# Quadraticproblem assignment

Quadratic assignment problem is one of the most known and challenging combinatorial optimization problems. In this study, a new tabu search algorithm is proposed to solve the quadratic assignment problem. The performance of the proposed approach is tested on with 25, 50 and 100-department instances which are taken from QAPLib.

Quadratic assignment problem (QAP) is introduced by Koopmans and Beckman in 1957. It can be described as follows: given n? n matrices A = (aij ) and B = (bij ) where matrices represent ? ow and distance, respectively. Find a permutation ? ? minimizing n n min f (? ) Q ?? (n) = i= 1 j= 1 aij b? i ? j where (n) is the set of permutations of n elements [1]. Shani and Gonzalez have shown that QAP is NP-hard [8]. Solving this problem optimality for the large instances is computationally infeasible.

Therefore, heuristic approaches have to be used for solving medium- and large-scale QAPs. In this 1 study, tabu search (TS) algorithm is used to solve QAP. Tabu search technique was developed by Glover [2, 3]. This method has become very popular and is widely used for a variety of problems [4]. Tabu search is based on the neighborhood search with local-optima avoidance but in a rather deterministic way. The key idea of tabu search is allowing climbing moves when no improving neighboring solution exists. However, some moves are to be forbidden at a present search iteration in order to avoid cycling.

The proposed tabu search algorithm is tested with di? erent neighborhood sizes, tabu tenors and termination conditions. During the tests 25department [7] and 50, 100-department [9] instances are used which are taken from

QAPLib. This paper will proceed as follows. In section 2, mathematical model of the QAP is given. In section 3, proposed SA is presented. Finally in section 4 computational results are given. 2 Problem Formulation In this study, QAP with n departments and n locations with minimizing costs between placed departments is considered. The cost is obtained by product of ? ows and distances between departments.

Integer linear model of the problem is as follows. Set N : Deparment (or location) set where N = {1, 2, ... , n} Indices i, k : The depatment index used as unique identi? er for each department. j, l : The location index used as unique identi? er for each location. Parameters n : Number of departments(or locations) aik : Total ? ow department i to department k. bjl : Distance location j to department l. Decision Variables 1 If department i assigned to location j xij = 0 otherwise 2 Model n n n n min i= 1 j= 1 k= 1 l= 1 aik bjl xij xkl n (1) s. t. xij = 1 ? i j= 1 n (2) xij = 1 ? j i= 1 (3) (4) xij ? {0, 1} ? , j Equation (1) is the objective function that minimizes the costs. Equation (2) ensures that each department is assigned exactly one location. Similarly, equation (3) ensures that each location is assigned to exactly one department. Equation (4) is the binary integrality constraints. 3 Tabu Search Tabu search starts from an initial solution s where s ? S, S is the set of solution of combinatorial optimization problem. At each step of the procedure, a set N (s) of the neighboring solutions of the current solution s is considered and the move that improves most the objective function value is chosen.

If there are no improving moves, tabu search chooses one that least degrades the objective function. In order to avoid the returning to the local

optimal solution just visited, the reverse move must be prohibited. This is done by storing this move in a memory (or more precisely short-time-memory) managed like a circular list and called a tabu list. In that way, the tabu list keeps information on the last moves which have been done during the search process. However, it might be worth returning after a while to a solution visited previously to search in another direction.

Consequently, an aspiration criterion is introduced to permit the tabu status to be dropped under certain favorable circumstances [6]. In the following subsections elements of the proposed tabu search algorithm is explained. 3 Departments Locations 1 3 2 5 3 1 4 8 5 6 6 2 7 4 8 7 Table 1: Example solution representation Figure 1: Illustrative example for the solution representation 3. 1 Solution Representation In this study, solution is represented as permutation of locations (? ). Each solution represents assignment of i? th department to ? i ? th locations where i = {1, 2, ... , n}. Sample representation is shown in the Table 1.

In Table 1, there are n = 8 departments (and locations), and the example solution represents 1 ? 3, 2 ? 5,... , 8 ? 7 department-location assignments. The department-location assignment is also illustrated in Fig. 1. 3. 2 Cost Function The equation (5) is the cost function of the explained solution (? ) representation. The goal of the algorithm is to ? nd a solution which minimizes this value. n ?? n min f (? ) = Q (n) i= 1 j= 1 aij b? i ? j (5) 3. 3 Neighborhood Structure In this study, a standard swap operation is used as a neighborhood structure. Basically, location of two di? erent department is exchanged.

In this neighborhood structure (Neighborhood-1), all possible swaps are evaluated. So that complexity of the neighborhood structure is O(n2 ). The algorithm of the neighborhood structure is given in Algorithm 1. 4 Input: Solution s ? S Output: Neighbors of s for i 3. 4 Tabu List In the propped tabu search algorithm, moves are stored in the tabu list. For example, if the locations i and j are swapped, than swap(i, j) is added to tabu list. This tabu also restricts the swap(j, i) move. Another point about the tabu search is tabu list size. Two di? erent tabu lists are tested which are ? xed-sized and dynamic-sized.

In the ? xed-sized 5 No 1 2 3 4 5 6 Neighbors 2 1 3 4 3 2 1 4 4 2 3 1 1 3 2 4 1 4 3 2 1 2 4 3 No 1 2 3 Neighbors 2 1 3 4 1 3 2 4 1 2 4 3 Table 2: Comparison of neighbors for Neighborhood-1 and Neighborhood-2 tabu list, size of the tabu list remains same until the termination condition occurs. In the dynamic-sized tabu list, tabu list size is decreased by using equation (6). In this function T0 represents the initial tabu list size, TN represents ? nal tabu list size and n iter represents the number of iterations. This idea is similar to the cooling schedule for the simulated annealing. i? 1 ) + TN (6) Ti = (T0 ?

TN )(1 + cos( 2 n iter One of the most advantage side of the tabu search is memory usage. In this study, frequency based tabu list is implemented to use the bene? t of long term memory. Previously visited moves are penalized to devirsify the proposed tabu search. For this purpose, frequency of each visited move is stored and each move is penalized with equation (7). In this equation Zp represents penalized objective value, Zi, j represents the objective value of when swap(i, j) is applied, fi, j represents the frequency of

swap(i, j) and n iter represents the number of iterations. Zp = Zi, j + Zi, j fi, j n iter (7) 3. 5

Aspiration Criterion In the proposed tabu search an aspiration criterion is used. The tabu status of an attribute can be revoked if that would allow the search to reach a solution of smaller cost than the best known solution having the given attribute. 6 3. 6 Termination Criterion In tabu search procedure, two di? erent stopping criteria is used which are as follows. • Maximum number of iterations (n iter). • If the best-known solution is not improved consecutive " n imp" iterations. 4 Computational Results Proposed tabu search algorithm is tested on 25, 50 and 100 departments test instances which are taken from QAPLib.

Proposed algorithm is coded in C++ and all tests are conducted in Linux machine with 32GB ram. As the tabu search is an deterministic search technique, if the input solution and the other parameters of the tabu search remain same there will be no di? erence on the obtained solution between di? erent replicas. So that, only one replica is taken for each test case, and best obtained solution for the replica is reported. The test cases are as follows: • Initial solution is changed 5 times. • Di? erent tabu list sizes (t size) are tested. t size = {6, 8, 10} • Dynamic tabu list size is tested. Tabu list size is de? ned by using Equation (6). E? ect of aspiration criterion (with and without aspiration criterion). • Number of evaluated neighbors n 2 and n ? 1 • E? ect of long-term memory (with and without frequency based tabu list). These cases are tested by changing each of them one by one. So that, initial parameter setting should be de? ned. The initial parameters are as follows; • Initial solution is taken same for all tests. 7 • Fixed-sized tabu list and t size

= 8. • No aspiration criterion. • No long-term memory. • Neighborhood-1 is used, so that number of evaluated neighbors is n . 2 • Maximum number of iteration is used for the stopping criterion, and n iter = 250.

Nug25 test instance is solved with these initial parameters, and 3762 is obtained. During the evaluation process current solution and best-known solution are logged, and by using these logs two ? gures are plotted in Fig. 2. As seen from Fig. 2b, best solution is improved when the search procedure is close to termination. So that, best-known solution may be improved if the n iter is increased. A new replica is taken with n iter = 300, but no change is observed on the best-obtained solution. E? ect of Initial solution Five di? erent initial solution is evaluated, and following results are obtained 3748, 3768, 3772, 3744, 3788.

The average of the solutions is 3764 and the range of the solutions is 44. Relative di? erence gives more meaningful results. Relative range is obtained by dividing the range of the solutions to the of the average value, and this value is 0. 012. These results show that the performance of algorithm is not highly e? ected with the input solution. On the other hand, we cannot say initial solution has no e? ect on the performance of the tabu search. Because, one of obtained solution is 3744 that is an optimal solution, another obtained solution is 3788 that is a little bit far from optimal solution. E? ct of tabu list size Two more di? erent tabu list sizes are evaluated which are 6 and 10, and the initial tabu list size is not changing during the evaluation. When the tabu list size is 8 the best ojective value was obtained as 3762. Tabu list size is decrease to 6, and the best obtained solution’s objective value is 3774. As an another test case, tabu list size is increased to 10, and the best obtained

solution’s objective value is 3762. From the obtained results, it is possible say that increasing the tabu list size improves the performance of tabu 8 (a) Change of the current solution with respect to iteration b) Change of the best obtained solution with respect to iteration Figure 2: Obtained solutions with respect to iterations 9 search. This situation occurs, because tabu list size is one of the diversi? cation operators of the tabu search. So, when we increase the tabu list size, we prevent tabu search from the early convergence. On the other, increasing size of the tabu search too much concludes with the bad result, because most of the possible moves may be restricted with the long lenghted tabu list. E? ect of dynamic tabu list As well as ? xed-length tabu list, dynamic-length tabu list is tested too.

Details of the dynamic tabu list is explained before. To determine the size of the tabu list in each iteration equation (6) is used. In this function, T0 is set to 10 and TN is set to 5. Non-integer results may be obtained by using equation (6), so that obtained function result is rounded to closest integer number to determine current iteration’s tabu list size. During the evaluation the result of the equation (6) (Fig. 3a) and current tabu list size (Fig. 3b) is logged, and the corresponding ? gures are given in Fig. 3. When the dynamic-length tabu list is used the best obtained solution’s objective value is 3770.

The solution is worse than previously obtained solutions. Probably, decreasing the size of the tabu list depending on iteration cause premature convergance. It is better to use ? xed-length tabu list instead of dynamic-length tabu list. E? ect of aspiration criterion In this test case e? ect of aspiration criterion is tested. Proposed tabu search is tested with initial test

cases including aspiration criterion, and the best obtained solution is 3748. Considerable di? erence is obtained with aspiretion criterion compared to initial parameter solution.

In this replica best-obtained solution is improved 5-times by using aspiration criterion. This considerable performance increase is obvious, because with aspiretion criterion it is allowed to increase the best-obtained solution even if this move is a tabu E? ect of neighborhood Instead of whole neighbourhood (Neighborhood-1), the performance of subset of this neighborhood (Neighborhood-2) is tested. E? ect of long-term memory Aspiration criterion and e? ect of long term memory is tested together. 10 (a) Value of the equation (6) with respect to iterations (b) Current tabu list size with respect to iterations

Figure 3: Dynamic tabu list size logs with respect to iterations 11 References [1] L. M. Gambardella, E. D. Taillard, and M. Dorigo. Ant colonies for the quadratic assignment problem. Journal of the Operational Research Society, 50: 167–176, 1999. [2] F. Glover. Tabu search: Part 1. ORSA Journal on Computing, 1: 190–206, 1989. [3] F. Glover. Tabu search: Part 2. ORSA Journal on Computing, 1: 4–32, 1990. [4] F. Glover and M. Laguna. Tabu Search. Kluwer, Dordrecht, 1997. [5] T. C. Koopmans and M. Beckmann. Assignment problems and the location of economics activities. Econometrica, 25: 53–76, 1957. 6] A. Misevicius. A tabu search algorithm for the quadratic assignment problem. Computational Optimization andApplications, 30: 95–111, 2005. [7] C. E. Nugent, T. E. Vollman, and J. Ruml. An experimental comparison of techniques for the assignment of facilities to locations. Operations Research, 16: 150–173, 1968. [8] S. Shani and T. Gonzalez. P-

complete approximation problems. Journal of the Association for Computing Machinery, 23: 555–565, 1976. [9] M. R. Wilhelm and T. L. Ward. Solving quadratic assignment problems by simulated annealing. IIE Transactions, 19(1): 107–119, 1987. 12