The Eighteenth International Symposium on Artificial Life and Robotics 2013 (AROB 18th '13),
Daejeon Convention Center, Daejeon, Korea, January 30-February 1, 2013

A metaheuristic for huge scale quadratic assignment problems

Hatsumi Nakaura[1], Fubito Toyama[1], Hiroshi Mori[1], Kenji Shoji[1], and Juichi Miyamichi[1]

[1]Graduate School of Engineering, Utsunomiya University, Japan
(Tel: 81-28-689-6244, Fax: 81-28-689-6244)

[1]hatsumi.nakaura@gmail.com

Abstract: The quadratic assignment problem (QAP) is one of the most difficult problems in the NP-hard class. Many metaheuristics have been proposed for the QAP. To evaluate the performance of these algorithms, QAPLIB which is a library of QAP instances is used. But QAPLIB does not have large scale QAP instances. QAPLIB cannot be used to evaluate the performance for large scale instances. Thus, these metaheuristics have been only applied to small scale QAP instances. It is difficult to apply these algorithms to large scale QAP instances because the number of combinations is huge. In this paper, we propose a metaheuristic approach for large scale QAP. The computational results showed that the proposed method outperformed conventional methods for huge scale QAP instances.

Keywords: Combinatorial optimization, Metaheuristics, Quadratic Assignment Problem, Tabu search

1 INTRODUCTION

The quadratic assignment problem (QAP) [1] is one of the most difficult problems in the NP-hard class [2]. In the QAP, n facilities have to be assigned to n locations at minimum cost when a $n \times n$ distance matrix and a $n \times n$ flow matrix are given. A solution is generally defined as a permutation of $\{1, 2, ..., n\}$, and the evaluation value of the solution is calculated using the two matrices. QAP has a large number of applications, including printed circuit design, facility layout, network design, and others.

Many metaheuristics have been proposed for the QAP [3][4]. To evaluate the performance of these algorithms, QAPLIB which is a library of QAP instances is used. But QAPLIB does not have large scale QAP instances. QAPLIB cannot be used to evaluate the performance for large scale instances. Thus, these metaheuristics have been only applied to small scale QAP instances. It is difficult to apply these algorithms to large scale QAP instances because the number of combinations is huge. 2-opt local search which uses the best improvement strategy is incorporated into the most of previous methods. 2-opt local search finds a better solution in a neighborhood of a current solution. The neighborhood is defined as the set of all solutions that can be reached from the current solution by swapping two elements in a permutation. Therefore, the number of solutions in the set of neighborhood is $n(n\text{-}1)/2$, where n represents the size of the problem. The computational cost of calculating neighborhood is $O(n^2)$. But after once evaluation values of the neighborhood are calculated, the computational cost of it can reduce to $O(n)$ by the difference value list between the current solution and the neighborhood solutions. If the problem size n is small, the first calculation of evaluation values of neighborhood solutions does not consume much time. But if the problem size n is huge, its calculation consumes huge time. Therefore it is hard to apply 2-opt local search to huge scale QAP instances. On the other hand, Genetic algorithms (GA) are often used for the QAP. GA works effectively for the problems in which the size of instances, n, is around 100. However, it is hard to apply GA to huge scale QAP instances (n is over 1000) because it takes too much time for GA to converge. Moreover, in order to improve search ability, most algorithms based on GA use hybrid GA (Memetic algorithm [5]) with 2-opt local search [6]. Therefore, GA cannot be applied for huge scale QAP because it is difficult even to apply 2-opt local search.

In this paper, a metaheuristic approach is proposed for the huge scale QAP. Robust Tabu Search (RoTS) [7] and the construction of difference value list are performed simultaneously in the proposed method. The construction of difference value list means to calculate evaluation values of the neighborhood solutions. Moving current solution is generally performed after the construction of the difference value list. But it is performed using the difference value list which is not constructed completely in the proposed method. The difference value list is constructed gradually while RoTS is performed. The proposed method can search for a relatively good solution in short time. Thus, it appears to be an appropriate algorithm for huge scale QAP instances.

In our experiments, the proposed method was compared to a fast local search and RoTS. The problem size n used in our experiments was set from 1000 to 10000. The

computational results showed that the proposed method outperformed the local search and RoTS for huge scale QAP instances.

2 QUADRATIC ASSIGNMENT PROBLEM

The QAP [7] is one of the most difficult problems in the NP-hard class. In the QAP, n facilities have to be assigned to n locations at minimum cost when a $n \times n$ distance matrix, and a $n \times n$ flow matrix are given. A solution is generally defined as a permutation of {1, 2, …, n}, and the evaluation value of the solution is calculated using the two matrices as follows:

$$F(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\pi(i)\pi(j)} \quad (1)$$

where π is a permutation of {1, 2, …, n}, a_{ij} is an element of flow matrix and b_{kl} is an element of distance matrix. The element a_{ij} represents the flow quantity between the facilities i and j. The element b_{kl} represents the distance between the locations k and l. If the facilities i and j are assigned to locations k and l, the part of permutation will be $\pi(i)=k$, $\pi(j)=l$. If π is the permutation that minimizes the objective expression (1), it is the optimum solution of the problem.

The most of benchmark instances of QAP are shown in QAPLIB. The largest instance in QAPLIB is tai256c in which the size n is 256. We set the size n from 1000 to 10000 for our experiments as larger instances than QAPLIB's.

3 PROPOSED ALGORITHM

The proposed algorithm is composed of two steps; the construction of the difference value list and Robust Tabu Search (RoTS). In the construction step, a gain between current solution and a 2-opt neighborhood solution which is randomly selected is calculated. Then its value is assigned to an element of the difference value list. 2-opt neighborhood is defined as a solution obtained by swapping two elements in a current solution. Therefore, in 2-opt neighborhoods, the number of neighborhood solutions is $n(n-1)/2$ (n: size of the problem). The difference value list (gain list) is defined as follows:

$$\Delta = \begin{bmatrix} * & \delta_{0,1} & \delta_{0,2} & \delta_{0,3} \\ * & * & \delta_{1,2} & \delta_{1,3} \\ * & * & * & \delta_{2,3} \\ * & * & * & * \end{bmatrix} \quad (2)$$

where $\delta_{i,j}$ represents a gain value obtained by subtracting the evaluation value of a current solution from the evaluation value of a neighborhood solution obtained by swapping two elements i and j in the current solution. At the beginning of a search, any gain values have not been calculated. Therefore the elements of the difference value list do not have the value. In the construction step, if swap elements 1 and 2 are first selected to obtain a neighborhood solution, gain value $\delta_{1,2}$ is calculated. Then the difference value list is shown as follows.

$$\Delta = \begin{bmatrix} * & * & * & * \\ * & * & \delta_{1,2} & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \quad (3)$$

Once a gain was assigned to the difference value list, the same two swap elements are not selected for the assignment process. When a current solution moves to a neighborhood solution, the assigned gain value is updated in the RoTS step. The computational cost required to calculate a gain is $O(n)$ and the cost of update a gain is $O(1)$.

The RoTS step is executed only if the assigned gain was a negative value which means the exchange makes current solution better. In the RoTS step, current solution moves to 2-opt neighborhoods within the limit of the difference value list. The tabu tenure which is the parameter of the RoTS is changed according to the number of assigned elements in the difference value list.

The pseudocode for the proposed method is shown in Fig. 1. n in Fig. 1 is the size of a problem, π is a permutation of current solution, π_{best} is a permutation of the best solution found so far, P is a set of pairs of elements which indicates the position of a permutation, and Δ is the difference value list: When $p=\{i,j\}$, $\Delta(p)=\delta_{i,j}$. After initializing P at the first double loop (lines 4 - 8 in Fig. 1.), P becomes as follows:

$$P = \{\{0,1\},\{0,2\},...,\{0,n\},...,\{n-1,n\}\} \quad (4)$$

The while loop between line 9 - 21 of Fig.1 is the main loop. The construction of the difference value list is performed in the first step of the main loop (line 10 - 14). The RoTS is applied in the next step (line 15 - 20). The construction of the difference value list and RoTS are repeated until some criterion is satisfied. In the construction step, first, a pair $p=\{i,j\}$ is selected randomly from the set P. Then the gain value, $\delta_{i,j}$ is calculated by SwapGain(π, p). In SwapGain(π, p), a difference value (gain value) between current solution and a 2-opt neighborhood solution obtained by swapping i and j of current solution is calculated. This gain value is assigned to the difference value list Δ. Then, the selected pair p is eliminated from the set P. The RoTS is performed if $\Delta(p)<0$ or $P=\emptyset$. Otherwise the performance of the RoTS is skipped. $\Delta(p)<0$ means the evaluation value of the neighborhood solution obtained by swapping pair p is better than current evaluation value. This skipping strategy can make the current solution better. The other condition to perform the RoTS, $P=\emptyset$, means the difference value list is constructed completely. Therefore, after the difference value

list was completed, the proposed method works the same as the RoTS. When the termination criterion was satisfied, the best solution, π_{best}, found in the search is returned.

```
Procedure Proposed Algorithm
begin
 1: generate a random solution π
 2: π_best := π;
 3: P := ∅;
 4: for i := 1 to n−1 do
 5:    for j := i+1 to n do
 6:       P := P∪{{i,j}};
 7:    endfor
 8: endfor
 9: while (termination criterion not satisfied) do
10:    if P≠∅ then
11:       select p from P randomly;
12:       Δ(p) := SwapGain(π, p);
13:       P := P−{p};
14:    endif
15:    if Δ(p)<0 or P=∅ then
16:       π := TabuSearch(π, Δ);
17:       if π<π_best then
18:          π_best := π;
19:       endif
20:    endif
21: endwhile
22: return π_best;
end
```

Fig. 1. Process of proposed algorithm

4 EXPERIMENTAL RESULTS

To evaluate the performance of the proposed method, we compare the performance of the proposed method with those of other methods on huge scale QAP instances. The size of huge instances used in our experiments was set from 1000 to 10000. The flow matrix and the distance matrix were asymmetric and their diagonals had non-zero elements. Values of these matrices were decided randomly between 0 and 99. Simple RoTS and 2-opt Local Search (2-opt-F) which uses first improvement strategy were applied to compare the performance. In simple RoTS, the complete difference value list is constructed at the beginning of search. In 2-opt Local Search using first improvement strategy, a current solution moves to 2-opt neighborhood obtained randomly if its gain value is better than that of current solution. The 2-opt-F does not make difference value list. Thus it does not consume much time to construct the difference value list and the convergence speed is very fast. A proposed algorithm without any conditions to execute RoTS is also tested for comparison of the performance of the proposed algorithm. In this method, RoTS is always executed after a gain value is assigned to the difference value list (even though $\Delta(p)>0$). We call this method "Proposed Algorithm (unconditional)''. In the experiment, the parameters of RoTS were the same as those used in the literature [7].

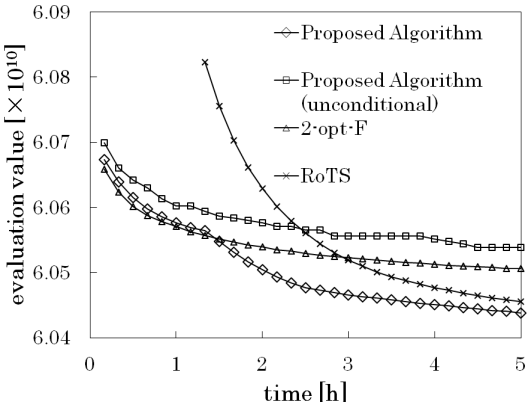

Fig. 2. Variation in evaluation value of best solution for n=5000 instance

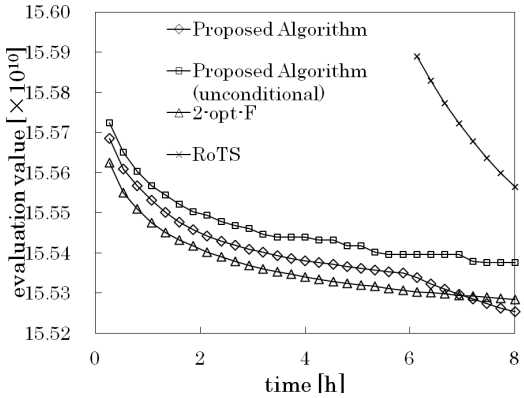

Fig. 3. Variation in evaluation value of best solution for n=8000 instance

First solutions are all the same for the methods on each instance. The experiment is executed on the same computer whose CPU is Intel Xeon X5650 2.67GHz and main memory has 64GB capacity. We used gcc4.4.6 as the compiler with –O3 option.

The results are shown in Table 1. n shows a size of instances, Time shows the the maximum run time, Best value shows the best evaluation values from all of 4 methods. "Deviation from best value" defined as the result of subtracting the "best value" from the evaluation values obtained by each method. Lower values are better than higher values, and 0 is the best value of 4 methods. As shown in Table 1, the proposed algorithm shows the best performance in 7 instances. Therefore the proposed algorithm is definitely very effective method for the huge scale QAP instances.

Fig. 2 and Fig. 3 show the changes in the evaluation value of the best solution against processing time for the

Table 1. Performance comparison to other methods

n	Time [h]	Best value	Deviation from best value			
			Proposed Algorithm	Proposed Algorithm (unconditional)	2-opt-F	RoTS
1000	1	2376195206	0	8.34×10^{6}	7.68×10^{6}	6.14×10^{5}
2000	2	9577858020	3.98×10^{4}	2.57×10^{7}	2.42×10^{7}	0
3000	3	21657258081	0	5.94×10^{7}	4.84×10^{7}	6.73×10^{6}
4000	4	38593328252	0	8.90×10^{7}	6.54×10^{7}	8.37×10^{6}
5000	5	60438065514	0	1.01×10^{8}	6.81×10^{7}	1.70×10^{7}
6000	6	87132554267	0	1.33×10^{8}	7.50×10^{7}	6.43×10^{7}
7000	7	118734259691	0	1.30×10^{8}	8.57×10^{7}	1.41×10^{8}
8000	8	155253875423	0	1.22×10^{8}	3.03×10^{7}	3.12×10^{8}
9000	9	196707554616	4.32×10^{7}	1.14×10^{8}	0	6.40×10^{8}
10000	10	242888642072	1.05×10^{8}	1.74×10^{8}	0	9.72×10^{8}

Table 2. Time to complete the difference value lists

n	Time [h]	Time rate [%]
1000	9.20×10^{-3}	0.920
2000	0.109	5.44
3000	0.294	9.78
4000	0.736	18.4
5000	1.37	27.5
6000	2.36	39.3
7000	3.81	54.4
8000	5.96	74.5
9000	-	-
10000	-	-

instances of n=5000 and n=8000, respectively. The horizontal axis represents the processing time and the vertical axis represents the evaluation value for the best solution obtained with each method. As shown in Fig. 2 and Fig. 3, the proposed method seems to find good solutions in shorter times than the previous method and the unconditional proposed method. The above results confirmed that the condition ($\Delta(p)<0$) to execute RoTS is an important factor to search huge QAP efficiently.

The time to complete the difference value lists on the proposed method is shown in Table 2. Time rate means the ratio of completion time to whole search time. On the instances of n=9000 and n=10000, the difference value lists were not completed until the ends of searches.

5 CONCLUSION

In this paper, we proposed an algorithm for huge scale QAP instances. The proposed algorithm is based on random 2-opt selection and Robust Tabu Search. Experimental results show the effectivities of the proposed algorithm for huge scale QAP instances. In order to use proposed algorithm effectively, the conditions to execute RoTS are required.

REFERENCES

[1] Koopmans TC, Beckman M. (1957), Assignment problems and the location of economic activities. Econometric 25:53-76

[2] S. Sahni, T. Gonzalez (1976), P-Complete Approximation Problems. Journal of the ACM 23(3):555-565

[3] A. S. Ramkumar, S. G. Ponnambalam, N. Jawahar et al (2008), Iterated fast local search algorithm for solving quadratic assignment problems. Robotics and Computer-Integrated Manufacturing 24:392-401

[4] Nilgun Fescioglu-Unver, Mieczyslaw M. Kokar (2011), Self Controlling Tabu Search algorithm for the Quadratic Assignment Problem. Computers & Industrial Engineering 60:310-319

[5] Moscato P. (2002), Memetic Algorithms. In: Handbook of Applied Optimization, P. M. Pardalos and M. G. C. Resende (eds), Oxford University Press, New York, pp. 157-168

[6] P. Merz, B. Freisleben (2000), Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Transactions on Evolutionary Computation 4:337-352

[7] E. D. Taillard (1991), Robust taboo search for the quadratic assignment problem. Parallel Computing 17:443-455