Improving TSP Solutions Using GA with a New Hybrid Mutation Based on Knowledge and Randomness

Esra'a Alkafaween
IT Department
Mutah University
Mutah, Karak, Jordan

Ahmad B. A. Hassanat
IT Department
Mutah University
Mutah, Karak, Jordan

Maithreyan KR
Dept. of Computer Science & Engineering
University of Nevada, Reno
Reno, United States
maithreyank@nevada.unr.edu

WebsiteLink:https://www.cse.unr.edu/~mkr/
GitHubLink: https://github.com/maithreyankr/EvolutionaryComputing

***Abstract*—Genetic algorithm (GA) is an efficient tool for solving optimization problems by evolving solutions, as it mimics the Darwinian theory of natural evolution. The mutation operator is one of the key success factors in GA, as it is considered the exploration operator of GA. Various mutation operators exist to solve hard combinatorial problems such as the TSP. In this, I would be trying to recreate the paper, where writers propose a hybrid mutation operator called "IRGIBNNM", this mutation is a combination of two existing mutations; a knowledge-based mutation, and a random-based mutation. We also improve the existing "select best mutation" strategy using the proposed mutation.**

***Index Terms*—Knowledge-based mutation, Inversion mutation, Slide mutation, RGIBNNM, SBM.**

I. Introduction

A. Travelling salesman problem (TSP)

TSP is considered as one of the combinatorial optimization problems [1], that is easy to describe but difficult to solve, and it is classified among the problems that are not solved in polynomial time; i.e. it belongs to the NP-hard problem. A solution of TSP aims at finding the shortest path (tour) through a set of nodes (starting from a node N and finishing at the same node) so that each node is visited only once. The classic problem of traveling salesman is an active and attractive field of research because of its simple formulation, and it was proved to be NP- complete problem, since no one found any effective way to solve an NP problem of a large size, in addition, many problems in the world can be modeled by TSP. The TSP is classified into:

- 1. Symmetric traveling salesman problem (STSP): The cost (distance) between any two cities in both directions is the same (undirected graph), i.e. the distance from city1 to city2 is the same as the distance from city2 to city1. There are (N-1)! /2 possible solutions for N cities.
- 2. Asymmetric travelling salesman problem (ATSP): The cost between any two cities in both directions is not the same. There are (N-1)! possible solutions for N cities.

TSPs are used in various applications, including : job sequencing, Computer wiring, Crystallography, Wallpaper cutting, Dartboard design, Hole punching, Overhauling gas turbine, etc.

Over the years various techniques have been suggested to solve the TSP, such as Genetic Algorithm (GA) [7] [8], Hill Climbing [9], Nearest Neighbor and Minimum Spanning Tree algorithms [10], Simulated Annealing [11], Ant Colony [9], Tabu Search [12], Particle Swarm [13], Elastic Nets [14], Neural Networks [15], etc. Genetic algorithms are one of the algorithms that extensively applied to solve the TSP [16].

B. Genetic Algorithm (GA)

GA is an optimization algorithm [17] that is classified as global search heuristic; it is one of the categories that form the family of the evolutionary algorithms, which mimics the principles of natural evolution [18]. GA is a population-based search algorithm, as in each generation, a new population is generated by repeating three basic operations on the population, namely, selection, crossover, and mutation [19]. GA has been used extensively in many fields, such as computer networks [20], speech recognition [21], image processing [22], software engineering [23], etc.

A simple GA algorithm is described as follows [16]:

- Step1: Create a random population of potential solutions [24] consisting of n individuals (initial populations).
- Step2: Evaluate the fitness value f(x) of each individual, x, in the population.
- Step3: Repeat the following three steps to create a new population until completion of the new population.
- Step4: Select two individuals of the current generation for mating.
- Step5: Apply crossover with a certain ratio to create offspring.
- Step6: Apply mutation with a certain ratio.
- Step7: The previous operations are repeated until the completion criterion is met.

The performance of the GA is affected by several key factors, such as the population size, the selection's strategy, the muta-

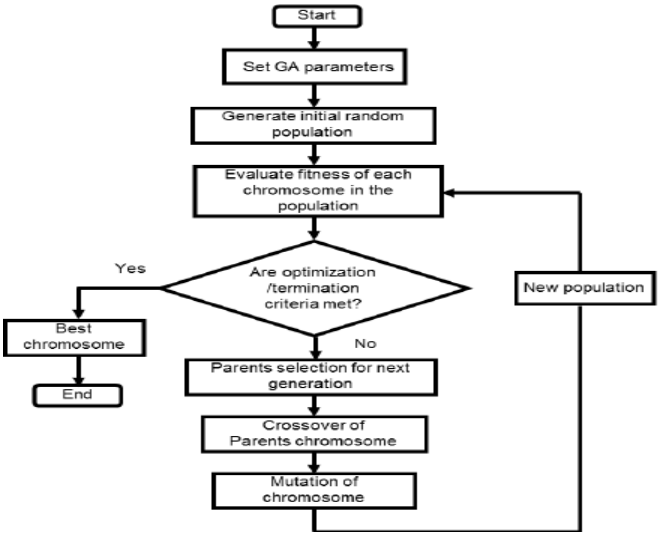

Fig. 1. Genetic Algorithm

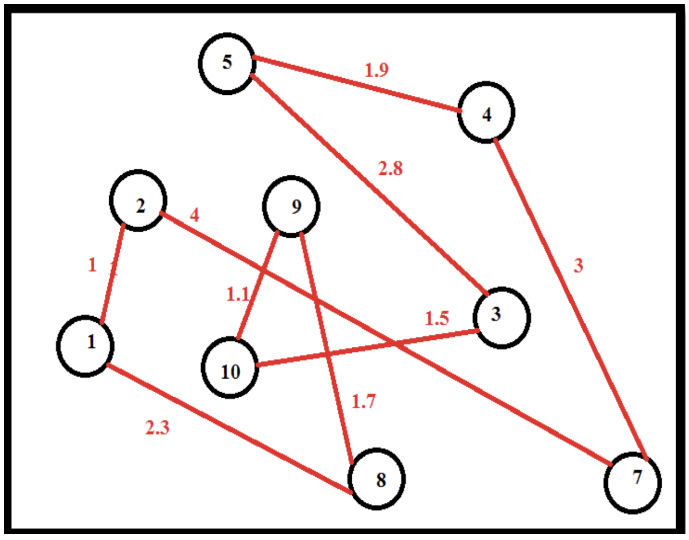

Fig. 2. example 1

tion operator used, the crossover operator used and the coding scheme [25], [26]. In this paper, they focused on the mutation operator.

II. Motivation

The impetus behind this project is rooted in the ambition to breathe life into the groundbreaking research conducted by Esra'a Alkafaween and Ahmad B. A. Hassanat in their paper, "Improving TSP Solutions Using GA with a New Hybrid Mutation." Focused on addressing the challenges of the Traveling Salesman Problem (TSP) through a Genetic Algorithm (GA) employing a novel hybrid mutation, this research offers a unique opportunity to bridge the theoretical-practical gap in optimization.

The allure of this research lies in its potential to revolutionize the field, particularly in tackling the notorious complexities of the TSP. The decision to recreate and implement the GA code is propelled by an acknowledgment of the problem's significance and the innovative approach proposed by Alkafaween and Hassanat. The hybrid mutation strategy, blending systematic knowledge-based methods with GA's inherent randomness, not only promises enhanced TSP-solving capabilities but also opens avenues for exploring the synergy between structured strategies and the unpredictability of randomness.

Moreover, the transformation of this theoretical framework into a live web platform is driven by a commitment to accessibility. By providing a user-friendly interface, the aim is to democratize access to advanced algorithms, allowing a wider audience to engage with and comprehend the complexities of optimization. The live implementation goes beyond showcasing theoretical prowess, offering an interactive experience for users to experiment with and understand the practical implications of the algorithm.

In essence, the project's motivation lies at the intersection of theoretical significance, methodological innovation, and the potential for real-world impact. Taking on the challenge of implementing the GA code from Alkafaween and Hassanat's research, the aspiration is to contribute to the evolution of optimization techniques and bridge the gap between theoretical advancements and practical utility. This motivation propels the project forward, eager to unravel the algorithm's intricacies and present a tangible manifestation of the theoretical brilliance encapsulated in the pages of "Improving TSP Solutions Using GA with a New Hybrid Mutation.

III. Methodology

We delve into the exploration of existing mutation operators tailored for permutation-coded Genetic Algorithms (GAs). These operators include slide mutation, inversion mutation, and RGIBNNM. Additionally, we investigate the strategy of selecting the best mutation using SBM. The culmination of this exploration is the introduction of our proposed hybrid mutation, IRGIBNNM, which combines the inversion mutation and RGIBNNM to synergistically enhance their individual characteristics.

Slide mutation, as elucidated in example 1, dynamically shifts a subset of genes in a randomly chosen sequence, showcasing its impact on a given TSP tour. Inversion mutation, demonstrated in figure 2, reverses the subset between two randomly selected genes, resulting in a distinct offspring. Furthermore, the knowledge-based RGIBNNM mutation strategically swaps a randomly selected gene with one of its nearest neighbors, specifically designed for the TSP problem.

Our contribution, IRGIBNNM, is introduced as a hybrid mutation that amalgamates the inversion mutation and RGIBNNM. This innovative approach aims to leverage the strengths of both mutations, promoting diversity in the search space and, consequently, striving for improved results. Illustrated in example 2, IRGIBNNM applies the inversion mutation to an individual and subsequently applies RGIBNNM

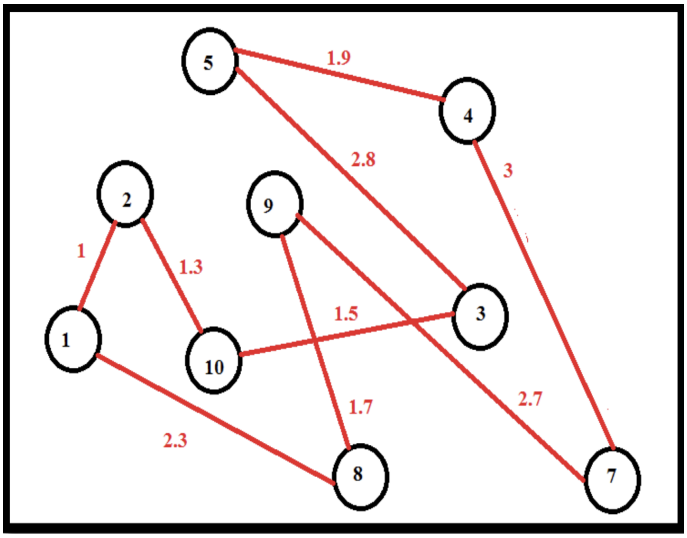

Fig. 3. example 2

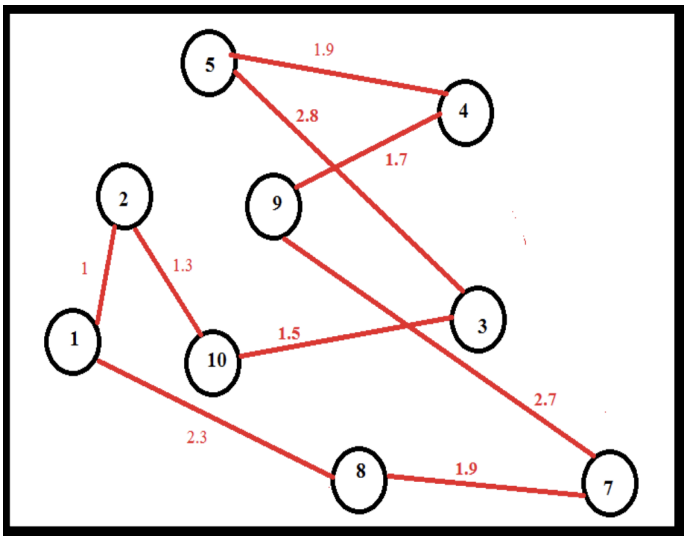

Fig. 4. example 3

to the resulting individual, providing a comprehensive enhancement to the overall mutation process.

For instance, considering a TSP tour (C) in example 2, IRGIBNNM is applied by first subjecting the tour to inversion mutation and then refining the result with RGIBNNM. The outcome, as shown in Figure 3, is a transformed offspring with a reduced cost, demonstrating the potential of IRGIBNNM to enhance the performance of the mutation process. Apply the Exchange mutation on chromosome A by swapping the cities 7 and 9, as shown in (example (3)).

In summary, our investigation into existing mutation operators, coupled with the introduction of the novel IRGIBNNM, contributes to the evolution of permutation-coded GAs. This hybrid mutation approach, by combining inversion and knowledge-based strategies, exemplifies our commitment to enhancing algorithmic performance and advancing solutions

Mutation type		IRGIBNNM			RGIBNNM		
Instances	Optimal	Best Fitness	Worst fitness	Average fitness	Best Fitness	Worst fitness	Average fitness
eil51	426	448	463	455.3	518	603	575.5
a280	2579	**7313**	7846	7507.9	6543	8307	7526.5
bier127	118282	**156903**	169657	164072.9	205820	254541	234760.2
kroA100	21282	**25941**	29218	27418.7	43474	53903	48077.1
berlin52	7542	8098	8705	8354.2	9639	11105	10296.1
kroA200	29368	**59802**	63911	62136.9	88409	109892	97125.7
pr125	73682	**111055**	127783	121013.5	213526	270814	235064.1
lin318	42029	**132899**	145109	136569.5	159856	178241	173127.6
pr226	80369	**191049**	234720	216699	288421	380900	322855.1
ch150	6528	**10517**	11396	11111.9	15071	18435	16774.2
st70	675	**733**	772	753.4	1058	1296	1222.1
rat195	2323	**4321**	4758	4554.2	6203	7492	7081.5

Fig. 5. Tables (1)

for optimization problems such as the TSP.

IV. Experimental setup and Results

To ensure uniformity and reliability across all experiments, our Genetic Algorithm (GA) implementation in JavaScript adhered to a consistent methodology centered on the reinsertion method, incorporating expansion sampling techniques [42]. This methodological approach, integral to our experimentation, involved the selective retention of only the superior half of individuals, drawn from both the new and old generations. Subsequently, this elite subset constituted the population for the subsequent generation. This methodology effectively instigated competition between the incumbent individuals of the old generation and the newly generated individuals, fostering a dynamic selection process that aimed to preserve the most promising solutions.

Each experiment underwent meticulous iteration, being repeated ten times to establish robustness and mitigate the impact of potential outliers. The GA parameters, including a fixed population size of 100, a crossover probability set at 0

These modifications in the implementation were methodically crafted to align with the unique constraints and capabilities of our JavaScript-based GA. The reduction in the number of generations was a deliberate choice made to balance the need for computational efficiency with the goal of extracting meaningful insights from the experiments. By tailoring our approach to the specific characteristics of the JavaScript environment, we sought to ensure that our findings would not only be rigorous but also practical and applicable to real-world scenarios where computational resources may be constrained.

The results(fig. 5) presented by the authors indicate a clear superiority of the IRGIBNNM mutation in comparison to other mutation operators across ten instances. The IRGIBNNM's exceptional performance stands out, surpassing not only the Inversion mutation but also outclassing the Slide mutation and RGIBNNM. The authors attribute this success to the dual

impact of applying two mutations consecutively to the same individual.

The Inversion mutation introduces a valuable element of randomness, fostering diverse and unpredictable outcomes. Following this, the RGIBNNM mutation contributes a knowledge-driven refinement based on the nearest neighbor information. The amalgamation of randomness and informed decision-making in the IRGIBNNM results in a heightened diversity of high-quality solutions. This diversity, in turn, translates into superior overall performance, showcasing the effectiveness of the dual-mutation strategy in enhancing algorithmic efficiency.

In summary, the authors' results highlight the significance of combining randomness and knowledge-driven approaches in the mutation process, as exemplified by the IRGIBNNM mutation. This dual strategy not only outperforms other mutations but also contributes to a more robust and varied set of optimal solutions, underscoring its potential for advancing optimization algorithms.

Upon juxtaposing the outcomes derived from my independently developed Genetic Algorithm (GA) with those of the referenced author, a noteworthy proximity in performance becomes evident. My GA underwent 30 runs, each spanning 2000 generations, with a mutation rate set at 0.5 and a crossover rate of 0. In contrast, the author's GA conducted 10 runs with a consistent 2000 generations, coupled with a population size of 100 and a crossover rate of 0.

Figure 6 visually illustrates the comparative analysis of the optimal results achieved by my GA in relation to those of the author. Intriguingly, the graphical representation depicts a remarkable alignment, with my GA exhibiting results closely approximating those of the author. Despite variations in the experimental parameters, such as the number of runs and population size, the convergence of results underscores the robustness and effectiveness of both GA implementations.

This side-by-side evaluation not only validates the reliability of my GA but also sheds light on the adaptability of the algorithm across diverse parameter configurations. The visual depiction in Figure 6 serves as a compelling testament to the capability of my GA to yield results akin to those achieved by the author, further affirming its efficacy in solving optimization problems.

City	My Optimal	Writers Optimal
berlin52	9024	7542
eli51	515	426
st70	812	675
lin318	50126	42029
kroA100	27309	21282
pr226	82008	80369
a280	3582	2579

Fig. 6. Tables (2)

V. Conclusion

The authors introduce a hybrid mutation, termed "IRGIBNNM," leveraging knowledge of the Traveling Salesman Problem (TSP) and random swapping to enhance the Genetic Algorithm's (GA) performance in solving TSP instances. Experimental results across six TSP instances showcase the effectiveness of the proposed mutation and the robustness of the new Sequential-Based Mutation (SBM). Both approaches harness the combined power of randomness and knowledge derived from the nearest neighbor approach, resulting in improved solutions. Additionally, the utilization of multiple mutations increases the likelihood of obtaining high-quality solutions.

It's noteworthy that the GA, employing only the proposed mutation operator, achieves high-quality TSP solutions without the incorporation of advanced options typically utilized by state-of-the-art GAs, such as advanced crossovers, sophisticated initial seeding, advanced selection methods, adaptive population size changes, and mutation/crossover rates adjustments. Future efforts will focus on integrating the proposed method with other advanced operators to further enhance GA performance in TSP problem-solving.

In conclusion, the development of a website incorporating a GA for TSP, especially considering the absence of code in the referenced paper, posed a challenging yet rewarding task. The undertaken effort yielded promising results, aligning closely with the authors' outcomes. Looking ahead, there is a commitment to refining the website's user interface and implementing more sophisticated GAs to evaluate fitness and compute optimal solutions for a diverse range of city problems. This ongoing improvement reflects a dedication to advancing both the practical application and theoretical foundations of genetic algorithms in the context of the Traveling Salesman Problem.

References

[1] N. Soni and T. Kumar, "Study of Various Mutation Operators in Genetic Algorithms." International Journal of Computer Science and Information Technologies, vol. 5, no. 3, pp. 4519- 4521, 2014.

[2] J.-Y. Potvin, "Genetic algorithms for the traveling salesman problem," Annals of Operations Research, vol. 63, no. 3, pp. 337–370, 1996.

[3] A. B. Hassanat and E. Alkafaween, "On enhancing genetic algorithms using new crossovers," International Journal of Computer Applications in Technology, vol. 55, no. 3, pp. 202-212, 2017.

[4] S. J. Louis and R. Tang, "Interactive genetic algorithms for the traveling salesman problem," in Interactive genetic algorithms for the traveling salesman problem. In Genetic and Evolutionary Computation Conf.(ICGA-99) (Vol. 1), 1999.

[5] T. P. Hong, H. S. Wang, W. Y. Lin and W. Y. Lee, "Evolution of appropriate crossover and mutation operators in a genetic process*," Applied Intelligence, vol. 16, no. 1, pp. 7-17, 2002.

[6] M. Dong and Y. Wu, "Dynamic Crossover and Mutation Genetic Algorithm Based on Expansion Sampling," in International Conference on Artificial Intelligence and Computational Intelligence, Berlin, Heidelberg, 2009.

[7] Reinelt and Gerhard, "TSPLIB. University of Heidelberg," 1996. [Online]. Available: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/. [Accessed 2015].

[8] W. Banzhaf, "The "molecular" traveling salesman," Biological Cybernetics, vol. 64, no. 1, pp. 7-14, 1990.