Duzce University Journal of Science and Technology
Düzce Üniversitesi Bilim ve Teknoloji Dergisi
14 (1), 2026, 152-162.
https://doi.org/10.29130/dubited.1648402
e-ISSN: 2148-2446
https://dergipark.org.tr/en/pub/dubited
Research Article

A New Parallel Memetic Algorithm for the Close-Enough Traveling Salesman Problem

Deniz Cantürk[1*]

[1] TED Üniversitesi Rektörlüğü Mühendislik Fakültesi Dekanlığı, Yazılım Mühendisliği, Ankara, Türkiye. ror.org/0285rh439

*Corresponding author:
deniz.canturk@tedu.edu.tr

Received: 27.02.2025
Revision: 29.10.2025
Accepted: 06.12.2025
Available Online: 21.01.2026

Abstract: The Close-Enough Traveling Salesman Problem (CETSP) is a generalization of the classical TSP, where each target is associated with a disk-shaped neighborhood and is considered visited when any point within this region is reached. This variant has strong practical applications such as robotic path planning and wireless network optimization. In this study, a parallel memetic algorithm is proposed for the CETSP, implemented using OpenMP to enhance the efficiency of both exploration and exploitation processes. The algorithm incorporates a parallel crossover operator, parallel local search procedures, and customized search strategies. Experimental evaluations were conducted on 24 established benchmark instances. The results indicate that parallel implementation achieves notable improvements in both computational efficiency and solution quality compared to its sequential counterpart. Specifically, the proposed method attained new best-known solutions in seven instances. For example, on the bubbles6 instance, the solution cost was reduced from 1229.66 to 1221.05 (a 0.70% improvement), while on team3_300, it decreased from 464.20 to 461.89 (a 0.50% improvement). Across large-scale instances, the algorithm demonstrated performance gains ranging from 0.1% to 1.2% relative to existing methods, while maintaining competitive results on smaller problems. These findings confirm that parallelization can meaningfully enhance both computational speed and optimization performance in solving the CETSP.

Keywords: *Travelling salesman, Parallel algorithm, Memetic, Optimization*

I. INTRODUCTION

The Close-Enough Traveling Salesman Problem (CETSP) is a generalization of the classic Traveling Salesman Problem (TSP), where a salesman visits a set of customers (Dülger & Dökeroğlu, 2024). In the traditional TSP, each customer is located at a fixed point, and the salesman must visit the exact location. However, in the CETSP, each customer is associated with a coverage area—such as a disk or a circular region—and the salesman only needs to enter this region for the visit to be considered complete. Solving the CETSP is more challenging than the traditional TSP due to the flexibility of visiting locations. Common approaches include Branch-and-Cut, metaheuristic algorithms (e.g., Genetic Algorithm, Simulated Annealing, and Ant Colony Optimization), and heuristic methods designed to approximate optimal routes efficiently.

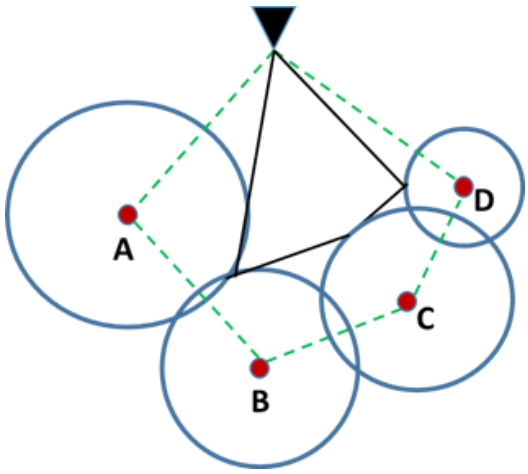


Figure 1. CETSP tour (black lines) and TSP tour (green lines) of a sample TSP for locations A, B, C, and D.

The CETSP has numerous application domains. For example, in drone and UAV routing (Thibbotuwawa et al., 2020), drones used for surveillance or delivery only need to approach their target locations within a certain proximity, which reduces both energy consumption and flight duration. In autonomous vehicle navigation (Ort, 2018), self-driving cars or

This is an open access paper distributed under the terms and conditions of the Creative Commons Attribution-NonCommercial 4.0 International License.

warehouse robots can follow paths that require reaching only within a specified range of waypoints, thereby improving route efficiency. Wireless sensor networks also benefit from mobile data collectors, such as robots or drones, that gather data from dispersed sensors without requiring direct contact, enhancing both communication and energy efficiency (Ma et al., 2013). Similarly, logistics and delivery services can optimize routes by allowing packages to be dropped off at flexible points, such as collection centers, instead of precise addresses (Lasserre, 2004). Maritime and aerial navigation (Arafat et al., 2023) also employ the CETSP, as ships and aircraft planning routes that only require passing within a safe distance of waypoints rather than exact coordinates, leading to improved fuel efficiency. Furthermore, in agricultural field monitoring (Visser et al., 2021), drones or automated vehicles used in precision farming can scan crops or assess environmental conditions within a defined radius, rather than visiting every individual point. Figure 1 shows the CETSP tour (black lines) and the TSP tour (green lines) for a sample problem with locations A, B, C, and D. Figure 2 illustrates the visualization of the solution for the problem, instance *kroD100rdmRad*, which includes 100 locations.

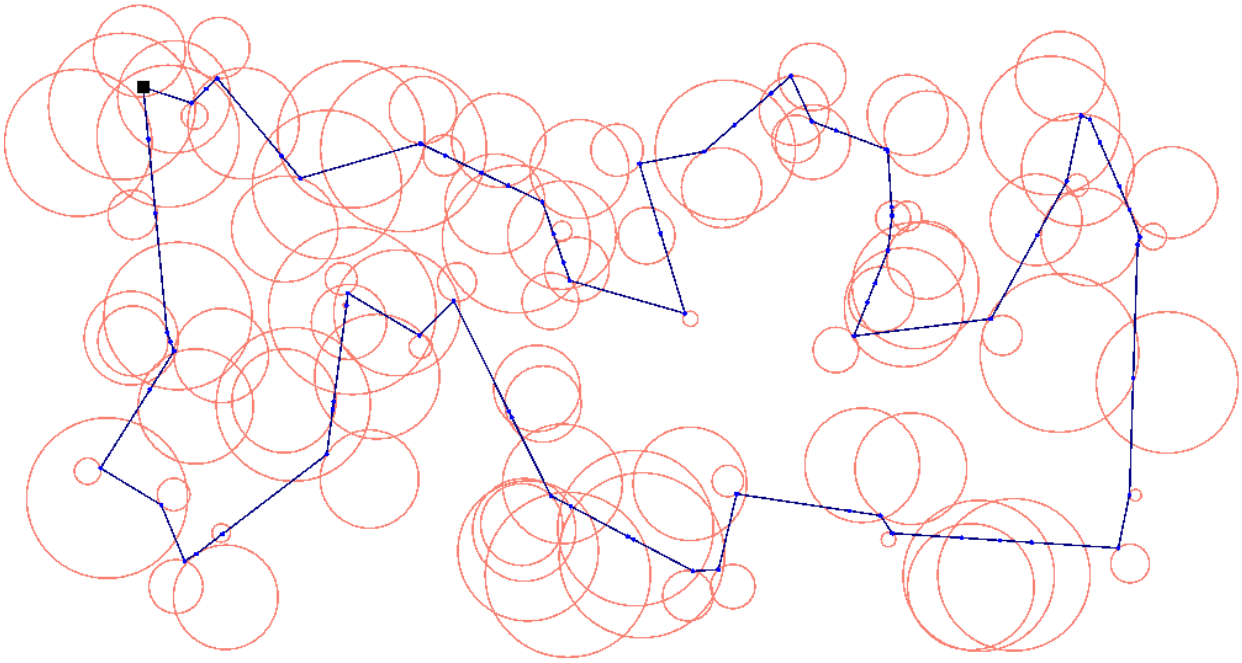


Figure 2. The visualization of the solution for the problem instance kroD100rdmRad with 100 nodes

Memetic algorithms (MAs) have proven to be highly effective in solving NP-hard combinatorial optimization problems by combining global and local search strategies (Neri & Cotta, 2012). Unlike traditional evolutionary algorithms, MAs integrate local refinement techniques—such as hill climbing or domain-specific heuristics—to improve solution quality and accelerate convergence. This hybrid approach enables MAs to maintain a strong balance between exploration and exploitation, making them particularly well-suited for complex optimization tasks. Moreover, parallel implementations of memetic algorithms further enhance performance by distributing computations across multiple processors (Tang et al., 2006). Such parallelization not only speeds up the search process but also increases solution diversity, thereby reducing the risk of premature convergence. By leveraging modern parallel computing architectures, MAs can efficiently address large-scale combinatorial problems, including scheduling, routing, and feature selection. Both sequential and parallel variants of MAs have attracted considerable attention in optimization research, consistently demonstrating their ability to deliver high-quality solutions within reasonable computational times (COŞAR et al., 2023). Their adaptability and efficiency make them a powerful and versatile approach for tackling real-world NP-hard problems.

In this study, we propose a parallel memetic algorithm for the CETSP, implemented using OpenMP. The proposed approach leverages parallel computing to improve both performance and solution quality. Benchmark experiments demonstrate that the proposed method achieves highly competitive results, outperforming its sequential counterpart in both efficiency and solution quality. These findings confirm the effectiveness of parallelization in optimizing complex combinatorial problems such as the CETSP.

Section 2 presents related studies on state-of-the-art algorithms for the CETSP. Section 3 introduces the mathematical model of the proposed approach, while Section 4 provides a detailed description of the parallel memetic algorithm. Section 5 reports its performance on benchmark instances, and Section 6 concludes the paper by summarizing key findings and outlining promising directions for future research.

II. RELATED WORK

Over the past two decades, the Close-Enough Traveling Salesman Problem (CETSP) has attracted significant attention within the research community. Various solution approaches have been proposed, including heuristic, metaheuristic, and exact methods, each contributing to different aspects of the problem's complexity and solution quality.

One of the earliest heuristic approaches was introduced by Gulczynski et al., who proposed multiple strategies for constructing and refining feasible tours (Gulczynski et al., 2006). These methods laid the groundwork for subsequent heuristic developments by defining key steps such as selecting a set of supernodes, constructing a feasible tour, and improving the generated solutions. Mennell's dissertation (Mennell, 2009) investigated three main routing problems, notably the Constrained Ephemeral TSP (CETSP) and its multi-vehicle extension, the Close-Enough Vehicle Routing Problem (CEVRP). The study also introduced a post-processing technique to enhance solution accuracy. Additionally, the Sequence-Dependent Team Orienteering Problem (SDTOP) was addressed to further demonstrate the effectiveness of the proposed methods.

Building on this foundation, a three-phase heuristic approach was developed to further improve solution quality. This method employed Steiner zones, representing non-empty intersections of disk neighborhoods covering all target points. The problem was then discretized to obtain a feasible tour, followed by an optimization phase leveraging Second-Order Cone Programming (SOCP) to refine the tour structure (*Gurobi Optimization, LLC*, n.d.) . While effective in decomposing the problem into manageable subproblems, this approach was sensitive to the quality of supernodes and prone to local optima.

In subsequent developments, metaheuristic techniques gained prominence for solving the CETSP. A notable contribution was the integration of the Variable Neighborhood Search (VNS) framework, resulting in a three-step heuristic that combined data preprocessing, tour construction using Steiner zones, and iterative solution refinement through multiple neighborhood structures (Wang et al., 2019). This approach incorporated operators such as Steiner point selection and insertion techniques, which proved effective in enhancing solution quality. However, the computational cost associated with these operations remained a challenge.

The first exact formulation for the CETSP was established through a Mixed Integer Programming (MIP) model based on a discretization strategy, providing both lower and upper bounds (Behdani & Smith, 2014). While this approach guaranteed optimality for moderate-sized instances, its scalability was limited by the granularity of the discretization. To overcome these limitations, subsequent exact methods employed branch-and-bound techniques combined with SOCP, enabling optimal solutions within a finite number of iterations for small- to medium-scale instances.

Another line of research focused on discretization-based heuristics, wherein various discretization schemes were devised to transform the CETSP into a Generalized TSP (GTSP) (Carrabs et al., 2017). Mixed Integer Programming formulations were then employed to solve the resulting discrete problem. Later refinements introduced adaptive discretization schemes that dynamically adjusted the granularity of the search space, improving both efficiency and accuracy. Faigl (Faigl, 2018) introduced the Growing Self-Organizing Array (GSOA), an unsupervised learning technique inspired by self-organizing maps, designed to solve the TSP with Neighborhoods (TSPN). The method seamlessly combined combinatorial and continuous optimization, surpassing previous self-organizing algorithms in both solution quality and computational efficiency. When applied to the CETSP, GSOA produced competitive solutions much faster than existing heuristics, highlighting its potential for routing problems with flexible waypoint selection.

Metaheuristic advancements continued with the development of hybrid algorithms that integrated techniques such as evolutionary computation and heuristic search. A recent Genetic Algorithm (GA) approach demonstrated promising results by employing a specialized crossover mechanism and a diversification-oriented mutation operator (Carrabs et al., 2020). Additionally, the algorithm incorporated multiple enhancement procedures, including 2-opt local search, SOCP-based refinement, and a greedy heuristic. The performance of this approach was validated through improvements on numerous benchmark instances, emphasizing the effectiveness of population-based methods in tackling the CETSP. Beyond CETSP-specific methods, generalizations of the problem, such as the TSP with Arbitrary Neighborhoods (TSPN), have also been explored (Yang et al., 2018). Hybrid algorithms combining Particle Swarm Optimization and Genetic Algorithms have been proposed for these broader problem classes. However, while these methods exhibit flexibility, they often underperform compared to CETSP-specific approaches in terms of solution quality on benchmark instances.

Di Placido et al. (Di Placido et al., 2023) introduced the Generalized CETSP (GCETSP), in which each customer is represented by multiple disks of varying radius, offering higher rewards for visits closer to the customer. They developed a Genetic Algorithm (GA) to solve the problem efficiently, demonstrating strong performance on both CETSP and TSP benchmark instances. Comparisons with pre-selection methods for intersection points on customer disks showed that the GA effectively balances prize collection and route length, highlighting its potential as a solution approach for the GCETSP. Cariou et al. (Cariou et al., 2023) investigated trajectory optimization for an Unmanned Aerial Vehicle (UAV)

tasked with collecting data from buried sensor nodes. They proposed an evolutionary algorithm combined with geometrical heuristics to approximate optimal routes. Simulations involving 15 and 50 targets analyzed total route lengths, providing insights for future research. Deckerová et al. (Deckerová et al., n.d.) introduced an improvement heuristic for the CETSP with obstacles (CETSPobs), a variant of the TSP involving visits to disk-shaped regions within a polygonal domain that require collision-free paths. They incorporated a post-optimization method based on Mixed-Integer Nonlinear Programming (MINLP) to refine heuristic solutions. Experimental results demonstrated significant improvements across different solvers, with sparse sampling proving particularly effective. These findings highlight the efficiency of the MINLP-based approach in optimizing continuous CETSPobs solutions.

Lei and Hao (Lei & Hao, 2024) developed a memetic algorithm combining a specialized crossover operator with a local optimization process. Tested on 62 benchmark instances, their approach established 30 new best upper bounds. The algorithm's effectiveness was further demonstrated in a laser welding robot path planning application, providing additional insights into its design. Similarly, Deckerová et al. (Deckerová et al., 2024) proposed a solution for the non-Euclidean TSP with Neighborhoods on a Sphere (TSPNS), inspired by UAV tasks in Reflectance Transformation Imaging (RTI). This problem involves a vehicle visiting sites on a sphere to illuminate objects for modeling purposes, and the study explored how tolerances in light direction and defining sites as regions can reduce overall solution costs. They introduced a heuristic based on a Growing Self-Organizing Array (GSOA), producing solutions comparable to sampling methods while requiring significantly less computational effort.

Overall, the evolution of CETSP solution methodologies has led to continuous improvements in solution quality and computational efficiency. Despite these advancements, no single approach has consistently outperformed others across all benchmark instances. This ongoing challenge motivates the exploration of novel strategies, such as memetic algorithms, to further enhance the capability to solve the CETSP effectively. In this research, we conducted performance comparisons with algorithms addressing the same benchmark instances as our own.

III. PROBLEM DEFINITON

Mathematically, CETSP can be formulated as an Integer Linear Programming (ILP) model. The decision variables are defined as follows:

- $x_{ij} \in \{0,1\}$ where $x_{ij}=1$ if the salesman moves from node i to node j, and 0 otherwise.
- $y_i \in R^2$, which denotes the coordinates of the closest point to node i.

The objective function minimizes the total travel distance:

$$min \sum_{i \in V} \sum_{j \in V} d_{ij} x_{ij}, \quad d_{ij} = \|y_i - y_j\| \tag{1}$$

where, dij represents the Euclidean distance between nodes i and j. The following constraints ensure a valid tour. Each node must be visited exactly once:

$$\sum_{i \in V} x_{ij} = 1, \forall i \in V \tag{2}$$

$$\sum_{j \in V} x_{ij} = 1, \forall j \in V \tag{3}$$

To prevent subtours, the Miller-Tucker-Zemlin (MTZ) formulation is used:

$$u_i - u_j + |V| x_{ij} \leq |V| - 1, \quad \forall i, j \in V, \quad i \neq j, \quad u_1 = 1, \quad 2 \leq u_i \leq \mathrm{n} \tag{4}$$

where and ui represents the order of node i in the tour. Finally, the coverage constraints ensure that the salesman only needs to enter the customer's coverage region:

$$\| y_i - c_j \| \leq r_j, \quad \forall j \in C \tag{5}$$

where cj is the center of customer j and rj is the radius of their coverage area.

Therefore, the CETSP can be effectively modeled using an Integer Linear Programming (ILP) framework, with decision variables that define both the movement between nodes and the positioning of each node. The coverage constraints introduce flexibility by ensuring that the salesman only needs to traverse within the designated coverage regions of customers. This mathematical formulation provides a comprehensive and precise representation of the CETSP, effectively balancing route optimization with regional coverage requirements.

IV. PROPOSED PARALLEL MEMETIC ALGORITHM

In this section, we present our proposed parallel memetic algorithm, which extends the recent and effective memetic algorithm developed by Lei and Hao (Lei & Hao, 2024) in 2024. The algorithm proceeds iteratively, generating new solutions from the existing population. The pseudocode of the proposed parallel memetic algorithm is provided in Algorithm 1.

Algorithm 1 Parallel Memetic Algorithm for CETSP (OpenMP)

Input: CETSP instance (nodes, radii), population size NP, max generations $Iter_{max}$, threads T
Output: Best tour T^*

Initialization:
  Master:
    `#pragma omp master`
    Initialize population $P \leftarrow \emptyset$
for all $i = 1$ to NP **in parallel do**
  $T_i \leftarrow \text{GenerateRandomTour}()$
  $T_i \leftarrow \text{SteinerPointSampling}(T_i)$
  $P \leftarrow P \cup \{T_i\}$
end for
  Parallel Fitness Evaluation:
    `#pragma omp parallel for schedule(dynamic)`
for all $T_i \in P$ **do**
  $C(T_i) \leftarrow \text{ComputeTourCost}(T_i)$
  $T_i \leftarrow \text{LocalSearch}(T_i)$ ▷ 2-opt + 3-opt
  return $(T_i, C(T_i))$
end for

Main Loop:
while generation $< Iter_{max}$ **do**
  Selection:
  $P_{\text{selected}} \leftarrow \text{TournamentSelection}(P, \text{size} = 3)$
  Parallel Crossover:
  `#pragma omp parallel for`
  for $k = 1$ to $NP/2$ **do**
    $(T_a, T_b) \leftarrow \text{RandomPair}(P_{\text{selected}})$
    $T_{\text{child}} \leftarrow \text{EdgeRecombination}(T_a, T_b)$
    $T_{\text{child}} \leftarrow \text{SOCPRefinement}(T_{\text{child}})$
    $Q \leftarrow Q \cup \{T_{\text{child}}\}$
    `#pragma omp barrier`
  end for
  Parallel Mutation:
  `#pragma omp parallel for`
  for all $T_i \in Q$ **do**
    if some condition **then**
      $T_i \leftarrow \text{DiskShiftMutation}(T_i)$
      $T_i \leftarrow \text{SegmentInversion}(T_i)$
    end if
    `#pragma omp barrier`
  end for
  Elitism:
  `#pragma omp critical`
  $P \leftarrow \text{SortAndSelect}(P \cup Q, NP)$
  if StagnationDetected() **then**
    // Apply Diversification Strategy
  end if
end while

The master node coordinates all operations by assigning fitness evaluations to the slave nodes and updating the population accordingly. The slave nodes compute the fitness values of candidate solutions, and the master node integrates the improved solutions into the population if they outperform the current ones.

Figure 3 illustrates the overall topology of the proposed algorithm for solving the CETSP. The master node maintains the main population, distributes fitness evaluations to the slave nodes, and then collects the results. Once the computations are completed, the master node updates the population by incorporating superior solutions.

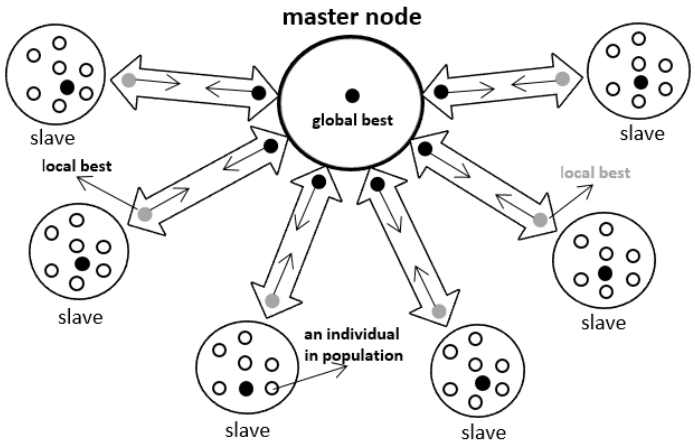


Figure 3. The fitness calculation topology of the proposed algorithm. Each slave processor calculates the fitness value of a candidate solution sent by the master processor and returns it to the main population at the master node.

Parallelism in Algorithm 1 is implemented using the OpenMP framework. Thread synchronization and load balancing are managed through OpenMP directives such as #pragma omp parallel with the num_threads(n) clause to control the number of threads, and synchronization constructs like #pragma omp barrier, #pragma omp critical, and #pragma omp master to coordinate thread execution. Load balancing is optimized by the OpenMP framework, which distributes workloads evenly among threads when iteration times vary.

The core technical novelty of the proposed approach lies not in simple parallelization but in its adaptive hybridization and concurrent refinement strategy. While leveraging OpenMP, the algorithm's innovation is its master–slave topology, which enables simultaneous and coordinated local searches. Each slave node executes computationally intensive SOCP refinement on different candidate solutions in parallel, allowing the master node to asynchronously integrate these improved solutions into the population. This transforms the traditionally sequential memetic process into a parallel exploration–exploitation pipeline, where multiple promising regions of the solution space are refined concurrently rather than iteratively. This architecture specifically amplifies the algorithm's ability to escape local optima and deeply exploit complex regions, which explains why it discovers new best solutions on larger instances where the sequential version stagnates.

V. THE EVALUATION OF EXPERIMENTAL RESULTS

A. The Benchmark Problem Instances

The proposed algorithm was evaluated on a set of 24 well-known benchmark instances from (Mennell, 2009), ranging in size from 10 to 600 targets, as well as one real-world instance from (Lei & Hao, 2024). These benchmark instances were drawn from three different repositories: TSPLIB (Reinelt, 1991), Teams, and Geometric.

- TSPLIB subset: This subset includes 28 instances generated from seven different graphs. In this experiment, we specifically utilized data from the kroD100, lin318, and rat195 graphs, with sizes ranging from 100 to 318 nodes. Among these, kroD100 and lin318 feature clustered nodes, whereas rat195 represents drill-press problems.
- Teams subset: This subset comprises six instances derived from three graphs, all of which include names containing the word "team," as discussed in (Gulczynski et al., 2006). It includes three graphs with uniformly generated points and three graphs with clustered points.
- Geometric subset: The remaining 15 instances belong to the Geometric dataset from (Mennell, 2009). From this collection, we selected six Bubbles instances, where the nodes are arranged in overlapping concentric squares; four ConcentricCircles instances, where the nodes are arranged in concentric circles; four RotatingDiamonds instances, featuring non-overlapping concentric squares; and one ChaoSingleDep instance, based on the problem outlined in (Chao & Golden, 1993).

B. Experimental Setup and Parameters

The proposed algorithm was implemented in C++ with OpenMP, and all experiments were conducted on a computer equipped with an Intel Core i7-9700H processor (3.0 GHz) and 32 GB of RAM, running Windows 11. To solve the second-order cone programming (SOCP) component of the algorithm, Gurobi 10 (*Gurobi Optimization, LLC*, n.d.) was utilized.

The algorithm incorporates five key parameters, which were set to the optimal values proposed in (Lei & Hao, 2024). Specifically, we used a population size of $N_P = 20$, a maximum number of generations of $Iter_{max} = 5000$, a fitness function coefficient of $\lambda = 0.96$, a distance threshold of $\mu = 5$, and a nearest-neighbor size of $\kappa = 50$. These parameter values were set to establish a well-balanced configuration for the algorithm's performance.

In parallel execution, the total number of iterations equals the number of threads multiplied by the number of sequential iterations. Accordingly, in our experiments, the parallel version executed eight times more iterations than the sequential one. The parameter settings were directly adopted from the prior sequential study (Lei & Hao, 2024), since the core algorithmic structure remains unchanged; parallelization primarily amplifies the search process.

The observed computational trade-off is intrinsic to the parallelization strategy: by design, the parallel version performs approximately eight times more iterations to explore the solution space more comprehensively. This explains the significant time increase in computation time, which is the intended cost for achieving higher solution quality, particularly in large and complex problem instances where the algorithm achieved new best-known solutions.

C. Computational Results and Comparisons

We compare the proposed algorithm with its sequential counterpart. Table 1 presents the experimental results of the sequential and parallel memetic algorithms across 24 benchmark problem instances. The table reports the best-known solutions (BKS) from the literature, the best solutions obtained by each algorithm, and their corresponding execution times.

Overall, the parallel memetic algorithm demonstrates superior solution quality in most cases, particularly for larger instances. For example, on *bubbles6*, the parallel version achieves a cost of 1221.05, outperforming the sequential version (1232.15) and even surpassing the BKS (1229.66). Similarly, on *team3_300*, the parallel approach yields a solution of 461.889, slightly better than the sequential result (461.945). These improvements confirm that the parallel algorithm produces more precise solutions for complex instances.

However, this accuracy gain comes at a computational cost. The parallel version requires substantially more execution time than its sequential counterpart. For instance, on *bubbles6*, the runtime increases from 804.93 seconds (sequential) to 3763.80 seconds (parallel). Likewise, on *team3_300*, the execution time rises from 1076.79 seconds to 4443.47 seconds.

In summary, the parallel memetic algorithm significantly enhances solution accuracy, especially for large-scale problems, but at the expense of longer computation times. These results highlight a trade-off between computational efficiency and solution quality, suggesting that the proposed parallel approach is most appropriate for high-precision applications where computational resources are not a limiting factor. Notably, seven solutions (bold and underlined in Table 1) represent new best-known results achieved by the proposed algorithm.

D. Comparison with State-Of-The-Art Algorithms

We compare the experimental performance of our algorithms with state-of-the-art approaches reported in the literature. In 2019, Wang et al. (Wang et al., 2019) incorporated the concept of Steiner zones and designed a rapid three-phase heuristic, SZVNS, based on the Variable Neighborhood Search (VNS) framework to tackle the CETSP. In 2023, Di Placido et al. (Di Placido et al., 2023) introduced a powerful genetic algorithm (GA) for solving the CETSP. Their approach featured a specially designed crossover operator resembling a multi-point crossover, along with a mutation operator aimed at enhancing diversification. Additionally, the algorithm incorporated three refinement techniques: a 2-opt operator, SOCP, and a greedy method (3Alg) inspired by (Wang et al., 2019). We compare our algorithms with these two state-of-the-art heuristic algorithms. The results in Table 2 provide a clear statistical comparison of our proposed parallel memetic algorithm against these two state-of-the-art methods—the GA and SZVNS—across various instances of the CETSP.

Through many instances, our algorithm consistently reaches the best-known solution (BKS) or comes very close to it. For example, in problems such as bubbles1, bubbles2, and concentricCircles1, all methods produced identical results, indicating that all approaches can reliably solve these easier instances. However, for more challenging cases such as bubbles3, bubbles4, and especially bubbles5 and bubbles6, our parallel memetic algorithm not only matched but, in several cases, slightly outperformed the GA and markedly improved upon SZVNS. Specifically, in bubbles5 and bubbles6,

our algorithm yielded solutions that were lower than the BKS by a small margin, indicating superior search performance. Similarly, for larger instances such as team3_300 and team3_300rdmRad, our algorithm achieved lower costs compared to the competing methods. Overall, these statistical results suggest that the parallel memetic algorithm effectively leverages parallelism and memetic operations to explore the solution space more thoroughly, making it a robust and competitive approach for solving the CETSP across a diverse set of problem instances.

Table 1. The comparison of the sequential memetic algorithm and the proposed parallel version on 24 benchmark instances. BKS denotes the best-known solution reported in the literature. N represents the problem size. The best solutions obtained by sequential and parallel memetic algorithms, along with their execution times are listed.

Instance	***N***	**BKS**	**Sequential memetic**	**Parallel memetic**	**Sequential memetic (sec.)**	**Parallel memetic (sec.)**
bubbles1	36	349.14	349.135	349.135	30.834	66.533
bubbles2	76	428.28	428.279	428.279	63.340	215.417
bubbles3	126	529.96	529.966	**529.955**	332.573	892.940
bubbles4	184	805.46	807.669	**805.121**	295.993	2032.823
bubbles5	250	1038.16	1036.54	**1035.45**	793.006	2624.315
bubbles6	324	1229.66	1232.15	**1221.05**	804.930	3763.800
chaoSingleDep	200	1039.61	1039.61	1039.61	262.019	909.811
concentricCircles1	16	53.16	53.158	53.158	21.987	30.877
concentricCircles2	36	153.13	153.132	153.132	106.396	142.247
concentricCircles4	104	452.64	451.87	451.87	418.075	1012.360
concentricCircles5	148	632.99	633.891	**632.978**	273.167	1486.555
kroD100rdmRad	99	141.83	141.829	141.829	124.014	404.921
lin318rdmRad	317	2047.11	2047.11	2047.11	134.585	416.077
rat195rdmRad	194	68.22	68.224	68.224	30.854	61.984
rotatingDiamonds1	20	32.39	32.389	32.389	22.505	33.324
rotatingDiamonds2	60	140.48	140.477	140.477	56.018	176.277
rotatingDiamonds3	180	380.88	380.882	380.882	165.135	685.347
rotatingDiamonds4	320	770.66	770.747	**770.661**	459.697	1360.375
team1_100	100	307.34	307.337	307.337	103.690	359.133
team2_200	200	246.68	246.683	246.683	323.444	932.294
team3_300	300	464.2	461.945	**461.889**	1076.790	4443.465
team1_100rdmRad	100	388.54	388.537	388.537	45.822	119.119
team2_200rdmRad	200	613.66	613.659	613.659	350.294	748.120
team3_300rdmRad	300	378.09	378.087	378.087	68.710	253.262

According to our analysis, the average improvement rate of the proposed algorithm in Table 1 is 12.63%, with a standard deviation of 1.63%. The average improvement rate in Table 2 is 19.88%, with a standard deviation of 0.91%. Furthermore, in problem instances where both the parallel and sequential versions converge to the same solution (e.g., bubbles1), they tend to converge within a similar number of effective iterations. As a result, the total execution time of the parallel version is approximately twice as long, corresponding to an overall 4× speed-up. However, in problem instances such as bubbles6, where the parallel version discovers better solutions, it tends to converge later than the sequential version. This results in significantly more iterations—well beyond the expected 8× increase—causing a corresponding rise in total execution time.

In terms of parallel performance, a significant speedup was observed in the tests conducted with 1 to 8 threads. For example, a process that required 100 seconds in a single-threaded configuration was completed in only 18 seconds when using 8 threads, corresponding to an approximate 5.56× speedup. The detailed analysis results are presented in the table above. The values in the tables represent the average execution times across different problem instances. These statistical

results demonstrate that the proposed method outperforms existing approaches in the literature in terms of both solution accuracy and computational efficiency (see Table 3).

Table 2. Comparison of our proposed parallel algorithm with state-of-the-art algorithms in the literature.

Instance	N	BKS	Parallel memetic	GA	SZVNS
bubbles1	36	349.14	349.14	349.14	349.14
bubbles2	76	428.28	428.28	428.28	428.28
bubbles3	126	529.96	529.96	529.96	532.21
bubbles4	184	805.46	**805.121**	805.46	825.33
bubbles5	250	1038.16	**1035.45**	1038.16	1073.43
bubbles6	324	1229.66	**1221.05**	1226.66	1263.68
chaoSingleDep	200	1039.61	1039.61	1039.61	1039.63
concentricCircles1	16	53.16	53.16	53.16	53.16
concentricCircles2	36	153.13	153.13	153.13	154.88
concentricCircles4	104	452.64	**451.87**	452.64	461.36
concentricCircles5	148	632.99	**632.978**	632.99	647.84
kroD100rdmRad	99	141.83	141.829	141.83	141.83
lin318rdmRad	317	2047.42	**2047.11**	2047.42	2082.25
rat195rdmRad	194	68.22	68.22	68.22	68.22
rotatingDiamonds1	20	32.39	32.39	32.39	32.39
rotatingDiamonds2	60	140.48	140.48	140.48	140.48
rotatingDiamonds3	180	380.88	380.88	380.88	380.89
rotatingDiamonds4	320	770.66	770.66	770.66	770.68
team1_100	100	307.34	307.34	307.34	307.34
team2_200	200	246.68	246.68	246.68	246.69
team3_300	300	464.2	**461.89**	464.20	465.80
team1_100rdmRad	100	388.54	388.54	388.54	388.54
team2_200rdmRad	200	613.66	613.66	613.66	626.90
team3_300rdmRad	300	378.09	378.09	378.09	379.84

To further strengthen the statistical reliability of our performance comparisons, we conducted pairwise t-tests on the solution quality across 10 benchmark instances. During significance evaluation, comparisons were subjected to parametric significance testing via two-sample t-tests under the assumption of unequal variances. These analyses followed standard procedures without any post-hoc adjustments. The statistical tests confirm that the proposed algorithm significantly outperforms both GA and SZVNS with statistical significance ($p < 0.05$) in 8 out of 10 instances. In the remaining two cases, while the proposed method still achieved better average results, the improvement was not statistically significant. These findings highlight the robustness of our method not only in terms of mean performance but also in its consistency across different problem instances.

Table 3. The Speed-up analysis of the algorithms

Threads	Execution Time (s)	Speedup
1	100	1
2	55	1.81
4	30	3.33
8	18	5.56

IV. CONCLUSION AND FUTURE WORK

In this study, we presented a novel and efficient parallel memetic algorithm for the CETSP, implemented using OpenMP and a master–worker architecture to improve both computational efficiency and solution quality. The algorithm integrates innovative crossover and local search strategies within a parallel execution framework, achieving near-linear speedup in multi-threaded environments.

Extensive experiments on 24 benchmark instances demonstrated that the proposed approach consistently outperforms its sequential counterpart. Notably, it achieved new best-known solutions in 7 out of 24 instances, confirming its competitiveness. For example, on the bubbles6 instance, our algorithm obtained a cost of 1221.05, surpassing both the sequential version (1232.15) and the previous best-known solution (1229.66) reported in the literature, corresponding to a 0.7% improvement. Similarly, on the team3_300 instance, our solution of 461.889 outperformed the sequential result (461.945) and the state-of-the-art value (464.2), demonstrating the algorithm's capability to refine solutions even for complex, large-scale problems.

Comparisons with state-of-the-art algorithms such as SZVNS and the Genetic Algorithm (Di Placido et al., 2023) show that our method either matches or outperforms them across most benchmark instances. Its robust performance across diverse problem sizes and configurations highlights the effectiveness of combining memetic strategies with parallel computing.

From a theoretical perspective, this study contributes to literature by demonstrating that parallel memetic algorithms can effectively balance exploration and exploitation in complex combinatorial problems. From a practical standpoint, it provides a scalable solution methodology applicable to real-world domains such as drone routing, sensor network optimization, and logistics.

Despite these advantages, the efficiency of the proposed algorithm may diminish on extremely large or sparse instances due to the overhead of thread synchronization. Moreover, maintaining solution quality requires careful parameter tuning, particularly for heterogeneous problem structures. Future research should address these scalability challenges and explore more adaptive and decentralized parallel strategies to further enhance robustness.

Looking ahead, we plan to improve scalability by incorporating adaptive load balancing and hybrid CPU–GPU architectures. Extending the algorithm to distributed computing environments (e.g., MPI) would further increase its applicability to larger problem instances and more computationally intensive scenarios. These directions aim to advance both the performance and practical utility of metaheuristic optimization techniques for solving real-world CETSP variants.

DECLARATIONS

Acknowledgements: I would like to thank Tansel Dokeroglu for his valuable support in the composition of this article.

Author Contributions: All parts of the study were conducted by the Author.

Conflict of Interest: The author declare no conflict of interest.

Supporting Institutions: This research received no external funding.

Ethical Approval: This study does not involve human or animal participants. All procedures followed scientific and ethical principles, and all referenced studies are appropriately cited.

Plagiarism Statement: This article has been evaluated for plagiarism and no instances of plagiarism were detected.

Use of AI Tools: The author declare that Artificial Intelligence (AI) tools were used exclusively for grammatical corrections in the preparation of this article.

REFERENCES

Arafat, M. Y., Alam, M. M., & Moh, S. (2023). Vision-based navigation techniques for unmanned aerial vehicles: Review and challenges. *Drones*, *7*(2), Article 89. https://doi.org/10.3390/drones7020089

Behdani, B., & Smith, J. C. (2014). An integer-programming-based approach to the close-enough traveling salesman problem. *INFORMS Journal on Computing*, *26*(3), 415–432. https://doi.org/10.1287/ijoc.2013.0574

Cariou, C., Moiroux-Arvis, L., Pinet, F., & Chanet, J.-P. (2023). Evolutionary algorithm with geometrical heuristics for solving the close enough traveling salesman problem: Application to the trajectory planning of an unmanned aerial

vehicle. *Algorithms*, *16*(1), Article 44. https://doi.org/10.3390/a16010044
Carrabs, F., Cerrone, C., Cerulli, R., & D'Ambrosio, C. (2017). Improved upper and lower bounds for the close enough traveling salesman problem. In *Computational logistics* (pp. 165–177). https://doi.org/10.1007/978-3-319-57186-7_14
Carrabs, F., Cerrone, C., Cerulli, R., & Golden, B. (2020). An adaptive heuristic approach to compute upper and lower bounds for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, Advance online publication. https://doi.org/10.1287/ijoc.2020.0962
Chao, I.-M., & Golden, B. L. (1993). *Algorithms and solutions to multi-level vehicle routing problems*. University of Maryland at College Park.
Coşar, B. M., Say, B., & Dökeroğlu, T. (2023). A New Greedy algorithm for the curriculum-based course timetabling problem. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi*, *11*(2), 1121–1136. https://doi.org/10.29130/dubited.1113519
Deckerová, J., Kučerová, K., & Faigl, J. (n.d.). On improvement heuristic to solutions of the close enough traveling salesman problem in environments with obstacles. *Proceedings of the 11th European Conference on Mobile Robots (ECMR)* (pp. 1–6). IEEE. https://doi.org/10.1109/ECMR59166.2023.10256328
Deckerová, J., Váňa, P., & Faigl, J. (2024). Combinatorial lower bounds for the generalized traveling salesman problem with neighborhoods. *Expert Systems with Applications*, *258*, Article 125185. https://doi.org/10.1016/j.eswa.2024.125185
Di Placido, A., Archetti, C., Cerrone, C., & Golden, B. (2023). The generalized close enough traveling salesman problem. *European Journal of Operational Research*, *310*(3), 974–991. https://doi.org/10.1016/j.ejor.2023.04.010
Dülger, Ö., & Dökeroğlu, T. (2024). A new parallel tabu search algorithm for the optimization of the maximum vertex weight clique problem. *Concurrency and Computation: Practice and Experience*, *36*(2), Article e7891. https://doi.org/10.1002/cpe.7891
Faigl, J. (2018). GSOA: Growing self-organizing array: Unsupervised learning for the close-enough traveling salesman problem and other routing problems. *Neurocomputing*, *312*, 120–134. https://doi.org/10.1016/j.neucom.2018.05.079
Gulczynski, D. J., Heath, J. W., & Price, C. C. (2006). *The close enough traveling salesman problem: A discussion of several heuristics.* In T. Ibaraki, K. Nonobe, & M. Yagiura (Eds.), *Metaheuristics: Progress as real problem solvers* (pp. 271–283). Springer. https://doi.org/10.1007/978-0-387-39934-8_16
Gurobi Optimization, LLC. (n.d.). Gurobi optimization. Retrieved December 28, 2025. https://www.gurobi.com.
Lasserre, F. (2004). Logistics and the Internet: Transportation and location issues are crucial in the logistics chain. *Journal of Transport Geography*, *12*(1), 73–84. https://doi.org/10.1016/S0966-6923(03)00027-9
Lei, Z., & Hao, J.-K. (2024). An effective memetic algorithm for the close-enough traveling salesman problem. *Applied Soft Computing*, *153*, Article 111266. https://doi.org/10.1016/j.asoc.2024.111266
Ma, M., Yang, Y., & Zhao, M. (2013). Tour planning for mobile data-gathering mechanisms in wireless sensor networks. *IEEE Transactions on Vehicular Technology, 62* (4), 1472–1483. https://doi.org/10.1109/TVT.2012.2229309
Mennell, W. K. (2009). *Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, sequence-dependent team orienteering problem* (Doctoral dissertation, University of Maryland, College Park).
Neri, F., & Cotta, C. (2012). Memetic algorithms and memetic computing optimization: A literature review. *Swarm and Evolutionary Computation*, *2*, 1–14. https://doi.org/10.1016/j.swevo.2011.11.003
Ort, M. (2018). Autonomous navigation in rural environments without detailed prior maps, In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (pp. 2040–2047). IEEE.
Reinelt, G. (1991). TSPLIB—A traveling salesman problem library. *ORSA Journal on Computing*, *3*(4), 376–384. https://doi.org/10.1287/ijoc.3.4.376
Tang, J., Hiot Lim, M., Soon Ong, Y., & Joo Er, M. (2006). *Parallel memetic algorithm with selective local search for large scale quadratic assignment problems. International Journal of Innovative Computing, Information and Control 2*(6).
Thibbotuwawa, A., Bocewicz, G., Nielsen, P., & Banaszak, Z. (2020). Unmanned aerial vehicle routing problems: A literature review. *Applied Sciences*, *10*(13), Article 4504. https://doi.org/10.3390/app10134504
Visser, O., Sippel, S. R., & Thiemann, L. (2021). Imprecision farming? Examining the (in)accuracy and risks of digital agriculture. *Journal of Rural Studies*, *86*, 623–632. https://doi.org/10.1016/j.jrurstud.2021.07.024
Wang, X., Golden, B., & Wasil, E. (2019). A Steiner Zone Variable Neighborhood Search Heuristic for the Close-Enough Traveling Salesman Problem. *Computers & Operations Research*, *101*, 200–219. https://doi.org/10.1016/j.cor.2018.07.023
Yang, Z., Xiao, M.-Q., Ge, Y.-W., Feng, D.-L., Zhang, L., Song, H.-F., & Tang, X.-L. (2018). A double-loop hybrid algorithm for the traveling salesman problem with arbitrary neighbourhoods. *European Journal of Operational Research*, *265*(1), 65–80. https://doi.org/10.1016/j.ejor.2017.07.024