www.editada.org

International Journal of Combinatorial Optimization Problems and Informatics, 16(4), Sep-Dec 2025, 486-496. ISSN: 2007-1558. https://doi.org/10.61467/2007.1558.2025.v16i4.563

A metaheuristic proposal using Java for the quadratic assignment problem

Rogelio González Velázquez[1], Erika Granillo Martínez[2], María Beatriz Bernabé-Loranca[1], Abraham Sánchez-López[1]

[1] Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, Av. San Claudio, C.P. 72590 Puebla, México

[2] Facultad de Administración, Benemérita Universidad Autónoma de Puebla, Av. San Claudio, C.P. 72590 Puebla, México

rogelio.gonzalez@correo.buap.mx,erika.granillom@correo.buap.mx,beatriz.bernabe@gmail.com, abraham.sanchez@correo.buap.mx

Abstract. The classic combinatorial optimization problems belonging to the NP-hard class is the quadratic assignment problem. The interest in solving the problem lies in its high computational complexity, as well as its applications in logistics, gate assignment in airports, among others. In this work, the Greedy Random Adaptive Search Procedure metaheuristic was implemented to find its solutions. The main contribution of this work is the adaptation of a neighborhood structure contained in k-exchanges in the post-processing phase. The tests were performed for 29 large-scale instances whose dimensions range from 64 to 254 taken from the QAPLIB. The approximate solutions were found through a metaheuristic that bases its search on neighborhoods and local search algorithms. Java was the programming language used for the implementation of metaheuristics; its execution allowed balancing the parameters to obtain competitive results with respect to the values known in literature. The results reported achieved the proposed objectives.

Keywords: Metaheuristic, NP-hard, 2-exchange, neighborhood, combinatorial optimization.

Article Info
Received November 29, 2024
Accepted December 4, 2024

1 Introduction

Within the literature there are algorithms proposed to solve various problems, whether for industrial sectors within their daily operations or for science and research, whatever the application of the algorithms, they allow the efficiency of operations. in different areas. Algorithms, also called heuristic or metaheuristic methods, are basically elements that consist of finding or discovering an appropriate algorithm to solve optimization problems. Currently the word optimization is constantly used to refer to the fact of finding an improvement in a process, on the other hand, in contexts such as science and research go beyond the simple fact of improving, since it focuses on finding the best solution for a specific problem.

Optimization problems can present diverse solutions, but a criterion must be established to eliminate possibilities and find the best possible solution. In this way, optimization problems are formulated to find the value of decision variables subject to restrictions that determine their value, in addition to an objective function to determine a maximum or minimum value within the operation. The most general case of the optimization problem can be written as:

$$Optimize \quad z = f(x) \tag{1}$$

$$subjetc\ to\ \ x\ \in \Omega.$$

Where z is the objective function in equation 1, Ω is the set of feasible solutions and f is a real function called the objective function that associates each feasible solution $x\ \in \Omega$ with a cost $f(x)$.

The word optimize can be changed to minimize or maximize. In the case of the minimization problem, a global minimum is sought, that is, a solution $x^* \in \Omega$ such that $f(x^*) \leq f(x), \forall x \in \Omega$ and in the case of the maximization problem, searches for a global maximum, that is, a solution $x^* \in \Omega$ such that $f(x^*) \geq f(x), \forall x \in \Omega$.

Optimization problems are commonly classified into two classes, one when the mathematical model has continuous variables that take real values, and its feasibility space is infinite and the other has discrete variables that take integer values, and its feasibility space can be of finite cardinality.

It is mentioned that it is a combinatorial optimization problem (COP), if the decision variables only admit integer values and its solution space is formed by permutations or subsets of natural numbers. Therefore, a COP is a discrete optimization problem. Various combinatorial optimization problems can be stated as integer programming problems and solved using exact integer programming methods such as Branch and Bound. However, such formulations sometimes involve endless variables and constraints, and although several of these methods have demonstrated their theoretical convergence, they cannot cope with very large problems, which are generally those that arise in real-life practice. One of the main problems with integer programming is that the feasibility region is not convex, so there is no guarantee that an integer problem can be solved in a reasonable time for a computer.

Within the combinatorial optimization problems there is a class of problems of high computational complexity known as NP-hard (Sahni and González, 1976) for which there is no known efficient algorithm that can solve it in polynomial time and given the inefficiency of the methods. To solve this type of problem, approximation methods called Metaheuristics (MH) are used. In general, MHs are general-purpose algorithms that guide a heuristic in the search for solutions in strategic regions of the feasibility space. The search strategies most used in MH are by neighborhoods or by populations with an approach inspired by natural processes.

This work focuses on a solution search method using neighborhoods for which it will be said that the neighborhood of a solution $x \in \Omega$ is a subset $N(x) = \{x_1, x_2, \cdots x_p\} \subseteq \Omega$, each solution can reach to a $N(x)$ from x using an operator called move. This work focuses on the search for solutions to the Quadratic Assignment Problem (QAP), through the well-known Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic procedures (Resende and Ribeiro, 2016). QAP consists of finding an optimal allocation between n facilities and n locations considering the distance between the locations and the flow of materials between the facilities so that the cost of material transportation is minimized. The QAP is a classic combinatorial optimization problem and is classified as NP-hard.

The main objective of this work is the adaptation of neighborhood structures in the post-processing phase of GRASP to compare the best-known value (VMC) against the result of the best-found value (VME). With the results that are intended to be found, it will be possible to visualize and verify if the implemented procedure is considered a robust metaheuristic to solve this type of problems. Furthermore, the efficiency of GRASP was analyzed by running two sets of parameters, the first with $\{\alpha = 0.2, \beta = 0.2\}$ and the second with $\{\alpha = 0.5, \beta = 0.1\}$.

The tests were carried out on 29 instances, 2 by B. Eschermann and H.J. Wunderlich, 8 by Y. Li and P.M. Pardalos, 10 for Skorin Kapov, 6 for E.D. Taillard, one for U.W. Thonemann and A. Bölte and 2 for M.R. Wilhelm and T.L. Ward, taken from the QAPLIB library (Burkard et al, 2024). The proposed objective is to find approximate solutions to those reported in the literature using metaheuristic procedures as a class of approximation methods designed to solve difficult combinatorial optimization problems (Díaz et al, 1996).

Optimization problems are based on choosing the best configuration among a set of feasible solutions to achieve an objective (Cela, 1995) and are divided into two categories: problems with continuous and discrete variables. For this case, the second category will be taken, which attempts to find an objective that is selected within a finite and discrete set, an integer, a set of integers, a permutation or a graph. The two types of problems have different solution methods; However, combinatorial optimization problems belong to the second category (Cela, 1995) as an example, the Greedy Random Adaptive Search Procedure (GRASP).

GRASP is an iterative procedure where each step consists of a construction phase and an improvement phase. In the construction phase, a constructive heuristic procedure is applied to obtain a good initial solution. This solution is improved in the second phase using a local search algorithm. The best of all the solutions examined is the result of (Li et al, 1994) and (Resende, Pardalos and Li, 1996). The improvement phase could be a simplified local search or apply another metaheuristic to form hybrid algorithms.

Other metaheuristics can be used as mentioned above such as: Variable Neighborhood Search (VNS), Taboo Search (TS), genetic algorithms (GA) among others (Resende and Ribeiro, 2016).

The main contribution of this work is based on experimentation and analysis of the results of the method, obtaining a good initial solution and improving it later.
This article is organized as follows: Section 1 addresses an introduction to the work; section 2 presents related work. In section 3, the formulation of the quadratic assignment problem is described, in section 4, the metaheuristic is shown, in section 4.1 GRASP is defined, in section 4.2 the implementation of GRASP to QAP is located. Section 5 contains the results and discussion; section 6 contains conclusions and future work. Finally, in section 7 references.

2 Related work

The Quadratic Allocation Problem (QAP) is a combinatorial optimization problem that consists of finding an optimal allocation of n resources to n locations to minimize the transportation cost. Additionally, two matrices are needed, one for the requirements of the units to be transported and the Second is the cost of transportation per unit between locations.

The QAP was proposed by (Koopmans and Beckmann, 1957). Subsequently, two decades later, it was shown that the QAP is NP-hard (Sahni and González, 1976). So far, optimal solutions have been found using exact methods for instances of size 30 (Roucairol, 1987). QAP is used in multiple applications such as: computer keyboard design, production scheduling, airport terminal design and communication processes. An exact branching and bounding algorithm with some variants has been used to solve the QAP that was presented by (Roucairol, 1987), however, recently solutions were proposed with different metaheuristic techniques such as in (Zhou, Hao and Duval, 2020), (Hafiz and Abdennour, 2016): genetic algorithms, simulated annealing, tabu search (Skarin-kapov, 1990) and GRASP (Li et al, 1994), (Chmiel et at, 2017). In (Pardalos et al, 1995) they implemented GRASP in parallel to solve the QAP.

Other works related to the search for QAP solutions are based on the particle swarm algorithm, operator recombination for genetic algorithms and stall-aware cooperative parallel to local search (Aksan, Dokeroglu and Cosar, 2017), (Tosun, 2014), (Obdelkafi et al, 2016). QAP in large instances is a notoriously difficult problem to find a solution and the performance of metaheuristic algorithms varies, as mentioned in (Saifullah Hussin and Stützle, 2014) where two algorithms are compared with results that depend on the size of the problem.

In (Kumar,Sahu and Mitra, 2021) simulated annealing algorithms, genetic algorithms, iterated local search, tabu search are implemented and compared to solve the QAP for large instances. Another example of parallel processing can be seen in (Dokeroglu, Sevenic and Cosar, 2019) where the metaheuristic is implemented with tabu search to work with large QAP instances. Recently, some algorithms such as NSGA-II combined in parallel with Greedy have been used to provide a solution to the multi-objective QAP problem. On the other hand, in recent years algorithms inspired by nature have been applied, such as the Intelligent optimization algorithm (IOA), which has been used by different researchers to provide a quality solution to QAP applications (Dokeroglu, Sevenic and Cosar, 2019).

Within the research activities and especially in the challenges to solve optimization problems as well as the design of hybrid metaheuristics to apply them to the QAP, there is the challenge to establish an appropriate diversification of procedure to facilitate efficiency in the search for solutions. to said problem. For this reason, intensive work has been done on a parallel memetic iterated tabu search (PMITS) metaheuristic to solve the QAP and its variants. Furthermore, it has been seen that extending the non-parallel PMITS iterations improves sequential performance by adding a modified uniform random version of a crossover operator to a parallel environment (Silva et al., 2021) to solve the most difficult instances of the QAPLIB (Burkard et al., 2024).

On the other hand, the development of hybrid methods applied to facility layout problems(FLP) using discrete differential evolution algorithms(DDE) to solve the QAP, which aim to find the root where the problems tend to get worse to provide the best solution, once the problem is found, then an algorithm combining DDE algorithms and tabu search (TS) is proposed to facilitate the exploration of the mechanism in the DDE algorithm and provide optimal solutions (Hameed, Asaad Shakir et al., 2021)

3 Formulation for the Quadratic Assignment Problem

QAP consists of finding an optimal allocation that minimizes the cost of transporting materials, between n facilities in n locations, considering the distance between locations and the flow of materials between facilities. The QAP can be formulated using a combinatorial optimization model (Resende, Pardalos and Li, 1996).
Given a set of $N = \{1,2 \dots, n\}$ and two asymmetric matrices of size $n\ x\ n$ where: $F = (f_{ij})$ and $D = (d_{kl})$, a permutation of $p \in \prod_N$ that must be found to minimize.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} d_{p(i)p(j)} \tag{2}$$

Where $\prod_N$ is the set of all permutations of N and F is the material flow matrix between the facilities and D is the distance matrix between locations $i, j, k, l = 1, \cdots, n$. As shown in equation 2.
There are other mathematical formulations for this type of problem, for example, expressed as a binary integer programming problem like the following:

$$\min \sum_{i}\sum_{j}\sum_{k>i}\sum_{l\neq j} C_{i,j,k,l}\, x_{i,j} x_{k,l} \tag{3}$$

Subject to:

$$\sum_{i=1}^{n} x_{ij} \ = 1, (j = 1,2, \dots, n) \tag{4}$$

$$\sum_{j=1}^{n} x_{ij} \ = 1, (i = 1,2, \dots, n) \tag{5}$$

$$x_{ij} = 0,1 \ \ (i, j = 1,2, \dots, n) \tag{6}$$

The equation 3 is the objective function of binary problem, the next equation 4, 5, indicates that each node corresponds to one installation and each installation has only one location. Equation 6 indicates that the variables are binary.

The previous objective function is very difficult to display, so it is easier to use the following equivalence:

$$tr(FXDX^t) = \sum_{i}\sum_{j}\sum_{k>i}\sum_{l\neq j} C_{i,j,k,l}\, x_{i,j} x_{k,l} \tag{7}$$

The above means the trace of this product of matrices where X is the matrix of binary variables. This approach is easier to develop with the solver add-in in Excel.

4 Metaheuristic

The main drawback that heuristic techniques face is the existence of local optima that are not absolute. If during the search there is a local optimum, the heuristic would not be able to continue the process and would be "stuck" at the same point. To solve the problem, it is recommended to restart the search from another initial solution and verify that the new search explores other paths (Resende and Ribeiro, 2016).

Most combinatorial optimization problems are specific problems, so a heuristic technique algorithm that works for one problem is sometimes not useful for solving other problems. However, in recent times, general-purpose heuristics called metaheuristics have been developed that attempt to solve the above drawbacks. Most metaheuristics are developed with neighborhood search methods (Resende and Ribeiro, 2016).

The word metaheuristic was coined (Díaz et al, 1996) and the term Taboo Search emerged (1986). A metaheuristic is a master strategy that guides and modifies other heuristics to generate better solutions than those typically presented by other methods (Mishmast and Gelareh, 2007).

There are different successful metaheuristics in solving combinatorial problems. The Greedy Randomized Adaptive Search Problem (GRASP) metaheuristic is one of the most recent techniques. It was originally developed (Feo and Resendes, 1995) at the time of studying coverage problems of high combinatorial complexity (Li et al, 1994). Each iteration in GRASP generally consists of two steps: the construction phase and the local search procedure. In the first stage, an initial solution is built, which is subsequently improved through post-processing to perfect the solution obtained in the first stage until a local optimum is obtained. There are works where this metaheuristic is applied to optimization problems in big data (Palmieri et al, 2016). In addition, there are other variants such as: GRASP-path relinking, GRASP-reactive, GRASP-parallel, GRASP-hybrids, with some other metaheuristics whose search is based on neighborhoods (Festas and Resendes, 2011).

4.1 Greedy Randomized Adaptive Search Problem (GRASP)

A GRASP is an iterative process, each iteration consists of two phases: the construction phase and the local search procedure. In the first, an initial feasible solution is constructed, in the second, called post-processing, an initial permutation obtained from the construction is implemented through a local search applied, subsequently it is improved through an exchange procedure until a local optimum is obtained. (Koopmans and Beckmann, 1957). Once the two phases have been executed, the solution obtained is stored and another iteration is carried out, each time saving the best solution found at that moment.
An algorithm that exemplifies metaheuristics is shown in Figure 1.

```
Procedure GRASP
   InstanceEntry ();
   While(criterion not satisfied) do
            Phase       1       Solution
      Construction ();
          Phase 2 post-processing ();
          Update Solution ();
   End
   Go back (Best solution)
End
```

Fig. 1. GRASP Generic Pseudocode. Source (Li et al, 1994).

The overview of the main components of GRASP are: The Greedy component that uses a myopic algorithm for the selection of components that guide the construction of solutions, the Random used for random selections from a list of elite candidates that determine the path of the search, the Adaptive has the mission of updating each result obtained from the components of the solution that is built (El Mouayani et al., 2019).

4.2 GRASP for the Quadratic Assignment Problem (QAP)

Some researchers have used the GRASP design to solve the QAP in different cases (Díaz et al, 1996), (Resende, Pardalos and Li, 1996), (Sahni and González, 1976). It should be noted that the solutions are a permutation of length n, summarizing the procedure as follows: Initial construction phase: stage 1, generation of a list of candidates, which has previously been restricted by two parameters $\alpha \; and \; \beta$, then, randomly one is taken from these candidates from which the first two assignments are derived. Stage 2, the remaining $n-2$ mappings are added relative to the *Greedy* procedure, once the process is complete, the permutation is completed, producing a feasible solution. Improvement phase 2: the solution generated from phase 1 is taken as the initial solution of some local search procedure, at the end of the procedure a local optimal solution will be obtained, which could also be the global optimal one.

The neighborhood structure used is: 2-exchange, which consists of an initial solution of S_0, generating a neighborhood with cardinality C_k^n. The k-exchange neighborhood structure has been used in permutation search problems such as the traveling agent

problem, the graph partitioning problem, among others. The 2-exchange neighborhood structure has achieved good results, if $k \geq 3$ the search could have a high computational cost (Resende and Ribeiro, 2016). An example of the neighborhood of $n = 5$ is shown in table 1:

$$C_2^5 = \frac{5!}{(5-2)!\,2!} = \frac{5*4}{2} = 10 \quad (8)$$

Table 1. Neighborhood 2- exchange of initial solution S_0 for Nugent 5 matrix

Neighborhood 2-exchange related to S_0						
	Permutation					**Obj. function**
S_0	5	1	4	2	3	45
S_1	1	5	4	2	3	41
S_2	4	1	5	2	3	38
S_3	2	1	4	5	3	35
S_4	3	1	4	2	5	39
S_5	5	4	1	2	3	39
S_6	5	2	4	1	3	38
S_7	5	3	4	2	1	33
S_8	5	1	2	4	3	35
S_9	5	1	3	2	4	40
S_{10}	5	1	4	3	2	45

In practice the flow matrices are taken as well as the distance matrix and their elements are listed separately. the flows are ordered from largest to smallest and the distances from smallest to largest, both lists are restricted with a parameter of $0 < \alpha < 1$, and are multiplied generating a new list of elements of the form $f_{ij}*d_{kl}$ that contain large flows and short distances.

This list is restricted with a parameter $0 < \beta < 1$, with these operations a restricted list of candidates (RLC) is obtained, from this list an element of the form $f_{ij}*d_{kl}$ is randomly selected, producing the first assignment pair $(i, k), (j, l)$, interpreted as facility k is assigned to location i and facility l is assigned to location j. Finally, there are $n - 2$ components left to assign, again with a greedy the C_{ik} costs are calculated with respect to the 2 assignments against the remaining possible assignments, another RLC is formed and one of these candidates is selected, thereby generating the third assignment and so on. successively until completing the permutation of n components called initial solution S_0 which is submitted to phase 2, called post-processing phase, completing an iteration of GRASP (Riffi, Saju and Barkatou, 2017).

5 Results and discussion

This section shows the results produced by running the GRASP metaheuristic for 36 instances extracted from the QAPLIB library (Burkard et al, 2024) to show the efficiency of the proposed methodology since the instances vary in size ranging from $n = 20$ to $n = 256$, considered large-scale instances. To constrain the LRC, GRASP was first run with the parameters of $\alpha = 0.2$ and $\beta = 0.2$, then processed a second time with the parameters of $\alpha = 0.5$ and $\beta = 0.1$ (Li et al, 1994). The purpose of the experiment is to observe the efficiency of GRASP with variations in the RLC parameters.

Table 2 shows the largest instances of the experiment, later in Table 3 the instances of considerable size of the same case are presented. Below is the following table with the aforementioned values.

Table 2. The results presented are for a set of 29 instances of dimensions from 60 to 256

Instances	BVF	BKVF[1]	GAP[1]	CPUT[1]	BKVF[2]	GAP[2]	CPUT[2]
esc 64a	116	116	0.00	16194	116	0.00	17422
esc128	64	64	0.00	289358	64	0.00	278097
Lipa 60a	107218	108136	0.86	31230	108132	0.85	31688
Lipa 60 b	2520135	3007542	19.34	26586	3006630	19.30	39909
Lipa70 a	169755	170930	0.69	60449	170968	0.71	60414
Lipa 70 b	4603200	4614766	0.25	46703	4614766	0.25	50074
Lipa 80 a	253195	254930	0.69	116748	254928	0.68	96351
Lipa 80 b	7763962	9388572	20.93	77328	9398638	21.05	152594
Lipa 90 a	360630	360630	0.00	151700	360632	0.00	152587
Lipa 90 b	12490441	15154028	21.33	121205	15163102	21.40	129387
sko 64	48498	49014	1.06	60200	49066	1.17	69899
Sko72	66256	66728	0.71	116687	66744	0.74	114110
Sko 81	90998	91820	0.90	455312	91970	1.07	156591
Sko 90	115534	116654	0.97	316722	116646	0.96	246604
Sko 100 a	152002	153150	0.76	402552	153282	0.84	381774
Sko 100 b	153890	155344	0.94	377645	155032	0.74	379244
Sko 100 c	147862	155032	4.85	379244	149398	1.04	378249
Sko 100 d	149576	150990	0.95	369676	150272	0.47	414197
Sko 100 e	149150	150674	1.02	377337	150444	0.87	377573
Sko 100 f	149036	150430	0.94	365562	150444	0.94	410407
Tai 60 a	7205962	7438638	3.23	25455	7445042	3.32	26845
Tai 64c	1855928	1855919	0.00	18817	1855919	0.00	26845
Tai 80 a	13499184	13865020	2.71	18817	13915442	3.08	74053
Tai 80 b	818415043	818415043	0.00	152698	818415043	0.00	152594
Tai 150	498896643	498896643	0.00	144817	498912784	0.003	145599
Tai 256	44759294	44890864	0.29	377645	44906536	0.33	379244
Tho 150	8133398	8147865	0.18	377645	8137865	0.05	383645
Wil 100	273038	274312	0.47	466797	274312	0.47	389396
Wil 50	48816	49010	0.40	23901	49034	0.45	23351

Table 2 shows the results of the experiment, as well as the instances of Lipa from 60 to 90, Sko from 64 to 100, Tai from 60 to 256, Tho 150 and Wil from 50 to 100. The first column shows the test instances, the second, with the header BVF contains the best value found, then the 3 columns best known value found (BKVF1) and 4 GAP1 corresponds to GRASP with parameters of α=0.2 and β=0.2, the fifth column represents the CPUT1 computing time. Finally, columns 6 BKVF2 and 7 GAP2 are from GRASP with parameters of α=0.5 and β=0.1, the computation time is shown in the last column.

Table 3. The results presented are for Nugent instances of dimensions from 20 to 30

Instances	BVF	BKVF[1]	GAP[1]	CPUT[1]	BKVF[2]	GAP[2]	CPUT[2]
Nug 20	2570	2594	0.9	8.904	2570	0.0	0.67
Nug 22	3596	3596	0.0	7.833	3596	0.0	3.596
Nug 24	3488	3488	0.0	10.253	3496	0.2	3.488
Nug 25	3744	3746	0.1	7.759	3748	0.1	3.926
Nug 27	5234	5234	0.0	7.981	5236	0.0	5.236
Nug 28	5166	5182	0.3	7.013	5178	0.2	5.178
Nug 30	6124	6136	0.2	7.248	6156	0.5	6.156

Table 3 shows the following: the first column shows the test instances, the second, with the header BVF contains the best value found, then the 3 columns best known value found (BKVF1) and 4 GAP1 corresponds to GRASP with parameters of α=0.2 and β=0.2, the fifth column represents the CPUT1 computing time. Finally, columns 6 BKVF2 and 7 GAP2 are from GRASP with parameters of α=0.5 and β=0.1, the computation time is shown in the last column.

It can be seen in the results within table 2 in the GAP[1] column that for the sets executed with the parameters of $\alpha = 0.2, \beta = 0.2$ the percentage of error with respect to the BVF and the BKVF[1],2, of 9 instances was 0.0%, 12 cases ranging from 0.40% to 0.97%, the error being less than 1%. On the other hand, an error of 1% was obtained for 2 instances, for the Sko c and Tai 60 a, 80a matrices, the error ranges between 2% and 3% finally, the Lipa 60b, 80b, 90b instances, which have the percentages The highest error rates range from 19% to 21% due to its nature as it is a larger scale instance.

Regarding the parameters of $\alpha = 0.5, \beta = 0.1$ in the GAP[2] column, it was found that again 9 instances share the error percentage of 0.0%, likewise, 12 instances have an error less than 1%, there was an increase of two to three instances with a value of 1%, decreasing to two instances with error percentages of 3%. The instances that remained with the same error percentage of 19% to 21% were Lipa 60b, 80b, 90b.

For the 29 instances, it is observed that 72% of the test sets have a gap of $0 < \text{GAP} < 1$, the remaining 28% includes the rest of the instances. Therefore, and according to the data provided, it is shown that the implementation of GRASP is efficient and at the same time robust (see figure 2).

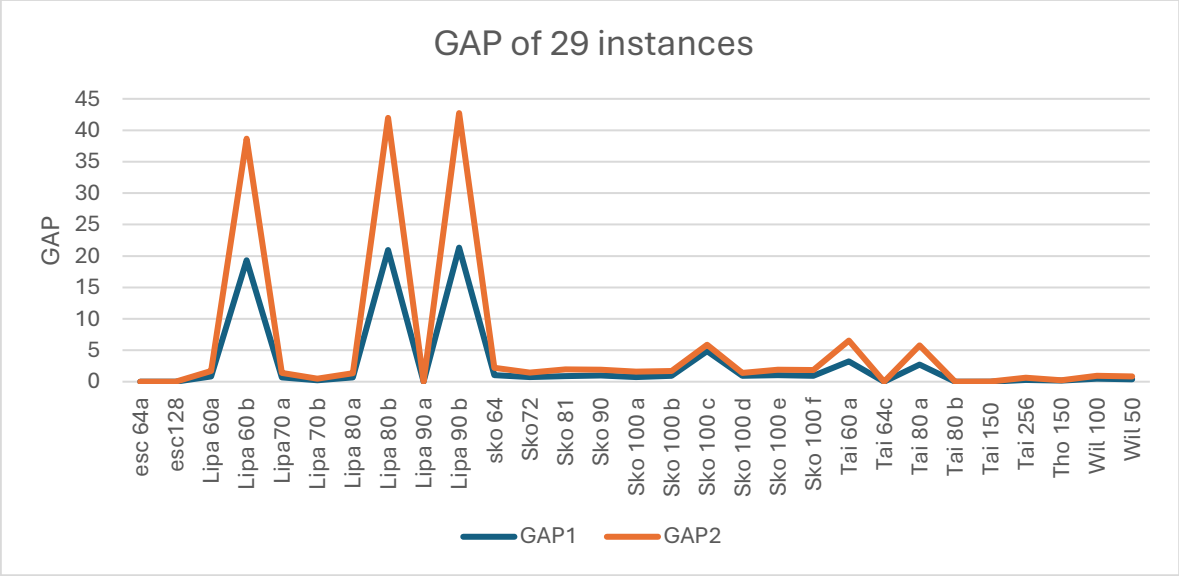

Fig. 2. Gap comparison between the 29 instances

The average computing time for each instance is compared, the results show the efficiency of the strategy for all cases as seen in figure 3.
It is worth mentioning that the program in use was designed in the JAVA programming language, so the experiment was processed on an Intel Evo Core i5 computer with an average computing time of 189832.7 milliseconds and with a difference in the maximum time in relation to the minimum. of 450603 milliseconds for the parameter $\alpha = 0.2, \beta = 0.2$. Finally, for the parameters $\alpha = 0.5, \beta = 0.1$, the average computing time was 190750 milliseconds with a difference between the maximum and minimum time of 396775 milliseconds.

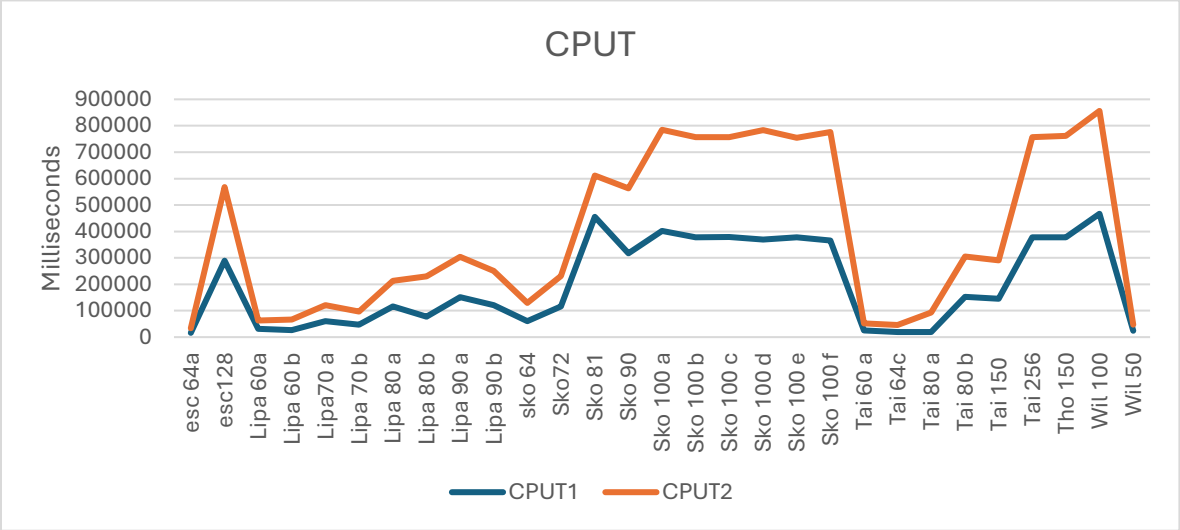

Fig. 3. CPUT comparison between the 29 instances

As indicated at the beginning of this section, another experiment was used with Nugent's matrices. The results obtained for these instances of considerable size are shown in Figures 4 and 5. Figure 4 shows the performance of the algorithm for different parameters of α and β. It is observed that matrices of moderate size ranging from $n = 20$ to $n = 30$ have lower variability in error, except in $n = 20$ and $n = 30$, so it is stated that the algorithm remains robust to the small changes shown.

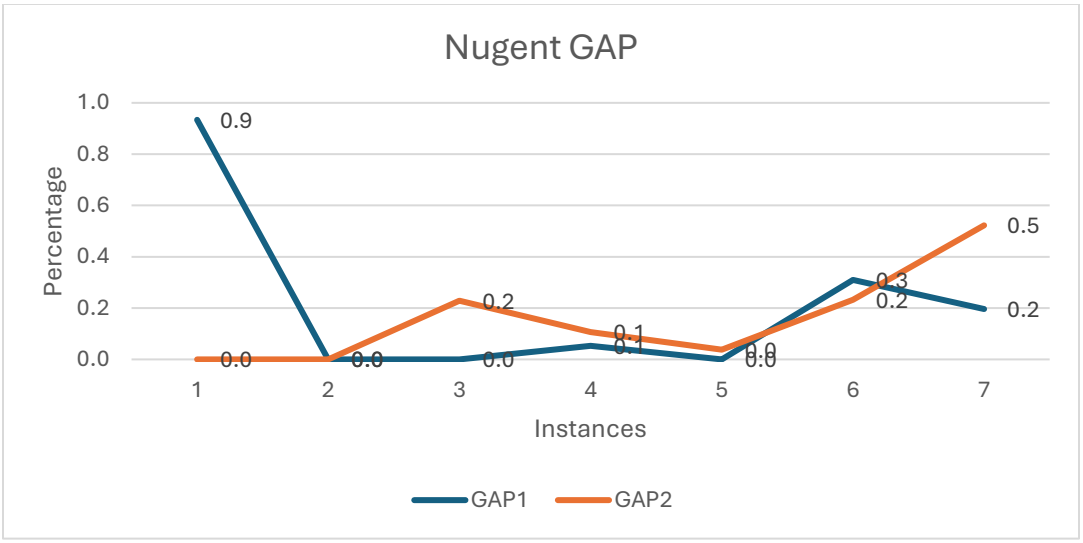

Fig. 4. Nugent Gap comparison

Figure 5, it is shown that the algorithm executed with the parameters α= 0.5 and β= 0.1 presents better performance in the CPUT time, reaching a balance in relation to the GAP.

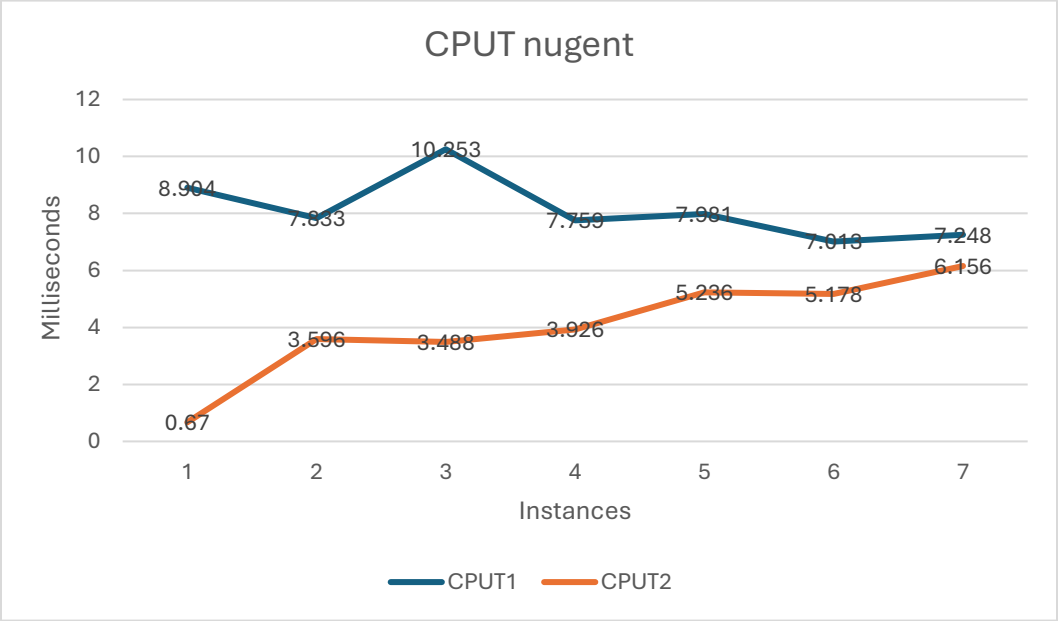

Fig. 5. Nugent CPUT comparison

6 Conclusions and future work

As indicated at the beginning of this document, the main objective of this work is the adaptation of the structures of a neighborhood in the post-processing phase of GRASP to compare the best value found (BVF) against the result of the best-known value found (BKVF). With the results that are intended to be obtained, it will be possible to visualize and verify if the implemented procedure is considered a robust metaheuristic to solve this type of problems.

In this work, the implementation of a well-known GRASP metaheuristic in a JAVA programming language for the search for approximate solutions for a well-known classical COP, NP-hard QAP, has been discussed. The search is carried out through a 2-exchange neighborhood structure. Tests were performed for a wide range of large-scale instances taken from QAPLIB (Burkard et al, 2024) with two separate parameter sets to constrain the RLC component of GRASP in order to show variation in the efficiency of this approach. For the 29 instances, it is observed that 72% of the test sets have a gap of $0 < GAP < 1$, the remaining 28% includes the other instances. Therefore, and according to the data provided, it is shown that the implementation of GRASP is efficient and at the same time robust. Based on the discussion of the results in section 5, we claim the robust nature of this work and that the RLC cuts help in the construction phase to produce high-quality solutions to minimize the effort of the post-processing phase. As future work, we will consider introducing a combination of GRASP with recently created bioinspired MH such as Gray Wolf Optimizer – (GWO), Firefly Algorithm (FA) and harmonic search among others.

References

Aksan, Y., Dokeroglu, T., & Cosar, A. (2017). A stagnation-aware cooperative parallel breakout local search algorithm for the quadratic assignment problem. *Computers & Industrial Engineering*, 103, 105-115. https://doi.org/10.1016/j.cie.2016.11.023

Burkard, R.E., Karisc, S.E., & Rendl, F. QAPLIB- A Quadractic Assignment Problem. Retrieved July 23,2014, from http://www.imm.dtu.dk/~sk/qaplib/ins.html

Cela, E. (1995) The Quadratic Assignment Problem: Special Cases and Relatives (Technical report). Institut für Mathematik B Technische Iniversität Graz. Graz Austria.

Chmiel, W., Kadłuczka, P., Kwiecień, J., & Filipowicz, B. (2017). A comparison of nature inspired algorithms for the quadratic assignment problema. *Bulletin of the Polish Academy of Sciences. Technical Sciences*, 65 (4), 513-522. https://doi.org/10.1515/bpasts-2017-0056

Díaz,A.D., Glober,H., Ghaziri,M, Gonzalez,J.L.,Moscato,P., & Tseng,F.T. (1996). *Optimización Heurística y Redes Neuronales en Dirección de Operaciones e Ingeniería*. Paraninfo, Madrid.

Dokeroglu, T., Sevenic, E., y Cosar, A. (2019). Artificial bee colony optimization for the quadratic assignment problema. *Applied Soft Computing Journal*, 76, 595-60. https://doi.org/10.1016/j.asoc.2019.01.001

Mouayni. El., Demesure, I., Bril-El Haouzi G., Charpentier, H., P., & Siadat, A. (2019). Jobs scheduling within Industry 4.0 with consideration of worker's fatigue and reliability using Greedy Randomized Adaptive Search Procedure. *IFAC-PapersOnLine*, 52(19), 85-90. https://doi.org/10.1016/j.ifacol.2019.12.114

Feo, T.A., & Resendes, M.G.C. (1995). Greedy Randomized Adaptive Search Procedures. *Journal of Global Optimization*, 6,109-133. https://doi.org/10.1007/BF01096763

Festa, P., y Resendes, M.G.C. (2011). GRASP: basic components and enhancements. *Telecommun Syst*., 46, 253-271. https://doi.org/10.1007/s11235-010-9289-z

Hafiz, F., y Abdennour, A. (2016). Particle Swarm Algorithm variants for Quadratic Assignment Problems-A probabilistic learning approach. *Expert Systems with Applications*, 44, 413-431. https://doi.org/10.1016/j.eswa.2015.09.032

Hameed, Asaad Shakir, Mutar, Modhi Lafta, Alrikabi, Haiffa Muhsan B., Ahmed, Zakir Hussain, Abdul–Razaq, Abeer A., Nasser., & Huda Karem. (2021). A Hybrid Method Integrating a Discrete Differential Evolution Algorithm with Tabu Search Algorithm for the Quadratic Assignment Problem: A New Approach for Locating Hospital Departments. *Mathematical Problems in Engineering*, 2021, 1 -21. https://doi.org/10.1155/2021/6653056

Koopmans,T.C., & Beckmann, M.J. (1957). Assignment problems and the location of economic activities. *Econometrica,* 25, 53-76. https://doi.org/10.2307/1907742

Kumar, M., Sahu, A., & Mitra, Pinaki. (2021). A comparison of different metaheuristics for the quadratic assignment problem in accelerated systems. *Applied Soft Computing Journal*, 100, 1-16. https://doi.org/10.1016/j.asoc.2020.106927

Li, Yong., Pardalos, P.M., & Resende, M.G.C. (1994). A Greedy Randomized Adaptative Search Procedure for the Quadratic Assignment Problem" In P.M. Pardalos and H. Wolkowicz, (eds.): Quadratic assignment and related problems, *Series on Discrete Mathematics and Theoretical Computer Science* (Of DIMACS, Vol. 16, pp.237-261). American Mathematical Society, Rhode Island

Mishmast, H., & Gelareh, S. (2007). A Survey of Meta-Heuristic Solution Methods for the Quadratic Assignment Problem, *Applied Mathematical Sciences*, 1 (46), 2293 – 2312

Obdelkafi, O., Idoumghar, L., & Lepagnot, J. (2016). Data Exchange Topologies for the DISCO- HITS Algorithm to Solve the QAP. *International Conference on Swarm Intelligence Based Optimization* (pp. 57-64) Mulhouse, France. DOI: 10.1007/978-3-319-50307-3 4

Palmieri, F., Fiore, U., Ricciardi, S., $ Castiglione, A. (2016). GRASP- based resourced re- optimization for effective big data access in federated clouds. *Future Generation Computer Systems,* 54, 168-179. https://doi.org/10.1016/j.future.2015.01.017

Pardalos, P.M., Pitssoulis, L.S., & Resende, M.G.C. (1995). A Parallel GRASP implementation for the Quadratic Assignment Problem. In A. Ferreira & J. Rolim (Eds.), *Parallel Algorithms for Irregularly Structured Problems – Irregular'94* (Lectures in Computer Science. Vol. 1253, pp.111-130). Kluwer Academic Publishers.

Resende,G.C.M., y Riberiro, C. (2016). *Optimization by GRASP: Greedy Randomize Adaptive Search Procedures*. Springer, New York.

Resende, M.G.C., Pardalos, P.M., & Li, Yong. (1996). Algorithm 754: Fortran subroutines for approximate solution of dense quadratic assignment problem using GRASP. *ACM Transactions on mathematical software*, 22 (1), 104-118. https://doi.org/10.1145/225545.225553

Riffi, M. E., Saju, Y., & Barkatou, M. (2017). Incorporating a modified uniform crossover and 2-exchange neighborhood mechanism in a discrete bat algorithm to solve the quadratic assignment problem. *Egyptian Informatics Journal*, 18(3), 221-232, https://doi.org/10.1016/j.eij.2017.02.003

Roucairol, C. (1987). A parallel branch and bound algorithm for the quadratic assignment problem. *Discrete Applied Mathematics*, 18, 211-225. https://doi.org/10.1016/0166-218X(87)90022-9

Sahni, S., & Gonzalez T. (1976). P-complete aproximations problems. *J. Asssoc. Comp. Machine*, 23 (3), 555-565. https://dl.acm.org/doi/10.1145/321958.321975

Saifullah Hussin, M., & Stützle, T. (2014). Tabu search vs. Simulated annealing as a function of the size of quadratic assignment problem instances. *Computers & Operations Research,* 43, 286-291, https://doi.org/10.1016/j.cor.2013.10.007

Silva, A., Coelho, L.C. & Darvish, M. (2021). Quadratic assignment problem variants: A survey and an effective parallel memetic iterated tabu search. *European Journal of Operational Research*, 292 (3), 1066-1084, https://doi.org/10.1016/j.ejor.2020.11.035

Skorin-kapov, J. (1990). Tabu Applied to the Quadratic Assignment Problem. *ORSA Journal on Computing,* 2(1), 33-45, https://doi.org/10.1287/ijoc.2.1.33

Tosun, U. (2014). A New Recombination Operator for the Genetic Algorithm Solution of the Quadratic Assignment Problem. *Procedia Computer Science*, 32, 29–36 https://doi.org/10.1016/j.procs.2014.05.394

Zhou, Y., Hao, J. K., y Duval, B. (2020). "Frequent Pattern-Based Search: A Case Study on the Quadratic Assignment Problem". IEEE Transactions on Systems, Man, and Cybernetics: Systems, pp. 1-13, DOI: 10.1109/TSMC.2020.3027860