Metaheuristics with Local Search Miscellany Applied to the Quadratic Assignment Problem for Large-Scale Instances

Rogelio González-Velázquez[1], Erika Granillo-Martínez[2](✉), María Beatriz Bernábe-Loranca[1], and Jairo E. Powell-González[1]

[1] Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, Prol. 14 sur Esq. Av. Sn. Claudio, C.P 72590 Puebla, Mexico
jairoe.powell@viep.com.mx

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

Abstract. The quadratic assignment problem (QAP) is classified as an NP-hard problem, so metaheuristic procedures are often used to solve it. QAP is a classic combinatorial optimization problem with real applications, for example in supply chain, logistics, manufacturing, finance, among others. In this article, to search for QAP solutions, the design of a program code for the Greedy Randomized Adaptive Search Procedure (GRASP) metaheuristic was implemented with three different neighborhood structures contained in k-exchange mode in order to perform the local search. The experimental procedure was applied for large-scale test instances available from QAPLIB. Finally, the results of the approximations to the optimal solutions are reported.

Keywords: Metaheuristics · NP-hard · k-exchange · Neighborhood

1 Introduction

Metaheuristic procedures are a class of approximation methods, designed to solve difficult combinatorial optimization problems [1]. Optimization problems are based on choosing the best configuration from a set of feasible solutions to achieve an objective [2] and are divided into two categories: problems with continuous and discrete variables. For this case, the second category will be taken that tries to find an objective that is taken 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 [2] as an example, Greedy Random Adaptive Search Procedures (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 by a local search algorithm. The best of all the solutions examined is the final result [3, 4].

J. C. Figueroa-García et al. (Eds.): WEA 2021, CCIS 1431, pp. 327–334, 2021.
https://doi.org/10.1007/978-3-030-86702-7_28

The main contribution of this work is based on the experimentation and analysis of the results in the method, obtaining a good initial solution and subsequently improving it.

This paper is organized as follows: Sect. 1, contains an introduction to this work, in Sect. 2, related work is presented. In Sect. 3, the formulation for the quadratic Assignment Problem (QAP); Sect. 4, shows the metaheuristics, Sect. 4.1, contains a description of GRASP, as well as the design of the pseudocode, subsequently in Sect. 4.2 the use of GRASP for the Quadratic Assignment Problem.

Section 5, shows the experiments that were done, and the results that were obtained, respectively. Finally, in Sect. 6, it will find a discussion, conclusions and future work.

2 Related Work

The Quadratic Assignment Problem (QAP) is a combinatorial optimization problem that consists in finding an optimal allocation of n resources to n locations in order to minimize the cost of transportation, additionally, two matrixes are needed, one for the requirements of the units to be transported and the second is the cost of transport per unit between the localities.

QAP was proposed by [5] in 1957. Later in 1976, Shani and González proved that QAP is an NP-complete problem [6]. So far, optimal solutions have been found using exact methods for instances of size 30 [7]. The QAP is used in many applications such as: computer keyboard design, manufacturing programming, airport terminal design, and communication processes. An exact branch and bound algorithm is used with some variants to solve the QAP and was proposed by [7], however, recently solutions were proposed with different metaheuristics techniques such as [8, 9]: genetic algorithms, simulated annealing, tabu search [10] and GRASP [3, 11]. In [12] implemented GRASP in parallel for QAP.

Other works related to the search for QAP solutions are based on the particle swarm algorithm, recombination of operators for genetic algorithms and stagnation aware cooperative parallel to local search [13, 14].

QAP in large instances is a notably hard problem to find a solution to and the performance of metaheuristic algorithms varies, as mentioned in [15] where two algorithms are compared with results that depend on the size of the problem.

In [16] simulated annealing, particle swarm optimization, genetic algorithms, iterated local search, tabu search and crow search algorithms are implemented and compared in massive parallel processing units to solve QAP for large instances. Another example of parallel processing can be seen in [17] where this metaheuristic is implemented with Tabu search to work with large size QAP.

3 Formulation for the Quadratic Assignment Problem

The QAP consists of finding an optimal allocation that minimizes the cost of transporting materials, among n facilities in n locations, considering the distance between locations and the flow of materials between facilities. The QAP can be formulated by a combinatorial optimization model (CP).

Given a set N = {1, 2,.., n} and two symmetric matrices of size $n \ x \ n$ where: F = (f_{ij}) y D = (d_{kl}), a permutation p Є Π_N must be found that minimizes

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

Where Π_N is the set of all permutations of N, F is the material flow matrix between facilities and D is the distance matrix between cities.

4 Metaheuristics

The major drawbacks that heuristic techniques face is the existence of local optimum that are not absolute. If during the search there is a local optimum, the heuristic could not continue the process and would be “trapped” at the same point. In order to solve the problem, it is recommended to restart the search from another initial solution and verify that the new search explores other paths.

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 try to solve the above drawbacks. Most metaheuristics are developed with neighborhood search methods.

The word metaheuristics was coined [1] at the same time that the term Tabu Search emerged (1986). A metaheuristic is a master strategy that guides and modifies other heuristics to generate better solutions than are normally presented by other methods [18].

There are several 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 [19] at the time of studying coverage problems of high combinatorial complexity [3]. 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 that is later improved by post-processing to perfect the solution obtained in the first stage until obtaining a local optimum.

There are works where this metaheuristic is applied for optimization problems in big data [20]. Additionally, 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 [21].

4.1 Greedy Randomized Adaptive Search Problem GRASP

A GRASP is an iterative process, each iteration consists of two steps: the construction phase and the local search procedure. In the first, an initial feasible solution is construct, later it is improved by means of an exchange procedure until obtaining a local optimum [4].

Once the two phases have been executed, the solution obtained is stored and another iteration is carried out, each time saving the best solution that has been found so far.

An algorithm that exemplifies metaheuristics is shown in Fig. 1.

```
Procedure GRASP
  InputInstance();
  While (stop criterion not satisfied) do
        ConstructSolutionGreedyRandomizeAdaptative();
        Post-proccesing();
        UpdateSolution();
  End {While}
  Return (Best solution)
End {GRASP}
```

Fig. 1. Generic GRASP pseudocode.

The general description of the main components of GRASP are: The Greedy component that uses a myopic algorithm for the selection of the components that guide the construction of solutions, the Randomized used for the random selections of an elite list of candidates that determine the path of search, the Adaptive has the mission of updating each result obtained from the components of the solution that is built [22].

4.2 GRASP for the Quadratic Assignment Problem

The GRASP design has been used by some researchers to solve the QAP for different instances [1, 4, 6]. 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, then one is randomly taken of these candidates from which the first 2 assignments are derived. Stage 2, the remaining $n-2$ assignments are added in relation to the Greedy procedure, once the process is finished, the permutation is completed, producing a feasible solution. Phase 2 of improvement: 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 globally optimal.

The neighborhood structures used are: 2-exchange, N* and λ-exchange. In practice, the flow matrixes are taken, as well as the distance matrix and their elements are listed separately. The flows (matrix) are ordered from highest to lowest and the distances from least to greatest, both lists are restricted with a parameter $0 < \alpha < 1$, they are multiplied generating a new list of elements of the form $f_{ij} * d_{kl}$ that contains large flows and short distances. This list is restricted with a parameter $0 < \beta < 1$, with these operations you have a restricted list of candidates (CRL), from this list an element of the form is randomly selected $f_{ij} * d_{kl}$, 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 $n-2$ components remain to be assigned, again with a greedy we calculate the C_{ik} costs with respect to the 2 assignments against the remaining possible assignments, another CRL is formed and one of these candidates is selected with which the third assignment is generated and so on. Until completing the permutation of n components called initial solution S0 which is subjected to phase 2, called improvement phase, this is an iteration of GRASP [23].

5 Results

This section shows the results obtained for instances taken from [24], the instances dimensions are greater than 42 up to 100 considering as a great scale. A data table is shown for each of the three implemented neighborhood structures. It also shows three comparative tables of the neighborhood structures in terms of the number of iterations, the execution time TCPU and the percentage in which they reach the optimal or best known value. Likewise, a table with the GAP percentages is presented.

All the results shown in this section were obtained by restricting the list of candidates (RLC) with the parameters $\alpha = 0.2$ and $\beta = 0.3$, which were determined experimentally. The neighborhood topologies is discussed in [3, 25] in which the size of the neighborhood is commented as follows: for 2-exchange is C_n^2, λ-exchange, for the size is $\frac{n^3}{24}$, N^* is $\frac{n^4}{8}$, finally, the pseudocode algorithms for the previous neighborhood structures are shown in [3].

Table 1. GRASP results with local search λ-exchange, Skorin-Kapov instances

Instance	Neighborhood	Option	BKV	BFV GRASP	Error %	TCPU
Sko 42	λ-exchange	Random	15812	15966	0.97	36380
Sko 49	λ-exchange	Greedy 2 stage	23386	23596	0.90	64381
Sko 56	λ-exchange	Greedy 2 stage	34458	34694	0.68	121372
Sko 64	λ-exchange	N. initial Sol.	48498	48686	0.39	207061
Sko 72	λ-exchange	Random	66256	66686	0.65	282684
Sko 81	λ-exchange	Random	90998	91772	0.85	462095
Sko 90	λ-exchange	Greedy 2 stage	115534	116582	0.91	797597
Sko 100a	λ-exchange	Random	152002	153208	0.79	1064643
Sko 100b	λ-exchange	Greedy 2 stage	153890	155188	0.84	1251686
Sko 100c	λ-exchange	Random	147862	149074	0.82	11543734
Sko 100d	λ-exchange	Greedy 2 stage	149576	150958	0.92	1245131
Sko 100e	λ-exchange	Random	149150	150454	0.87	1141179

Table 2. GRASP results with local search 2-exchange, Skorin-Kapov instances

Instance	Neighborhood	Option	BKV	BFV GRASP	% error	TCPU
Sko 42	2-exchange	Random	15812	15922	0.70	28288
Sko 49	2-exchange	Greedy 2 stage	23386	23594	0.89	64184
Sko 56	2-exchange	Random	34458	34840	1.11	96746
Sko 64	2-exchange	Greedy 2 stage	48498	49002	1.04	197513
Sko 72	2-exchange	Random	66256	66794	0.81	281584
Sko 81	2-exchange	Greedy 2 stage	90998	91640	0.71	537896
Sko 90	2-exchange	Random	115534	116448	0.79	757746
Sko 100a	2-exchange	Random	152002	153012	0.66	1162220
Sko 100b	2-exchange	Random	153890	154916	0.67	1262583
Sko 100c	2-exchange	Greedy 2 stage	147862	148736	0.59	1290115
Sko 100d	2 exchange	Greedy 2 Stage	149576	150800	0.82	647448
Sko 100e	2 exchange	Greedy 2 stage	149150	150724	1.06	1372944

Table 3. GRASP results with local search N*, Skorin-Kapov instances

Instance	Neighborhood	Option	BKV	BFV GRASP	Error %	TCPU
Sko 42	N*	Random	15812	15950	0.87	351405
Sko 49	N*	Greedy 2 stage	23386	23554	0.72	606638
Sko 56	N*	Random	34458	34780	0.93	1318201
Sko 64	N*	Greedy 2 stage	48498	48912	0.85	192592
Sko 72	N*	Random	66256	66830	0.87	289931
Sko 81	N*	Random	90998	91868	0.96	489550
Sko 90	N*	Greedy 2 stage	115534	116176	0.56	860121
Sko 100a	N*	Random	152002	152942	0.62	2186983
Sko 100b	N*	Random	153890	154928	0.67	1060393
Sko 100c	N*	Random	147862	148854	0.67	1094756
Sko 100d	N*	Random	149576	150772	0.80	1197065
Sko 100e	N*	Random	149150	150704	1.04	961112956

6 Conclusions

The results of Tables 1, 2 and 3 shows that GRASP is a robust metaheuristic in the search for solutions to NP-hard problems. The solutions reached in the work are compared with the best known values according with QAPLIB [24]. As shown in Tables 1, 2 and 3 and in the % error column, the results obtained oscillate between 0.56 and 1.11 in error percentage. The seventh column shows the execution time for each instances in milliseconds, which is a reasonable computing time for large-scale instances.

Within the research it was shown that for the Sko 42 instance case, the best implementation was 2-exchange, in the case of Sko 64 the best combination was λ-exchange and for Sko 90 the best execution was Greedy 2 stage. Talking about the case of matrices of size 100 that represent the largest scale, the results are considered favorable because the error is less than 1%, only in the case of the Sko100e instance the maximum error is 1.06.

In the present document, the results obtained were carried out in the Java programming language for the three local searches with GRASP and were executed on an Intel i7 processor. Therefore, with the results shown, the usefulness of metaheuristics to solve highly complex problems was verified, as well as the verification to obtain approximate optimal solutions, given the inefficiency of the exact methods.

As future work, it is proposed to implement the metaheuristic variable neighborhood search (VNS) with the neighborhood structures 2-exchange, N* and λ-exchange as an improvement phase for GRASP to form a hybrid GRASP-VNS.

References

1. Díaz, A.D., Glober, F., Ghaziri, H.M., Gonzalez, J.L., Moscato, P., Tseng, F.T.: Optimización Heurística y Redes Neuronales en Dirección de Operaciones e Ingeniería. Ma-drid (1996)
2. Cela, E.: The Quadratic Assignment Problem: Special Cases and Relatives. Tesis doctoral. Institut für Mathematik B Technische Iniversität Graz, Graz, Austria (1995)
3. Li, Y., Pardalos, P.M., Resende, M.G.C.: A Greedy Randomized Adaptative Search Pro-cedure for the Quadratic Assignment Problem. In: Pardalos, P.M., Wolkowicz, H. (eds.) Quadratic Assignment and Related Problems, Vol. 16 of DIMACS Series on Discrete Mathematics and Theoretical Computer Science, pp. 237–261. American Mathematical Society, Rhode Island (1994)
4. Resende, M., Pardalos, P., Li, Y.: Algorithm 754: Fortran subroutines for approximate solution of dense quadratic assignment problems using GRASP. ACM Trans. Math. Softw. **22**(1), 104–118 (1996). https://doi.org/10.1145/225545.225553
5. Koopmans, T.C., Beckmann, M.J.: Assignment problems and the location of economic activities. Econometrica **25**, 53–76 (1957)
6. Sahni, S., Gonzalez, T.: P-complete approximations problems. J. Asssoc. Comp. Machine **23**, 555–565 (1976)
7. Roucairol, C.: A parallel branch and bound algorithm for the quadratic assignment problem. Discrete Appl. Math. **18**(2), 211–225 (1987). https://doi.org/10.1016/0166-218X(87)90022-9
8. Zhou, Y., Hao, J.K., Duval, B.: Frequent pattern-based search: a case study on the quadratic assignment problem. IEEE Trans. Syst. Man Cybern. Syst. (2020)
9. Hafiz, F., Abdennour, A.: Particle swarm algorithm variants for quadratic assignment problems—a probabilistic learning approach. Expert Syst. Appl. **44**, 413–431 (2016)
10. Skorin-Kapov, J.: Tabu search applied to the quadratic assignment problem. ORSA J. Comput. **2**(1), 33–45 (1990). https://doi.org/10.1287/ijoc.2.1.33
11. Chmiel, W., Kadłuczka, P., Kwiecień, J., Filipowicz, B.: A comparison of nature inspired algorithms for the quadratic assignment problem. Bull. Pol. Acad. Sci. Tech. Sci. 65(4) (2017)
12. Pardalos, P.M., Pitssoulis, L.S., Resende, M.G.C.: A parallel GRASP implementation for the quadratic assignment problem. In: Ferreira, A., Rolim, J. (eds.) Parallel Algorithms for Irregularly Structured Problems – Irregular 1994, pp. 111–130. Klower, Boston (1995)

13. Aksan, Y., Dokeroglu, T., Cosar, A.: A stagnation-aware cooperative parallel breakout local search algorithm for the quadratic assignment problem. Comput. Ind. Eng. **103**, 105–115 (2017)
14. Tosun, U.: A new recombination operator for the genetic algorithm solution of the quadratic assignment problem. Procedia Comput. Sci. **32**, 29–36 (2014)
15. Saifullah Hussin, M., Stützle, T.: Tabu search vs. simulated annealing as a function of the size of quadratic assignment problem instances. Comput. Oper. Res. **43**(2), 286–291 (2014)
16. Kumar, M., Sahu, A., Mitra, P.: A comparison of different metaheuristics for the quadratic assignment problem in accelerated systems. Appl. Soft Comput. **100**, 106927 (2021). https://doi.org/10.1016/j.asoc.2020.106927
17. Dokeroglu, T., Sevinc, E., Cosar, A.: Artificial bee colony optimization for the quadratic assignment problem. Appl. Soft Comput. J. **76**, 595–606 (2019)
18. Mishmast, H., Gelareh, S.: A survey of meta-heuristic solution methods for the quadratic assignment problem. Appl. Math. Sci. **46**(1), 2293–2312 (2007)
19. Feo, T.A., Resende, M.G.C.: Greedy randomized adaptive search procedures. J. Global Optim. **6**, 109–133 (1995)
20. Palmieri, F., Fiore, U., Ricciardi, S., Castiglione, A.: GRASP-based resourced re-optimization for effective big data access in federated clouds. Futur. Gener. Comput. Syst. **54**, 168–179 (2016)
21. Festa, P., Resende, M.G.C.: GRASP: basic components and enhancements. Telecommun Syst. **46**, 253–271 (2011). https://doi.org/10.1007/s11235-010-9289-z
22. El Mouayni, I., Demesure, G., Bril-El Haouzi, H., Charpentier, P., Siadat, A.: 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 (2019). https://doi.org/10.1016/j.ifacol.2019.12.114
23. Riffi, M.E., Saji, Y., Barkatou, M.: Incorporating a modified uniform crossover and 2-exchange neighborhood mechanism in a discrete bat algorithm to solve the quadratic assignment problem. Egypt. Informat. J. **18**(3), 221–232 (2017)
24. Burkard, R.E., Karisc, S.E., Rendl, F.: QAPLIB – A Quadratic Assignment Problem, Library, http://www.imm.dtu.dk/~sk/qaplib/ins.html
25. Obdelkafi, O., Idoumghar, L., Lepagnot, J., Brévilliers, M.: Data exchange topologies for the DISCO-HITS algorithm to solve the QAP. In Siarry, P., et al. (eds.) ICSIBO, LNCS 10103, pp. 57–64 (2016). https://doi.org/10.1007/978-3-319-50307-3_4