Comparing metaheuristic approaches to solving the quadratic assignment problem

COMP4240 Draft Paper

Mitchell Metcalfe

October 10, 2015

1 Introduction

The Quadratic Assignment Problem (QAP) was introduced by Koopmans and Beckmann (1957), motivated by the problem of optimally assigning manufacturing plants to locations in a way that maximises total revenue. The given a matrix of revenues that plants will generate in different locations, a matrix of distances between the locations, and a 'flow' matrix containing the quantity of commodity bundles that must be transported between plants per time unit.

More formally, the matrix $[r_{ki}]$ contains the known revenue each plant k will have a when placed at location i. The matrix $[a_{kl}]$ contains the required commodity flow between plant k and plant l, and the matrix $[b_{ij}]$ contains the cost of transport per unit flow between location i and location j. These are often referred to as the flow matrix and the distance matrix, respectively. The problem is then to find a permutation $\pi^* \in S_n$ (S_n is the symmetric group of order n) that maximises the total revenue:

$$\pi^* = \max_{\pi} \left(\sum_k a_{k\pi(k)} - \sum_k \sum_l a_{kl} b_{\pi(k)\pi(l)} \right)$$

where $\pi(k) = i$ indicates that plant k is to be placed at location i.

Many authors ignore the linear term $\sum_k a_{k\pi(k)}$, both because it is not necessary in some applications, and because it does not significantly increase the difficulty of the problem. For this reason, the QAP is often expressed as a minimisation problem as presented in (1).

$$\pi^* = \min_{\pi} \sum_k \sum_l a_{kl} b_{\pi(k)\pi(l)} \tag{1}$$

Koopmans and Beckmann established the place of the QAP as a difficult NP-Hard problem by indicating that the Travelling Salesman Problem can be expressed as a special case. The QAP has since been referred to as "one of the most difficult problems in the

NP-hard class" (Loiola et al., 2007), and has attracted much research attention due to its difficulty making it an attractive benchmark for newly developed generic optimisation algorithms and metaheuristics, due to its theoretical importance, and due to its many practical applications. Loiola et al. (2007) gives a detailed survey of approaches and applications of the QAP, referencing 365 papers published between its introduction in 1957 up until 2007. We discuss some more recent applications and solution approaches in Section 1.1 and Section 1.2 respectively.

1.1 Applications

Mention (Feng and Su, 2015) as a recent example of hospital layout.

1.2 Existing approaches

define steepest descent search

Meneses and Inostroza-Ponta (2011) investigated the effects of different memory schemas on the performance of the memetic algorithm for the QAP described in Inostroza-Ponta et al. (2008). Meneses and Inostroza-Ponta uses a structured population based on a ternary tree introduced by Inostroza-Ponta et al. (2008). This structure is considered to be a long-term memory schema, as it is used to guide the interaction of individuals in the algorithm. Four other memory schemas are considered in their comparisons; Fixed Tabu List (Glover, 1989), Variable Tabu List (VTL), Reduced Search (RS), and greedy Steepest Descent search (SD). Steepest Descent search repeatedly improves a solution by moving to the best solution in its 2-swap neighbourhood until a local maximum is found. The greedy variant used by Meneses and Inostroza-Ponta simply moves to the first improving solution rather than finding the best neighbouring solution. Reduced Search is an SD search where the equal positions among the parents of the initial individual are initially tabu. The eight combinations of memory schemas are evaluated by running them on 30 selected instances of the QAP taken from the literature. Their best classifier combined VTL, RS, and SD and the structured population to achieve an average deviation from the best known results of 0.133%, and their statistical tests indicated that combining memory schemas generally led to better performance of their algorithm.

A mutation operator is used that produces a new permutation with a Hamming distance of μ from its input. To do this it randomly generates a number of sequences of μ transpositions, ensuring that their composition has the correct Hamming distance from μ, and ranks each sequence according to a *disruptiveness* metric. The most disruptive mutation is chosen to produce the result permutation. The number of sequences to generate is a parameter of the algorithm. The parameter μ has an initial value $\mu_{\min}$ and is

Uses three intensification strategies. The first causes the tabu list is ignored once every I_1 iterations, allowing the best neighbouring solution to be accepted without a tabu test. The second strategy is to run a special 'fast' SD search to improve the current solution whenever a neighbouring solution is found that improves upon the current solution. This SD search is only run if it has already been run within the last I_2 iterations. The third strategy is to decrease all values in the tabu list by half whenever a new local optimum is encountered.

Make this section smoother.

Harris et al. (2015) introduces a memetic algorithm to solve the QAP. Their method represents candidate solutions by simply storing the image of the permutations in an array.

The ternary tree memory schema previously introduced in Inostroza-Ponta et al. (2008) is used to structure the population and to guide the other stages of the algorithm. The population is divided into 13 nodes which form a complete ternary tree. Each node contains 6 individuals, a *current* individual, and five *pocket* individuals. This population structure is illustrated in Figure 1.

Their algorithm uses the CX recombination operator used by Merz and Freisleben (2000) as a crossover operator, followed by improvement using tabu search to generate new individuals. The CX operator notably does not perform any implicit mutation, in the sense that every plant assigned to a location i in the child permutation is also assigned to location i in at least one of its parents. In fact, Harris et al. do not use random mutation in their algorithm, relying instead on a randomly initialised population, and a method of randomly restarting the population whenever the current best solution has not changed for a given number of iterations. The population is restarted by removing all but the best individual before adding a randomly generated individual (improved using tabu search) to each node.

Each iteration of the algorithm generates a new individual in each node using a crossover operator. One of the parents of the crossover is always chosen randomly from the source node, and the other parent is chosen randomly from the same node if the population diversity is high, or from a random node from a different subpopulation of the ternary tree if the population's diversity is low. This strategy is intended to prevent rapid diversity loss. Four simple rules are used to determine whether and how the new individual should be incorporated into the node's set of pocket solutions. These rules are also designed to reduce the loss of diversity.

The algorithm presented by Harris et al. was tested on the complete QAPLIB set of problems and achieved an average gap of 0.049% below the best known solutions when run with a time limit of 5 minutes. The average time before the best solution was found was 3.93 seconds.

Benlic and Hao (2015) proposed a new memetic algorithm for the QAP called BMA, which uses Breakout Local Search (BLS) as its local search procedure, the CX crossover operator used by Merz and Freisleben (2000), a fitness-based population replacement strategy, and an adaptive mutation mechanism.

BMA first randomly generates an initial population P, before improving each individual using BLS. BLS is a local search procedure that improves a solution by first performing a steepest descent search using the full 2-swap neighbourhood, before applying L *peturbation moves* intended to diversify the search. These moves amount to either

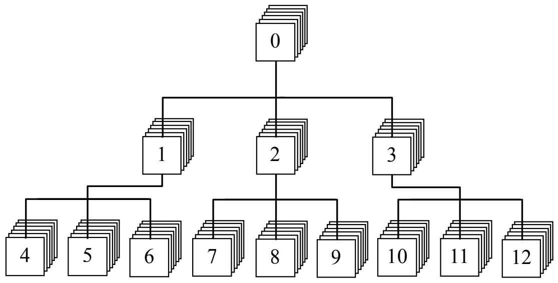

Figure 1: The ternary tree population introduced in Inostroza-Ponta et al. (2008), consisting of 13 nodes each containing a single *current* individual, and five *pocket* individuals.

L iterations of tabu search, or the application of L random transpositions to the current solution permutation. The choice of random or tabu search moves is probabilistic, and is based on the number of consecutive non-improving solutions found by the SD search. L is set to an initial value L_0 plus the number of consecutive visits the SD search has made to the immediate previous local optimum. The BLS search consists of a fixed number of iterations of SD search followed by perturbation moves, and then a final SD search. Benlic and Hao use the BLS-QAP algorithm presented in Benlic and Hao (2013).

Then in each generation, a subset $p \subset P$ of $|p|$ parent individuals are chosen from the population, by repeatedly randomly selecting λ individuals from P and adding the best of them into p if it is not already present in p. Two random parents from the set p are selected for crossover using the CX crossover operator used by Merz and Freisleben (2000) to produce a new individual. The new individual is improved using BLS, before being subjected to a replacement strategy to determine whether it should be included into the population. The replacement strategy simply replaces the worst solution in the population with the new solution as long as the new solution is better than the worst solution, and the new solution is not identical to any existing solution in the population.

If the best solution found is not improved after v generations, then a mutation procedure is performed on each individual in the population. The mutation operation produces a random permutation with a Hamming distance of μ from its input, where μ is chosen to be proportional to the number of iterations since the last improvement to the best solution less v.

BMA was able to achieve an average gap of 0.016% below the best known solutions among all problems in QAPLIB within a time limit of 30 minutes, and was able to find the best known solution within 2.5 minutes for all but the 21 ‘hardest’ instances.

Helal et al. (2015) presents variations of an algorithm called TBH-PSO, which combines Hierarchical Particle Swarm Optimisation (HPSO) and Tabu Local Search to solve to QAP.

HPSO is a metaheuristic that maintains a population of ‘particles’ each comprised of a state and a velocity. Each particle’s state represents a candidate solution to the problem being solved, and each velocity influences the way that the particle moves through the solution space. The implementation of HPSO presented by Helal et al. arranges the particles in a ternary tree structure similar to that of Harris et al. (2015), except that each node in the tree consists of a single particle. In each iteration of the algorithm, the velocity of each particle is updated using a function of its state, the state of its parent, the best states ever attained by itself and its parent, and its current depth in the hierarchy. The velocity components are then clamped between constant minimum and maximum values, and the state is recalculated from the velocity. Following the state-velocity update the value of each particle is compared to its immediate children, and if the value of the best child is better than the current particle's value, the two particles positions in the tree are swapped.

TBH-PSO applies PSO to the QAP by representing the velocity of each particle as an $n \times n$ matrix $[v_i]$ in which each v_{iyz} represents the tendency of particle i to assign plant y to location z. The algorithm defines a stochastic procedure to generate a particle's state from its velocity, which is run whenever a particle's velocity changes.

Tabu Search is used to improve the root particle in the hierarchy after each iteration of PSO.

Helal et al. ran TBH-PSO on 31 selected instances from QAPLIB, and achieved an average deviation from the best known solutions of 0.0919%, with an average running time of 44 seconds.

References

Benlic, U. and Hao, J.-K. (2013). Breakout local search for the quadratic assignment problem. *Applied Mathematics and Computation ()*, 219(9):4800–4815.

Benlic, U. and Hao, J.-K. (2015). Memetic search for the quadratic assignment problem. *Expert Syst. Appl. ()*, 42(1):584–595.

Feng, X. and Su, Q. (2015). An applied case of quadratic assignment problem in hospital department layout. In *2015 12th International Conference on Service Systems and Service Management (ICSSSM)*, pages 1–5. IEEE.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, 1(3):190–206.

Harris, M., Berretta, R., Inostroza-Ponta, M., and Moscato, P. (2015). A Memetic Algorithm for the Quadratic Assignment Problem with parallel local search. *CEC*, pages 838–845.

Helal, A. M., Jawdat, E., and Abdelbar, A. M. (2015). An improved Particle Swarm Optimization/Tabu search approach to the Quadratic Assignment Problem. *CEC*, pages 220–226.

Inostroza-Ponta, M., of Newcastle (N.S.W.). School of Electrical Engineering, U., and Science, C. (2008). An integrated and scalable approach based on combinatorial optimization techniques for the analysis of microarray data. School of Electrical Engineering and Computer Science.

Koopmans, T. C. and Beckmann, M. (1957). Assignment Problems and the Location of Economic Activities. *Econometrica*, 25(1):53.

Loiola, E. M., de Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., and Querido, T. (2007). A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176(2):657–690.

Meneses, H. and Inostroza-Ponta, M. (2011). Evaluating Memory Schemas in a Memetic Algorithm for the Quadratic Assignment Problem. In *2011 30th International Conference of the Chilean Computer Science Society (SCCC 2011)*, pages 14–18. IEEE.

Merz, P. and Freisleben, B. (2000). Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. *IEEE Trans. Evolutionary Computation ()*, 4(4):337–352.