

The Variable Neighborhood Tabu Search for Location/Allocation problems

José A. Moreno Pérez, J. Marcos Moreno Vega,
Inmaculada Rodríguez Martín.

D.E.I.O.C., Universidad de La Laguna. 38271 La Laguna, Tenerife, Spain.

{jamoreno, jmmoreno, irguez}@ull.es

Abstract

The Variable Neighborhood Tabu Search consists of an iterative combination of a local search and a shake procedure that uses tabu tools. When the local search stops at a nontabu local minimum, the shake procedure starts a new local search at a variable distance. The method is tested with the Median Cycle Location Problem consisting of finding the shortest cycle passing through a depot with an upper bound for the sum of the distance from the users to the cycle. For big instances from the TSPLIB library the procedure got in few seconds the best results known until the moment.

Key Words: Optimization, Location, Heuristics.

Introduction

A central topic in Nonlinear Global Optimization are the Metaheuristic Search Methods for Combinatorial Optimization Problems. A great variety of Combinatorial Optimization Problems that arise in Logistic Planning are nonlinear in nature. Among these problems the Location/Allocation problem plays a central role. In a Location/Allocation problem, a set of points is selected to hold facilities that must serve to all the users. In these situations there are two typical functions to be minimized: a measure of the connectivity structure among the opened facilities, and the cost of allocating the users to the facilities. These two functions play different roles in the optimization models. Also the evaluation of them could

mean different computations or simulations. The connectivity structure usually implies a unique investment, while the assignment or allocation of users to facilities implies diary costs. The connectivity cost is usually assumed by the server and the allocation costs by the users. In the cycle location problems the connectivity structure is a cycle passing through all the facility points and including a depot. Thus, a cycle location problem consists of selecting a set of locations for the facility, the cycle passing through the facility locations, and the allocation of the users to the locations. The Median Cycle Problem consists of minimizing the length of the cycle with an upper bound on the sum of the distances from the users to the cycle.

Besides of exact methods for modeling and solving the problems, heuristic search methods provide an intelligent approach to deal with the models. The Variable Neighborhood Tabu Search is a new metaheuristic, inspired in Variable Neighborhood Search and Tabu Search, that combines the use of tabu tools with local searches and shake procedures. The local search applies an improving move until no such move exists. The shake procedure applies a number of random moves. Since the shake could provide an infeasible solution, the local search considers as possible moves those providing a feasible solution or reducing the infeasibility. The standard moves for the Location/Allocation problems are the add, drop and add/drop moves. The application for solving generic Location/Allocation problems is analyzed. Some computational results on TSPLIB instances are reported.

The VNTS Heuristic

Local Search methods for global optimization proceed by performing a sequence of local changes in an initial solution which improve each time the value of the objective function until a local optimum is found. That is, at each iteration an improved solution X' in the neighborhood $\mathcal{N}(X)$ of the current solution X is obtained, until no further improvement is found. In recent years, several metaheuristics have been proposed which extend in various ways this scheme and avoid being trapped in a local optimum. The most famous of these metaheuristics are Simulated Annealing ([10]), Genetic Algorithms ([8]) and Tabu Search ([5] and [6]).

To avoid termination at a local minimum, the metaheuristics allow nonimproving moves. However, this presents the risk of cycling. In Tabu Search the use of a short term memory helps to forbid moves that might lead to recently visited solutions. Variable Neighborhood

Search ([13]) is a relatively new technique whose originality consists of trying to escape from local optima by changing the neighborhood structure. In its basic form, VNS explores a set of neighborhoods of the current solution, makes a local search from a neighbor solution to a local optimum, and moves to it only if there has been an improvement.

The metaheuristic method we propose, the Variable Neighborhood Tabu Search (VNTS), was originally inspired in VNS. However, our conception of the neighborhoods is somehow different. In VNTS the different neighborhoods used are nested and are defined from an original one \mathcal{N} in the following way:

$$\mathcal{N}_1(X) := \mathcal{N}(X), \mathcal{N}_2(X) := \bigcup_{Y \in \mathcal{N}(X)} \mathcal{N}_1(Y), \dots, \mathcal{N}_k(X) := \bigcup_{Y \in \mathcal{N}(X)} \mathcal{N}_{k-1}(Y).$$

That is, $\mathcal{N}_i(X)$ are the neighbors of X at distance from 1 to i .

VNTS comprises the following steps:

Initialization

1. Select a neighborhood structure \mathcal{N} .
2. Choose a stopping condition.
3. Find an initial solution X . Set $Best_X := X$.

Iterations. Repeat the following until the stopping condition is met:

1. Local Search: Apply a greedy tabu local search based on \mathcal{N} until a local minimum X' is found.
2. Improvement: If X' is better than $Best_X$, do $Best_X := X'$.
3. Shaking: Generate at random a solution X'' in $\mathcal{N}_s(X')$. Set $X := X''$.

Tabu search tools are incorporated to both local search and shaking procedures to modulate the intensification and diversification of the search. The local search tests all the nontabu moves, and that providing the best feasible solution is made. The shake consists of applying a number s of random moves (feasible or not). This number of random moves is the size of the shake. The shake could provide an infeasible solution without feasible neighbor. In those cases the local search is allowed to make the best nontabu move among those that reduce the infeasibility. Therefore, the local search consists of choosing the best feasible nontabu move, if there is some, or the best nontabu move that reduces the infeasibility. As

stopping criteria usual conditions can be applied: the total CPU time, the number of local searches, the number of moves, the relative number of moves without improving the best solution found.

To show the wide applicability of the VNTS metaheuristic consider the problems where each solution is given by a selection of a set of elements from an universe. Many of relevant combinatorial optimization problems can be formulated in this way. The standard moves for those problems are add, drop and interchange moves. These moves consist of adding a new member to the solution, dropping a member from the solution, or interchanging a member of the solution for another one out of the solution. The selection can be subject to some constraints. In those cases, the moves that provide a feasible solution are called feasible moves. The tabu tool proposed is a list with the last elements that have been dropped from the solution. The elements that are in the tabu list are not allowed to go immediately inside the solution again. This tabu tool is more powerful than the usual list of tabu moves, since with a short list of tabu elements many moves are forbidden.

Application of VNTS to Location/Allocation problems

In this section we study with more detail the application of VNTS to a general Location-Allocation problem. In the classical p -median and p -center problems the measure of the connection structure is only the number of facilities, that is fixed to p , and the allocation cost to be minimized is respectively the sum and the maximum of the distances between the facilities and the users assigned to them. The Uncapacitated Facility Location Problem consists of minimizing the sum of costs of the opened facilities and the allocation cost given by the sum of the distances between the users and the nearest facility. In other classical optimization problems, the connection structure is a cycle or a tree and the measure is its length. Let us mention, for example, the Shortest Covering Path Problem ([2]), the Maximal Direct Covering Tree Problem ([9]) and the Covering Tour Problem ([4]).

The allocation cost can also include a measure of a structure connecting the users with the facilities. These connection structure usually have tree, cycle or star shape for each facility. With the star structure each user is connected by a path to a facility, and the allocation cost is the sum of the distances between the users and their nearest facilities. One of these problems is the Median Cycle Problem (MCP), in which the connection structure is a cycle visiting a fixed depot and the allocation structure is of star type. The MCP consists

of minimizing the length of a tour passing through the depot while imposing an upper bound on the sum of the distances from all the users to the cycle. A modified version of this problem consists of minimizing the sum of connection and allocation costs of the cycle.

The MCP is also a Location/Allocation problem where the solution consists of selecting a set of points S to hold the facilities. However, given the selected set of points S , the cycle passing through the points of S and the depot and the allocation of the users to the opened facilities have to be chosen. Thus, the solution is evaluated by two functions: the length of the shortest cycle, given by $Len(S)$, and the optimal allocation cost, given by $Cos(S)$.

The usual moves considered in this kind of problems are the add, drop and interchange moves. Given a solution S and an element x_j not in the solution, an add move consists of adding x_j to S . A drop move consists of eliminating an element x_i from S . Given a solution S , an element x_i in the solution and an element x_j not in the solution, an interchange or add/drop move consists of dropping x_i from S and adding x_j to S .

If any constraint is imposed, not every move is feasible. The moves that provide a feasible solution are called feasible moves. If the constraint consists in an upper bound on a function of the solution then the moves that reduce the infeasibility can be used to get feasible solutions.

The selection of a solution coding that provides an efficient way of implementing the moves and evaluating the solutions is essential for the success of any search method. For cycle location problems, the evaluation of a solution means to solve another optimization problem. The evaluation of the length of the shortest cycle requires to solve a TSP. The efficiency when evaluating the modification produced in the objective function by a move is very important, because a great deal of moves are tested. Therefore, having an efficient procedure to (may be heuristically) evaluate the modifications of the solution is very useful. The information given by the current solution can be used to design an efficient way of evaluating a move, by considering only the local modifications that it would produce. So, when a solution is modified by adding and/or dropping a point, the length of the new solution is evaluated by extracting the leaving point and/or looking for the best position to insert the new one. In addition, to calculate the cost of the new solution we need to know the new optimal allocation of the users. Usually we only need to know if the entering point can improve the allocation of the users and if the users allocated to the leaving point need to be reallocated.

Constructive heuristics are generally used to find an initial solution. Standard constructive heuristics consists of a series of add moves starting from an empty solution. During the building process the length of the candidate solution increases while its allocation cost decreases. The procedure stops when the solution obtained is feasible. The more common strategies for adding elements to the solution are to use simple, random, and good add moves. The good properties that a constructive method should have in order to be used in MultiStart ([1]) or GRASP ([3]) metaheuristic are:

Efficacy: The degree of optimality of the solution proposed must be high.

Efficiency: The amount of resources (time) used must be small.

Randomness: The ability to cover the whole search space.

The VNTS for MCP

In order to apply the moves that modify the current solution we must take into account the following. The insertion of a new point x_j in the cycle implies to know in which position it is to be inserted. The assignment cost of the new solution does not depend on the insertion position. So we will always insert the new points in the best possible position, i.e., the position that provides a shortest cycle. The decrement in the cost depends only on the point inserted, and not on the position in the cycle.

The computation of the cost of the new solution after each adding move can be simplified by storing the individual allocation costs in a vector $Cos1$. The individual allocation cost of a point is the distance to the cycle. The new individual allocation costs obtained when the point x_j is inserted can be efficiently calculated. When deleting a point from the cycle the individual allocation costs have to be updated only for the points assigned to the going out point. The computation of the cost of the new solution obtained when a point is removed from the cycle is improved by using the second best allocation $Cos2$. When removing x_k , the first and second individual allocation costs have to be recomputed only for the points having x_k as is the first or second best allocation. The second best allocation cost is a tool similar to the one used for the VNDS in [7].

The interchange of two points consists of a combination of an add and a drop moves. A point is removed from the cycle, and a point is inserted in the best possible position in the cycle. However, using the values in $Cos1$ and $Cos2$, an improved procedure can be applied. We first test if the going in point is the best or the second best allocation for the users. Then

we need to update the individual costs of the users assigned to the dropped point.

We use a tabu list to improve the efficiency of the search by avoiding moves leading to recently visited solutions. In order to do so we put in a FIFO tabu list the points dropped from the cycle. The tabu list is kept at the end of the array that codes the solution (see [14]). Only points out of the tabu list are tested for being included in the cycle. So, a leaving point will be out of the cycle in at least t iterations (the size of the tabu list). Note that this tabu tool is more powerful than the usual list of tabu moves, since with a short list of tabu elements many moves are forbidden.

The shake procedure (used in the VNS heuristic in [13]) allows to escape from a local minimum without destroying completely the good properties of the current solution. Given a size s for the shake procedure, we choose s times two points at random, x_i and x_j . If the point x_i is in the cycle and x_j is outside the cycle we do the corresponding add/drop move, if both points are in the cycle we drop x_i , and if both points are outside the cycle we add x_j . The going-in point x_j is always chosen taking into account the tabu list, and it is inserted in the best possible position. The going-out point x_i , if dropped, goes to the tabu list, that is updated.

Given a solution, the greedy local search is implemented by choosing each time the best possible move among all the add, drop or add/drop moves. The best possible move is that providing the smallest increment in the length among all those having a feasible cost or a less infeasible cost (the latest only if there are not feasible moves). This is done to allow the process to move through feasibility if this property has been lost when making the shake.

After the set of points in the solution has been determined this way, the length of the cycle is improved by performing classical edge interchange moves such as 2-opt, 3-opt, Lin and Kernighan interchange, or some combination of them (see [12]).

The size t of the tabu list is initially 0. Each time a point is removed from the cycle it goes to the tabu list and t augments one unit. When t reaches a given bound (ten per cent of n in our case) it is fixed, and the tabu list starts to work as a FIFO list. Just before performing the shake, the tabu list is emptied again.

The parameter s that controls the shake procedure is initialized to 2. Each time a local minimum is reached the shake procedure is applied s times. If the current local minimum is equal to the one found just before, s is augmented by one unit. If the current local minimum is better than the best known solution, s is set again to 2.

In our computational experiments we use as stopping criterion the number of iterations without improving the best solution found. Let it be the current iteration and it^* be the iteration where the best solution was found, then we stop if $it - it^* > Max_it$ for a given constant Max_it .

See [15] for more details on the VNTS implementation for solving the MCP.

Computational results

The application of the VNTS to the MCP described in the previous sections were implemented in C++ and run on a SUN Ultra-60 computer running at 300 MHz. It was tested on a set of instances from the TSPLIB (see [16]) involving up to 200 points and having EUC2D format. We use the Euclidean distances obtained from the data of the instance. We test several choices of the upper bound for the total allocation cost.

Table 1 shows some of the results obtained for MCP when the choice of the upper bound yields optimal cycle visiting approximately 25% of the total number of points.

The column headings are defined as follows:

Name: instance name as it appears in the TSPLIB.

Exact-t: computing time for the exact procedure, in hh:mm:ss format.

Heur-t: computing time for the VNTS, in seconds.

Error: the relative deviation of the heuristic solution from the optimal one.

The maximum number of iterations Max_it was set to 200. Five runs of the algorithm were performed for each instance, and we show in columns **Heur-t** and **Error** the minimum, average and maximum values obtained.

The error is computed as $100(opt - heur)/opt$, being opt the solution given by the branch-and-cut algorithm described in [11], and $heur$ the heuristic solution. The solution given by the branch-and-cut algorithm is not really the optimal one in all the cases, since there was a time limit of two hours in the execution time for each instance. The negative values that appear in the error columns correspond to instances in which the heuristic solution is better than the best solution found by the branch-and-cut algorithm in two hours.

Name	Exact-t	Heur-t			Error		
		min	aver	max	min	aver	max
eil101.tsp	0:13:46	16,99	20,48	27,29	0,60	0,60	0,60
pr107.tsp	1:21:46	13,10	16,97	23,29	0,00	0,15	0,52
pr136.tsp	1:57:07	43,41	49,63	59,51	0,25	0,29	0,36
pr144.tsp	+2:00:00	25,18	43,68	56,65	-2,75	-1,93	-0,72
kroA150.tsp	+2:00:00	56,03	62,59	68,71	-7,06	-4,97	-1,80
kroB150.tsp	+2:00:00	41,74	50,24	62,88	-1,85	-1,41	-1,07
pr152.tsp	+2:00:00	59,71	78,65	89,21	-6,73	-6,73	-6,71
rat195.tsp	+2:00:00	108,82	133,67	186,20	-6,31	-6,21	-6,15
kroA200.tsp	+2:00:00	89,08	139,95	177,77	-5,89	-5,75	-5,65
kroB200.tsp	+2:00:00	103,46	140,39	197,85	-5,58	-5,34	-4,78

Table 1: *Some VNTS results for MCP.*

Conclusions

The VNTS metaheuristic proposed in this paper is a very general tool that can be used to look for good solutions to optimization problems. We show how to apply it to any optimization problem consisting of the selection of an optimal set of items, considering the standard add, drop and add/drop moves. The consideration of constraints in the formulation does not increase the complexity of the procedures, as is shown for the location/allocation problems. The detailed implementation for the Median Cycle Problem got acceptable results in very few seconds when tested on the TSPLIB instances. Furthermore, for some instances, the VNTS got solutions better than the best known until the moment.

Future research will be focus on applying the metaheuristic to other combinatorial optimization problems.

Acknowledgments

This research has been partially supported by Gobierno de Canarias through the project PI1999/116, and by Universidad de La Laguna through the project 221/52/99.

References

- [1] Kenneth D. Boese, Andrew B. Kahng, Sudhakar Muddu; A new adaptive multi-start technique for combinatorial global optimizations; *Operations Research Letters*, 16, 101-113, 1994.
- [2] J. Current, J.L. Cohon, C.S. ReVelle; The shortest covering path problem: an application of locational constraints to network design; *Journal of Regional Science*, 24, 161-183, 1984.
- [3] Thomas A. Feo, Mauricio G.C. Resende; Greedy randomized adaptive search procedures; *Journal of Global Optimization*, 6, 109-133, 1995.
- [4] M. Gendreau, G. Laporte, F. Semet; The Covering Tour Problem; *Operations Research*, 45, 568-576, 1997.
- [5] F. Glover; Tabu Search - Part I; *ORSA Journal on Computing*, 1, 190-206, 1989.
- [6] F. Glover; Tabu Search - Part II; *ORSA Journal on Computing*, 2, 4-32, 1990.
- [7] P. Hansen, N. Mladenović, D. Pérez-Brito; Variable Neighborhood Decomposition Search; to appear in *Journal of Heuristics*, 1999.
- [8] J.H. Holland; *Adaptation in Natural and Artificial Systems*; The University of Michigan Press, Ann Arbor, Michigan, 1975.
- [9] V.A. Hutson, C.S. ReVelle; Maximal direct covering tree problems; *Transportation Science*, 23, 188-299, 1989.
- [10] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi; Optimization by simulated annealing; *Science*, 220, 671-666, 1983.
- [11] M. Labbé, G. Laporte, I. Rodríguez Martín, J.J. Salazar González; The Median Cycle Problem; Working paper, CRT, Université de Montreal, 1999.
- [12] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys; *The Traveling Salesman Problem*; John Wiley & sons, New York, 1985.
- [13] N. Mladenović, P. Hansen; Variable Neighborhood Search; *Computers Oper. Res.*, 24, 1097-1100, 1997.

- [14] N. Mladenović, J.A. Moreno-Pérez, J. Marcos Moreno-Vega; A Chain-Interchange Heuristic Method; *Yugoslav J. Oper. Res*, 6, 41-54, 1996.
- [15] José A. Moreno Pérez, J. Marcos Moreno-Vega, I. Rodríguez Martín; The Variable Neighborhood Tabu Search and its application to the Median Cycle Problem; to appear in *European Journal of Operational Research*, 2000.
- [16] TSPLIB; <http://www.iwr.uni-heidelberg.de/iwr/comopt/software/TSPLIB95/>, 1995.