European Journal of Operational Research 209 (2011) 215–218

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

An efficient implementation of the robust tabu search heuristic for sparse quadratic assignment problems

G. Paul *

Center for Polymer Studies and Department of Physics, Boston University, Boston, MA 02215, USA

ARTICLE INFO

Article history:
Received 12 January 2010
Accepted 6 September 2010
Available online 15 September 2010

Keywords:
Combinatorial optimization
Computing science
Heuristics
Tabu search

ABSTRACT

We propose and develop an efficient implementation of the robust tabu search heuristic for sparse quadratic assignment problems. The traditional implementation of the heuristic applicable to all quadratic assignment problems is of $O(N^2)$ complexity per iteration for problems of size N. Using multiple priority queues to determine the next best move instead of scanning all possible moves, and using adjacency lists to minimize the operations needed to determine the cost of moves, we reduce the asymptotic ($N \to \infty$) complexity per iteration to $O(N \log N)$. For practical sized problems, the complexity is $O(N)$.

1. Introduction

The quadratic assignment problem (QAP) is a combinatorial optimization problem first introduced by Koopmans and Beckmann (1957). It is NP-hard and is considered to be one of the most difficult problems to be solved optimally. The problem was defined in the following context: A set of N facilities are to be located at N locations. The distance between locations i and j is $D_{i,j}$ and the quantity of materials which flow between facilities i and j is $F_{i,j}$. The problem is to assign to each location a single facility so as to minimize the cost

$$C = \sum_{i=1}^{N} \sum_{j=1}^{N} F_{i,j} D_{p(i),p(j)}, \tag{1}$$

where $p(i)$ represents the location of facility i. It will be helpful to think of the N facilities and the matrix of flows between them in graph theoretic terms as a graph of N nodes and weighted edges, respectively.

There is an extensive literature which addresses the QAP and is reviewed in Pardalos et al. (1994), Cela (1998), Anstreicher (2003), Loiola et al. (2007), and James et al. (2009a). With the exception of specially constructed cases, optimal algorithms have solved only relatively small instances with $N \leqslant 36$. Various heuristic approaches have been developed and applied to problems typically of size $N \approx 100$ or less. One of the most successful heuristics to date for large instances is *robust tabu search*, RTS, Taillard (1991). The use of tabu search for the quadratic assignment problem has been studied extensively (Drezner, 2005a, Hasegawa et al., 2000, James et al., 2009a,b, McLoughlin and Cedeno, 2005, Misevicius, 2007, Misevicius and Ostreika, 2007, Skorinkapov, 1994, and Wang, 2007). Some of the best available algorithms for the solution of the QAP are the hybrid genetic algorithms that use tabu search as an improvement mechanism. (See Drezner, 2002, 2003, 2005b,c, 2008, Drezner and Drezner, 2006, Drezner and Marcoulides, 2006, 2009).

Here we will consider the robust tabu heuristic applied to *sparse* QAP instances. That is, the number of non-zero entries in the either the flow matrix and/or the distance matrix is of $O(N)$ as opposed to $O(N^2)$. Without loss of generality we will assume the flow matrix is sparse. Many real world problems are sparse. In fact, this work was motivated by the study of random regular sparse graphs. These graphs are very robust to partitioning and collapse due to removal of nodes or edges. We are interested in the problem of determining how to assign the nodes of such a graph to locations in a metric space such that the total edge length of the graph is minimized; this problem maps directly to a quadratic assignment problem.

There has been some previous work on sparse quadratic assignment problems. Milis and Magirou (1995) developed a Lagrangian-relaxation lower-bound algorithm for sparse problems and Pardalos et al. (1997) developed a version of their GRASP heuristic for sparse problems. However, to the best of our knowledge, an efficient implementation of the robust tabu heuristic for sparse QAP instances has not been proposed.

2. Background-the tabu heuristic

The tabu heuristic for the quadratic assignment problem consists of repeatedly swapping locations of two nodes. A single iteration of the heuristic consists of

* Tel.: +1 781 861 6279.
E-mail address: gerryp@bu.edu

0377-2217/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2010.09.009

(a) Determining the move which most decreases the total cost. Under certain conditions (see Section 4), if a move which lowers the cost is not available, a move which raises the cost is made. So that cycles of the same moves are avoided, the same move is forbidden (*taboo*) until a specified later iteration; we call this later iteration the *eligible iteration* for a given move. This eligible iteration is traditionally stored in a *tabu list* or *tabu table*.
(b) Making the move.
(c) Recalculating the new cost of all moves.

The process is repeated for a specified number of iterations. Traditional implementations of robust tabu search require $O(N^2)$ operations per iteration. The complexity of $O(N^2)$ is achieved by maintaining a matrix of the costs $\Delta(p,u,v)$ of swapping u and v for all u and v, given a current assignment p. The complexity of updating $\Delta(p,u,v)$ is described in the Appendix A.

The complexity of the each step above is as follows:

(a) $O(N^2)$-all possible $N(N-1)/2$ moves are considered. The cost of each move is retrieved from $\Delta(p,r,s)$
(b) $O(1)$-the locations of the two swapped nodes are simply transposed.
(c) $O(N^2)$-based on the following observations of Taillard (1991):
  (i) the cost of moves which do not involve the two nodes in the previous move can be calculated in time $O(1)$ (see Appendix A). There are $O(N^2)$ of these moves.
  (ii) The cost of moves which do involve the two nodes in the previous move must be calculated from scratch. There are $O(N)$ of these moves and the complexity of calculating each is $O(N)$ (see Appendix A).

3. Approach

To reduce the complexity of step (a), instead of scanning all possible moves, we use multiple *priority queues* (PQs) to determine the best move. A priority queue is a data structure for maintaining a set of elements each of which has an associated value (priority) (see Cormen et al., 2009). A PQ supports the following operations:

- Insert an item
- Remove an item
- Return the item with the highest value

Priority queues are used to efficiently find an item with the highest value without searching through all of the items.

The maximum complexity of PQ operations is $O(\log N)$. We will see below that there will be $O(N)$ insertions and deletions in the PQs for each iteration so the asymptotic complexity of this step is reduced to $O(N \log N)$. Furthermore, we will show that for problems of any practical size, PQ operations are not the determinant of total complexity.

The complexity to recalculate the cost of moves in step (c), can be reduced to $O(N)$ as follows:

- As in the traditional robust tabu implementation, the cost of moves which do not involve the two nodes in the previous move can be calculated in time $O(1)$. On average, there are $2\langle k\rangle$ nodes which are connected to the two nodes in the previous moves, where $\langle k\rangle$ is the average degree (average number of nodes adjacent to a given node) of the graph corresponding to the flow matrix. For each of these $2\langle k\rangle$ nodes we must calculate the cost of $N-1$ possible moves. Thus, the cost is $O(\langle k\rangle N)$.
- The cost of moves which do involve the two nodes in the previous move must be calculated from scratch. There are $O(N)$ of these moves and the complexity of calculating each is $O(\langle k\rangle)$ since the cost of a node, u, being in a specific location depends only on the on-average k nodes adjacent to u.

Thus the complexity of step (c) is reduced to $O(N)$.

4. Implementation

To describe our implementation, we must first describe the rules for determining the next move of Taillard's robust tabu heuristic (Taillard, 1991). The following definitions for the possible *state* of a potential move are useful:

(i) If the current iteration is less than or equal to the eligible iteration, the move is *ineligible*.
(ii) If the current iteration is greater than the eligible iteration, the move is *authorized*.
(iii) If the current iteration minus an *aspiration constant* is greater than the eligible iteration the move is *aspired*.

The rules for determining the next move can then be stated as (Taillard, 1991):

(1) If a move which decreases the lowest total cost found so far is available, the move which most decreases this total cost is chosen, independent of whether the move is ineligible, authorized or aspired.
(2) If no move meets criterion (1), the aspired move, if one is available, which most decreases the current total cost is chosen.
(3) If no moves meet criteria (1) or (2), the lowest cost authorized move is chosen.

To implement these rules for sparse problems, we use two types of PQs: *delta* PQs which contain the cost delta for a given move and *tabu* PQs which contain entries ordered by the eligible iteration for the move. The tabu PQs control the change of state of a move. The delta PQs determine the lowest cost move in each state. Five PQs are used:

- ineligible tabu PQ – This PQ contains moves, ordered by eligible iteration, which are in the ineligible state. This PQ allows us to efficiently determine when the state of a move can be changed to authorized.
- authorized tabu PQ – This PQ contains moves, ordered by eligible iteration, which are in the authorized state. This PQ allows us to efficiently determine when the state of a move can be changed to aspired.
- ineligible delta PQ – This PQ contains moves, ordered by the cost of the move, which are in the ineligible state. This PQ together with the two other delta PQs allows for efficient determination of the overall lowest cost move as required by rule 1.
- aspired delta PQ – This PQ contains moves, ordered by the cost of the move, which are in the aspired state. This PQ allows for efficient determination of the lowest cost aspired move as required by rule 2.
- authorized delta PQ – This PQ contains moves, ordered by the cost of the move, which are in the authorized state. This PQ allows us to determine the lowest cost authorized move as needed by rule 3.

As illustrated in Fig. 1, moves are inserted and removed in the PQs under the following circumstances:

- At initialization all moves are inserted into the ineligible PQs.
- At the beginning of each iteration, any moves on the ineligible PQs which become authorized, because the iteration has