	QEA	GA	DE	TS	PSO	WOA
KP	avg. error	0.009%	0.002%*	0.369%	2.28%	0.446%	0.635%	1.285%	0.9%	2.031%
	rank	2	1*	3	9	4	5	7	6	8
MKP	avg. error	0.271%	0.251%	0.935%	0.561%	0.197%	0.023%*	0.648%	0.093%	0.362%
	rank	5	4	9	7	3	1*	8	2	6
UWLP	avg. error	0.145%	0.069%	2.405%	122.848%	0.591%	0.381%*	1.092%	0.16%	0.608%
	rank	3	2	8	9	5	1*	7	4	6
TSP	avg. error	11.15%	9.969%*	131.613%	170.216%	52.994%	38.746%	61.543%	45.188%	27.133%
	rank	2	1*	4	8	7	3	9	6	4

TABLE III: Computational efficiency and statistical stability of QiSA across KP, MKP, UWLP, and TSP benchmarks.

KP Benchmarks [16] (Avg. Eval. Imp.: 8.8% & Avg. CV: 0.616%)

CASE1 (Avg. CV: 0.642%)				CASE2 (Avg. CV: 1.776%)			
100 (CV: 1.547%)	**500 (CV: 0.745%)**	**1000 (CV: 0.005%)**	**3000 (CV: 0.269%)**	**100 (CV: 3.841%)**	**500 (CV: 1.54%)**	**1000 (CV: 1.107%)**	**3000 (CV: 0.617%)**
340136 (-240.1%) 610.938±9.452	256682 (-156.6%) 3068.108±22.542	180252 (-80.2%) 6137.958±30.228	74312* (25.6%) 18412.398±49.576	28370* (71.6%) 471.76±18.12	40836* (59.1%) 2369.775±36.501	47792* (52.2%) 4742.464±52.515	57218* (42.7%) 14188.771±87.475
CASE3 (Avg. CV: 0.031%)				**CASE4 (Avg. CV: 0.017%)**			
100 (CV: 0%)	**500 (CV: 0.067%)**	**1000 (CV: 0.036%)**	**3000 (CV: 0.022%)**	**100 (CV: 0.017%)**	**500 (CV: 0.024%)**	**1000 (CV: 0.013%)**	**3000 (CV: 0.012%)**
31978* (68.0%) 620±0	40942* (59.0%) 3103.9±2.071	43402* (56.5%) 6208.85±2.22	53292* (46.7%) 18626.8±4.155	50458* (49.5%) 2874.95±0.497	60510* (39.4%) 14384.4±3.412	68132* (31.8%) 28769.65±3.623	84230* (15.7%) 86309.4±10.776

MKP Benchmarks [17] (Avg. Eval. Imp.: 56.33% & Avg. CV: 0.183%)

HP1 (CV: 0.31%)	HP2 (CV: 0.339%)	PET2 (CV: 0%)	PET3 (CV: 0%)	PET4 (CV: 0%)	PET5 (CV: 0.011%)	PET6 (CV: 0.029%)	PET7 (CV: 0.055%)
Opt.: 3418 $err.$: 0.144% 49366* (50.6%) 3413.07±10.58	Opt.: 3186 $err.$: 0.761% 29758* (70.2%) 3161.76±10.705	Opt.: 87061 $err.$: 0% 14240* (85.7%) 87061±0	Opt.: 4015 $err.$: 0% 15652* (84.3%) 4015±0	Opt.: 6120 $err.$: 0% 21352* (78.6%) 6120±0	Opt.: 12400 $err.$: 0.002% 23874* (76.1%) 12399.8±1.4	Opt.: 10618 $err.$: 0.135% 34816* (65.1%) 10603.66±3.086	Opt.: 16537 $err.$: 0.032% 34874* (65.1%) 16531.66±9.073
WEING1 (CV: 0.035%)	**WEING2 (CV: 0.012%)**	**WEING3 (CV: 0.065%)**	**WEING4 (CV: 0.021%)**	**WEING5 (CV: 0%)**	**WEING6 (CV: 0.065%)**	**WEING7 (CV: 0.053%)**	**WEING8 (CV: 0.478%)**
Opt.: 141278 $err.$: 0.004% 20574* (79.4%) 141273±49.749	Opt.: 130883 $err.$: 0.001% 22546* (77.4%) 130881.4±15.92	Opt.: 95677 $err.$: 0.048% 23494* (76.5%) 95631.27±62.059	Opt.: 119337 $err.$: 0.002% 43908* (56.0%) 119334.51±24.775	Opt.: 98796 $err.$: 0% 20938* (79.0%) 98796±0	Opt.: 130623 $err.$: 0.015% 21752* (78.2%) 130603.5±84.999	Opt.: 1095445 $err.$: 0.027% 34106* (65.8%) 1095148.66±575.766	Opt.: 624319 $err.$: 1.127% 102054 (-2.0%) 617281.67±2951.571
PB1 (CV: 0.315%)	**PB2 (CV: 0.333%)**	**PB4 (CV: 1.374%)**	**PB5 (CV: 0.156%)**	**PB6 (CV: 0.278%)**	**PB7 (CV: 0.033%)**	**SENTO1 (CV: 0.307%)**	**SENTO2 (CV: 0.111%)**
Opt.: 3090 $err.$: 0.169% 36568* (63.4%) 3084.78±9.726	Opt.: 3186 $err.$: 0.421% 19656* (80.3%) 3172.58±10.551	Opt.: 95168 $err.$: 2.988% 79610* (20.3%) 92323.99±1268.333	Opt.: 2139 $err.$: 0.032% 56752* (43.2%) 2138.32±3.331	Opt.: 776 $err.$: 0.057% 184662 (-84.6%) 775.56±2.156	Opt.: 1035 $err.$: 0.006% 92440* (7.5%) 1034.94±0.341	Opt.: 7772 $err.$: 0.33% 31468* (68.5%) 7746.35±23.819	Opt.: 8722 $err.$: 0.208% 32590* (67.4%) 8703.87±9.632

UWLP Benchmarks [17] (Avg. Eval. Imp.: 76.92% & Avg. CV: 0.647%)

cap71 (CV: 0%)	cap72 (CV: 0%)	cap73 (CV: 0%)	cap74 (CV: 0.026%)	cap101 (CV: 0.011%)	cap102 (CV: 0.03%)	cap103 (CV: 0.067%)	cap104 (CV: 0.093%)
Opt.: 932615.75 $err.$: 0% 15528* (84.4%) 932615.75±0	Opt.: 977799.4 $err.$: 0% 15442* (84.5%) 977799.4±0	Opt.: 1010641.45 $err.$: 0% 17424* (82.5%) 1010641.45±0	Opt.: 1034976.975 $err.$: 0.003% 15722* (84.2%) 1035004.376±272.637	Opt.: 796648.437 $err.$: 0.001% 16738* (83.2%) 796657.04±85.598	Opt.: 854704.2 $err.$: 0.007% 19828* (80.1%) 854762.528±258.228	Opt.: 893782.112 $err.$: 0.013% 22534* (77.4%) 893894.238±603.288	Opt.: 928941.75 $err.$: 0.009% 18948* (81.0%) 929028.864±866.777
cap131 (CV: 0.053%)	**cap132 (CV: 0.068%)**	**cap133 (CV: 0.045%)**	**cap134 (CV: 0.218%)**		**capa (CV: 0.362%)**	**capb (CV: 0.462%)**	**capc (CV: 0.038%)**
Opt.: 793439.562 $err.$: 0.011% 18386* (81.6%) 793526.757±416.763	Opt.: 851495.325 $err.$: 0.045% 20464* (79.5%) 851874.574±576.487	Opt.: 893076.712 $err.$: 0.098% 23126* (76.8%) 893949.266±402.556	Opt.: 928941.75 $err.$: 0.1% 20420* (79.5%) 929871.124±2028.766		Opt.: 17156454.478 $err.$: 0.041% 25040* (74.9%) 17163423.739±62217.108	Opt.: 12979071.582 $err.$: 0.777% 28476* (71.5%) 13079866.681±60431.55	Opt.: 11505594.329 $err.$: 0.518% 29882* (70.1%) 11565221.076±44115.59
MO1 (CV: 0%)	**MO2 (CV: 0%)**	**MO3 (CV: 0.089%)**	**MO4 (CV: 5.741%)**	**MO5 (CV: 0.074%)**	**MP1 (CV: 0.913%)**	**MP2 (CV: 2.653%)**	**MP3 (CV: 0.101%)**
Opt.: 1305.95141 $err.$: 0% 20524* (79.4%) 1305.95141±0	Opt.: 1432.35732 $err.$: 0% 21134* (78.8%) 1432.35732±0	Opt.: 1516.773 $err.$: 0.028% 22562* (77.4%) 1517.19598±1.345	Opt.: 1442.23643 $err.$: 0.58% 20460* (79.5%) 1450.60584±83.275	Opt.: 1408.76638 $err.$: 0.045% 22516* (77.4%) 1409.39993±1.042	Opt.: 2686.47946 $err.$: 0.155% 23740* (76.2%) 2690.63133±24.571	Opt.: 2904.85901 $err.$: 0.285% 23922* (76.0%) 2913.14033±77.288	Opt.: 2623.70888 $err.$: 0.01% 22936* (77.0%) 2623.97445±2.642
MP4 (CV: 0.778%)	**MP5 (CV: 0.113%)**	**MQ1 (CV: 0.23%)**	**MQ2 (CV: 0%)**	**MQ3 (CV: 1.052%)**	**MQ4 (CV: 2.344%)**	**MQ5 (CV: 0.233%)**	**MR1 (CV: 5.624%)**
Opt.: 2938.75002 $err.$: 0.161% 24600* (75.4%) 2943.49313±22.904	Opt.: 2932.33111 $err.$: 0.067% 24646* (75.3%) 2934.30077±3.324	Opt.: 4091.00946 $err.$: 0.023% 24966* (75.0%) 4091.95688±9.427	Opt.: 4028.32573 $err.$: 0% 25502* (74.4%) 4028.32573±0	Opt.: 4275.43167 $err.$: 0.108% 25034* (74.9%) 4280.04763±45.025	Opt.: 4235.14701 $err.$: 0.297% 24864* (75.1%) 4247.70839±99.546	Opt.: 4080.7429 $err.$: 0.373% 26360* (73.6%) 4095.95326±9.559	Opt.: 2608.148329 $err.$: 0.606% 29062* (70.9%) 2623.964434±147.583
MR2 (CV: 0.75%)		**MR3 (CV: 0.11%)**		**MR4 (CV: 0.223%)**		**MR5 (CV: 0.14%)**	
Opt.: 2654.734663, $err.$: 0.489% 28898* (71.1%), 2667.719327±20.011		Opt.: 2788.250185, $err.$: 0.058% 28126* (71.8%), 2789.875989±3.08		Opt.: 2756.038599, $err.$: 0.086% 29132* (70.8%), 2758.406332±6.139		Opt.: 2505.047753, $err.$: 0.085% 28872* (71.1%), 2507.165555±3.522	

TSP Benchmarks [18] (Avg. Eval. Imp.: -4.47% & Avg. CV: 3.45%)[1]

burma14 (CV: 0.363%, Opt.: 3323)	ulysses16 (CV: 0.552%, Opt.: 6859)	ulysses22 (CV: 1.242%, Opt.: 7013)	eil51 (CV: 4.11%, Opt.: 426)	berlin52 (CV: 3.763%, Opt.: 7542)
$err.$: 0.037%, 3324.24±12.082 263935/200000 (-31.9%)	$err.$: 0.51%, 6894±38.056 388892/200000 (-94.4%)	$err.$: 1.2%, 7097.18±88.179 636148/400000 (-59.0%)	$err.$: 10.89%, 472.39±19.415 153406*/1000000 (84.6%)	$err.$: 12.732%, 8502.28±319.944 1025054/1000000 (-2.5%)
eil76 (CV: 2.845%, Opt.: 538)	**pr76 (CV: 4.287%, Opt.: 108159)**	**gr96 (CV: 4.615%, Opt.: 55209)**	**eil101 (CV: 3.568%, Opt.: 629)**	**gr137 (CV: 9.155%, Opt.: 55209)**
$err.$: 12.816%, 606.95±17.266 350840*/1400000 (74.9%)	$err.$: 15.001%, 124383.93±5332.906 3573784/1400000 (-155.2%)	$err.$: 21.305%, 66971.46±3090.8 338013*/1800000 (81.2%)	$err.$: 17.677%, 740.19±26.408 362600*/2000000 (81.8%)	$err.$: 19.33%, 83355.71±7631.039 3231691/2600000 (-24.2%)

Note: The evaluation limits are set to 100000 (200×500) for KP [16], MKP [17], and UWLP [17], and specific limits for TSP [18].
The average evaluation improvement is defined as Avg. Eval. Imp. $= \frac{1}{N}\sum_{i=1}^{N}\left(\frac{L_i - E_i}{L_i}\right) \times 100\%$, where E_i is the number of evaluations for QiSA, L_i is the evaluation limit for the i-th instance, and N is the total number of instances. CV denotes the Coefficient of Variation. Opt. stands for the optimal solution provided by the benchmark dataset. The metric $err.$ represents the average percentage gap from the known optimal solution.

* The symbol '*' indicates that the number of evaluations upon QiSA's automatic termination is lower than that of other algorithms.

[1] For TSP benchmarks [18], the values are presented as "Mean Evaluations / Max Evaluations assigned to other algorithms".

of-the-Art. Springer, 2009, pp. 219–241.

[17] J. E. Beasley, "OR-Library: Distributing test problems by electronic mail," *Journal of the operational research society*, vol. 41, no. 11, pp. 1069–1072, 1990.

[18] G. Reinelt, "TSPLIB95," *Interdisziplinäres Zentrum für Wissenschaftliches Rechnen, Heidelberg*, vol. 338, pp. 1–16, 1995.

[19] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–73, 1992.

[20] F. Glover, "Tabu search—part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.

[21] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

[22] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

APPENDIX

QiSA is publicly available under the MIT License at: https://github.com/kctseng-niu/Quantum-inspired-Search-Algorithm