Experimental Analysis with Variable Neighborhood Search for Discrete Optimization Problems

Marco Antonio Cruz-Chávez
Autonomous University of Morelos State, Mexico

Alina Martínez-Oropeza
Autonomous University of Morelos State, Mexico

Martín Martínez-Rangel
Autonomous University of Morelos State, Mexico

Pedro Moreno-Bernal
Autonomous University of Morelos State, Mexico

Federico Alonso-Pecina
Autonomous University of Morelos State, Mexico

Jazmín Yanel Juárez-Chávez
Autonomous University of Morelos State, Mexico

Mireya Flores-Pichardo
Autonomous University of Morelos State, Mexico

INTRODUCTION

Many problems within the Combinatorial Optimization area considered as NP-Complete problems (Papadimitriou & Steiglitz, 1998) require the use of heuristics to obtain high-quality solutions, due to their complexity and nature. One of the most frequently used is local search. It starts with an initial solution then applies a sequence of local changes in attempt to improve the value of the objective function and obtain the local optima. The types of movements applied in local searches are called neighborhood functions or neighborhood structures because they access to neighboring solutions and define the size of the neighborhood.

Sometimes the use of a sole heuristic is not enough to find good solutions for hard problems, because the solution space is very complex. In such cases, the variable neighborhood structures have demonstrated being efficient search methods for these problems.

This article presents two important optimization problems undertaken by different neighborhood structures, the Classical Symmetric Travelling Salesman Problem (CSTSP), and the Unrelated Parallel Machines Problem (UPMP).

An analysis of efficacy and efficiency is performed to identify the best neighborhood structure for each problem before its implementation within a heuristic or metaheuristic, such as Iterated Local Search, Tabu Search (Michaelewicz & Fogel, 2000), Memetic Algorithms (Cruz et al., 2008), Ant Colony (Alba, 2005), among others. The hybridization using heuristics and local search has been shown to be more efficient and effective in searching for solutions to NP problems than the heuristic alone.

There is some research in the literature about variable neighborhood search, as they are referred to in this article. Arajy y Abdullah (2010a) present a two-phase hybrid approach; the structure combines a variable neighborhood search (VNS) in the first-phase with an iterated local search in the second-phase, while always accepting the best solutions. The VNS involves 13 different neighborhood structures, which are randomly

DOI: 10.4018/978-1-4666-5888-2.ch403

selected during execution. Experimental results show the algorithm is competitive with other approaches in literature. For nine data sets, it obtained one improved and eight equal solutions.

In (Hansen & Mladenović, 1999) a VNS implemented in a local search algorithm is presented. In addition, some modifications of this approach are explained. Variants of VNS were tested in five problems: Travelling Salesman Problem (TSP), p-median Problem (PM), Multi-source Weber Problem (MW), Minimum Sum-of-squares Clustering Problem (MSSCC), and Bilinear Programming Problem with Bilinear Constraints (BBLP), showing competitive results especially for the PM and MW problems. In (Arajy & Abdullah, 2010b) a VNS is presented in the first-phase, with an iterated local search in the second-phase for the Attribute Reduction in Rough Set Theory. The approach was tested in over 13 well-known datasets. Experimental results demonstrate that the VNS was able to produce solutions competitive with the best techniques.

This research was motivated by the continuous need to find high-quality solutions for important combinatorial problems; such as TSP and UPMP, because in Metaheuristics, the local search is the most time-consuming procedure. Therefore, in this research a VNS is applied to an NP and an NP-Complete problem to observe its performance in different complexity problems, under the same conditions.

Experimental results show that the good performance of a VNS depends on the search space complexity of the problem. High-quality solutions are obtained for CSTSP (classified as NP-Complete problem), but poor-quality solutions for UPMP (a less complex NP problem). Although in most cases VNS produce very good solutions, the results show that this is not a rule for all discrete optimization problems. The contribution of this research is the finding that the performance of local search depends on the hardness of the problem. Contrary to what one might expect, the performance only is better for the NP-complete problem than for the NP problem, both studied in this article.

This article is organized as follows. Section two presents an introduction to the complexity problems undertaken, which are the CSTSP and the UPMP. Section four describes the neighborhood structures tested in this research. Section five details the proposed VNS and the experimental tests. Section six explains the statistical analysis of the obtained results for each structure. Finally, section seven presents conclusions.

BACKGROUND

This article presents four different neighborhood structures and a variable one applied to a model of the CSTSP and to the UPMP. Each of these problems has different complexities between them, being NP-complete the first one and NP de second one. A description of the characteristics of the undertaken problems is presented. All the neighborhood structures, including the variable one were applied on the CSTCP and the UPMP. Next, the tests run using the Solomon benchmarks are shown. Finally the conclusions of this article are presented.

COMPLEXITY PROBLEMS

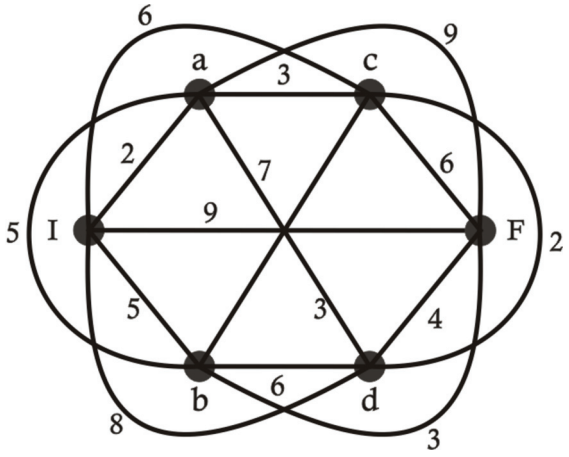
The Complexity Theory is an important part of computational sciences; it classifies the combinatorial problems taking into account their nature and complexity, according to time (steps required by an algorithm to solve a problem), and the space (amount of memory necessary to solve a problem) (Cortéz, 2004). Problems are classified as P, NP, and NP-Complete. P problems can be solved by exact methods in polynomial time. In the case of NP problems, there is no-known exact algorithm that solves them in all their instances, thus it is necessary to use heuristic methods to find high-quality solutions in a reasonable computational time. Problems classified as NP-Complete are the hardest problems.

Classical Symmetric Traveling Salesman Problem

The Classical Symmetric Traveling Salesman Problem (CSTSP) is a discrete optimization problem classified as NP-complete (Papadimitriou & Steiglitz, 1998, Ausiello, et al., 1999) due to its complexity and nature. The importance of this problem lies on its importance in several areas of industry, such as logistics and delivery, and is reflected in economic losses. This problem has been widely studied by means of many different optimization methods such as heuristics, including local search, which is used to find near-optimal solutions when solving larger instances in a reasonable computational time (Ausiello, et al., 1999).

The aim of the CSTSP is to minimize the total travel distance when visiting all the cities exactly

Figure 1. Undirected graph for the CSTSP



once and returning to the home town (Liu, et al., 2007). Considering the complexity of this problem, it is necessary to implement heuristics because of the size of the solution space as specified by $n!$, where n is the total of cities to visit. Therefore, the complexity increases as the input size grows.

This problem is generally modeled by a graph (Figure 1) $G=(V,E)$ of a finite set V of vertices, identifying the cities, and a finite multiset E of edges or distances between cities, that is, unordered pairs (i,j) of cities, where each city must be visited only once, trying to minimize the total travel distance. The tour has a beginning city and an ending one. Therefore, $E = \{(i,j): i, j \in V\}$ and c_{ij} is the cost (distance) associated with the edge (i,j).

The integer linear programming model (Fischetti, et al., 1995), shows the objective function as instruction 1, where the aim is to minimize the total travel cost based on the following constraints, which must be met to obtain feasible solutions. The set of constraints 2 specifies that only city i can be reached from city j; the set of constraints 3 specifies that only city j can be reached from city i and the last one ensures that all the cities have been visited.

Integer linear programming formulation for the CSTSP (Fischetti, et al., 1995):

$$Min \sum_{(i,j)\in E} C_{ij} X_{ij} \tag{1}$$

Subject to:

$$\sum_{\{C_j:(i,j)\in E\}} X_{ij} = 1 \qquad \forall i \in E \tag{2}$$

$$\sum_{\{C_i:(i,j)\in E\}} X_{ij} = 1 \qquad \forall i \in E \tag{3}$$

$$\sum_{\{(i,j)\in E,\ i\in S,\ j\in S\}} X_{ij} \leq |\, S \,| - 1 \quad \text{to} \quad S \subset V,\ 2 \leq |\, S \,| \leq |\, V \,| - 2 \quad \forall i \tag{4}$$

Unrelated Parallel Machines Problem

The Unrelated Parallel Machines Problem (UPMP) is a variant of the Classical Job Shop Scheduling Problem (CJSSP), where some constraints are relaxed to get a mapping of UPMP (Cruz, et al., 2010). This problem is classified as NP (Garey et al.,1976) because of its complexity. Consequently, it can be formulated by Binary Integer Linear Programming (BILP), which undertakes the problem by heuristics (Papadimitriou & Steiglitz, 1998).

The scheduling of a solution for UPMP is subject to the mathematical formulation (Pinedo, 2008), where expression 1 specifies the objective function, which minimizes the total completion time of processing all the jobs. The processing time kP_{ij} of job j depends on the machine and position where it was assigned, contributing to the value of the objective function. The set of constraints in expression 2 ensures that each job will be processed only once. Expression 3 guarantees that each position k processes at most one job j. The set of constraints 4 shows that the time of processing a certain job depends on the position of a certain machine, which involves a penalization according to the assigned position, where $x_{ijk}=1$ if the job was assigned and 0 otherwise.

Binary Integer Linear Programming Model for the UPMP (Pinedo, 2008):

$$\min f = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{n} kP_{ij} X_{ikj} \tag{5}$$

Subject to:

$$\sum_{i=1}^{m}\sum_{k=1}^{n} X_{ikj} = 1 \quad j = 1,\ldots,n \tag{6}$$

$$\sum_{j=1}^{n} X_{ikj} \leq 1 \quad i = 1,\ldots,m \quad k = 1,\ldots,n \tag{7}$$

$$X_{ikj} \in \{0,1\} \quad i = 1,\ldots,m \quad k = 1,\ldots,n \quad j = 1,\ldots,n \tag{8}$$

The features mentioned above show many similarities to the requirements of manufacturing systems currently used in industry. As demand increases, the machinery requirements grow, which is why the enterprises have to invest in new equipment, thus machines have different capacities.

Neighborhood Structures

Each optimization problem has a set of possible solutions, according to its classification within the Complexity Theory. The problem requires the use of techniques that allow better exploitation of the solution space, and consequently high-quality solutions. These types of techniques applied to local search are better-known as neighborhood structures, which defines the neighborhood and how to access neighboring solutions from the initial one. Therefore, any solution s' is directly reachable from s through a movement σ, thus $s' \in N(s)$. Accordingly, the neighborhood is defined as N: $S \rightarrow 2^s$. Consequently, the structure and the size of a neighborhood are defined by the type of movement applied in a local search.

A neighborhood structure is an iterative technique implemented with an algorithm, with the intent to improve the quality of solutions, according to the objective function of the problem. A critical aspect of designing some optimization algorithms is choosing an appropriate neighborhood structure.

In this research, four different neighborhood structures were applied to local search for the CSTSP, and the UPMP. The neighborhood structures are very similar for both problems, so the used for UPMP are explained as follows:

An Adjacent-Pair

Regardless of the problem undertaken, it is necessary to find an initial feasible solution s that is randomly generated at the beginning of the procedure. This solution is stored in a bi-dimensional array, so an array position *num* is randomly selected by the neighborhood structure. According to the features of the structure, that position is permutated with the position *num*+1, and a new neighboring solution s' (Lin & Kernighan, 1973 ; González & Bandala, 2009; Pacheco & Delgado, 2000) is obtained (Figure 2) for being evaluated. If s' improves the best solution so far, it will replace

Figure 2. Movement applied in a neighborhood structure of an adjacent-pair for the UPMP

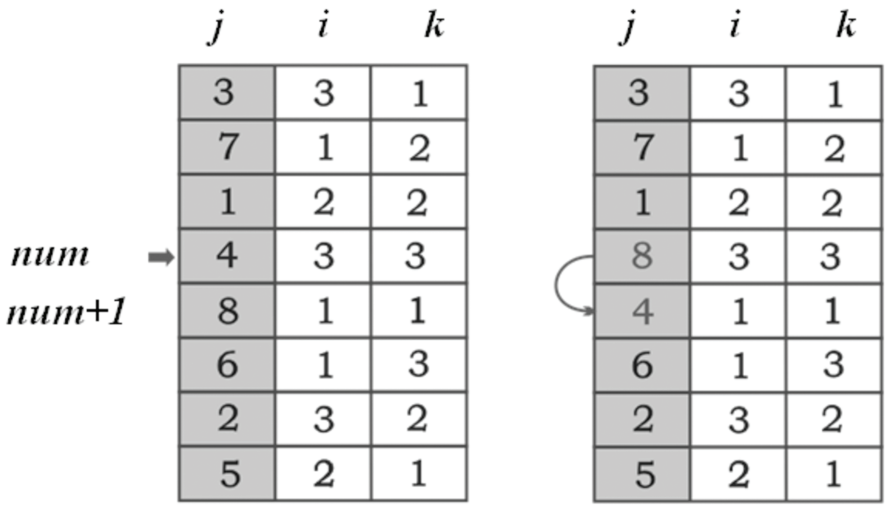


Figure 3. Movement applied in a neighborhood structure of a random-pair for the UPMP

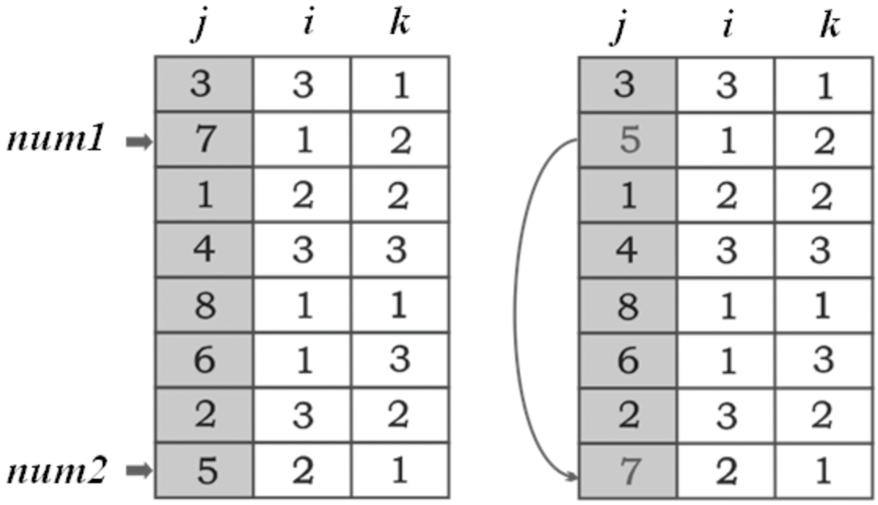
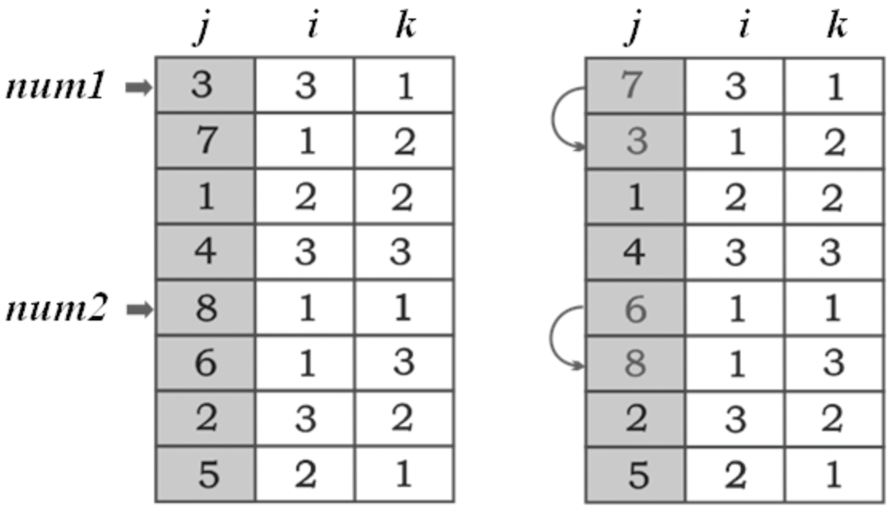


Figure 4. Movement applied in a neighborhood structure of two-adjacent pairs for the UPMP



by the new best solution. This procedure, called local search, is performed iteratively until the best solution cannot be improved.

In the case of CSTSP, the permutation is performed similarly; the difference lies on the representation of a solution, which is stored in a one-dimensional array. Unlike CSTSP, a solution to UPMP is composed of the job j, the machine i and position k, where j was processed.

A Random-Pair

Similarly, this neighborhood structure performs a single permutation (Figure 3) between two positions randomly selected *num*1 and *num*2 of a solution. (Papadimitriou & Steiglitz, 1998; González & Bandala, 2009; Kenneth, 1995). These positions must be different and not adjacent.

Two-Adjacent Pairs

This technique applies two permutations; contrary to the other two structures presented before. Therefore, it is necessary to select two random positions *num* and *num*1 in the current solution; which must be different and non-adjacent. Each movement is performed with its adjacent position (Lourenço, Martin, 2003); *num* permutes with *num*+1, and *num*1 with *num*1+1 (Figure 4).

Two-Random Pairs

This neighborhood structure increases in complexity, selecting four different and non-adjacent random positions (*num1, num2, num3,* and *num4*). After that, permutations take place (Figure 5) performing swaps in pairs (Kenneth, 1995; Martin et al., 1991; Martin et al., 1992).

When applying this neighborhood structure, the difference with the previous structures is greater between the initial solution and the neighboring one, because the neighborhood size is larger than in the other cases. As well as the computational complexity, this depends directly on the size of the neighborhood, and allows obtaining better local minima (Voudouris & Tsang, 1998).

Variable Neighborhood Search

A VNS is a technique that incorporates more than two different neighborhoods. In the literature, some researchers have developed approaches to exploit this concept, due to its improved performance as compared to basic neighborhood structures.

The development of the VNS in this article was based on the research of (Hansen & Mladenović, 1999; Cruz et al., 2010; Arajy & Abdullah, 2010a). This article presents a more straightforward VNS, which incorporates basic and low-complexity structures. This change attempts to optimize the time required to perform the local search.

The VNS is able to handle variable neighborhood sizes, due to the random interaction of the structures during the execution time, which improves the exploitation of the solution space. Neighborhood structures presented in this research were tested for the CSTSP and the UPMP, which are contrasting problems both in concept and in complexity. In both cases the neighborhood structures were applied to local search, although in the case of UPMP, the local search was implemented through the metaheuristic Ant Colony.

Movements performed in the VNS are shown below for the UPMP (Figure 6), being very similar its application in CSTSP. The only difference is that the neighborhood structures applied to UPMP were

Figure 5. Movement applied in a neighborhood structure of two-random pairs for the UPMP

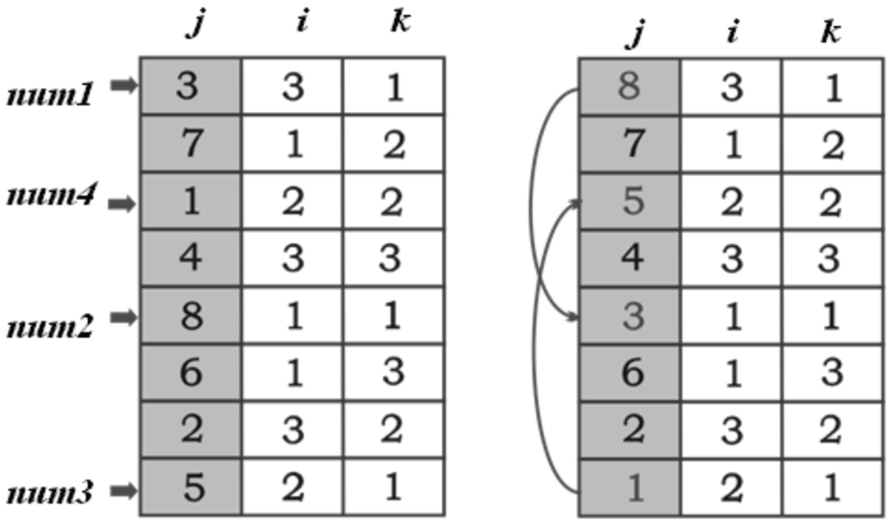
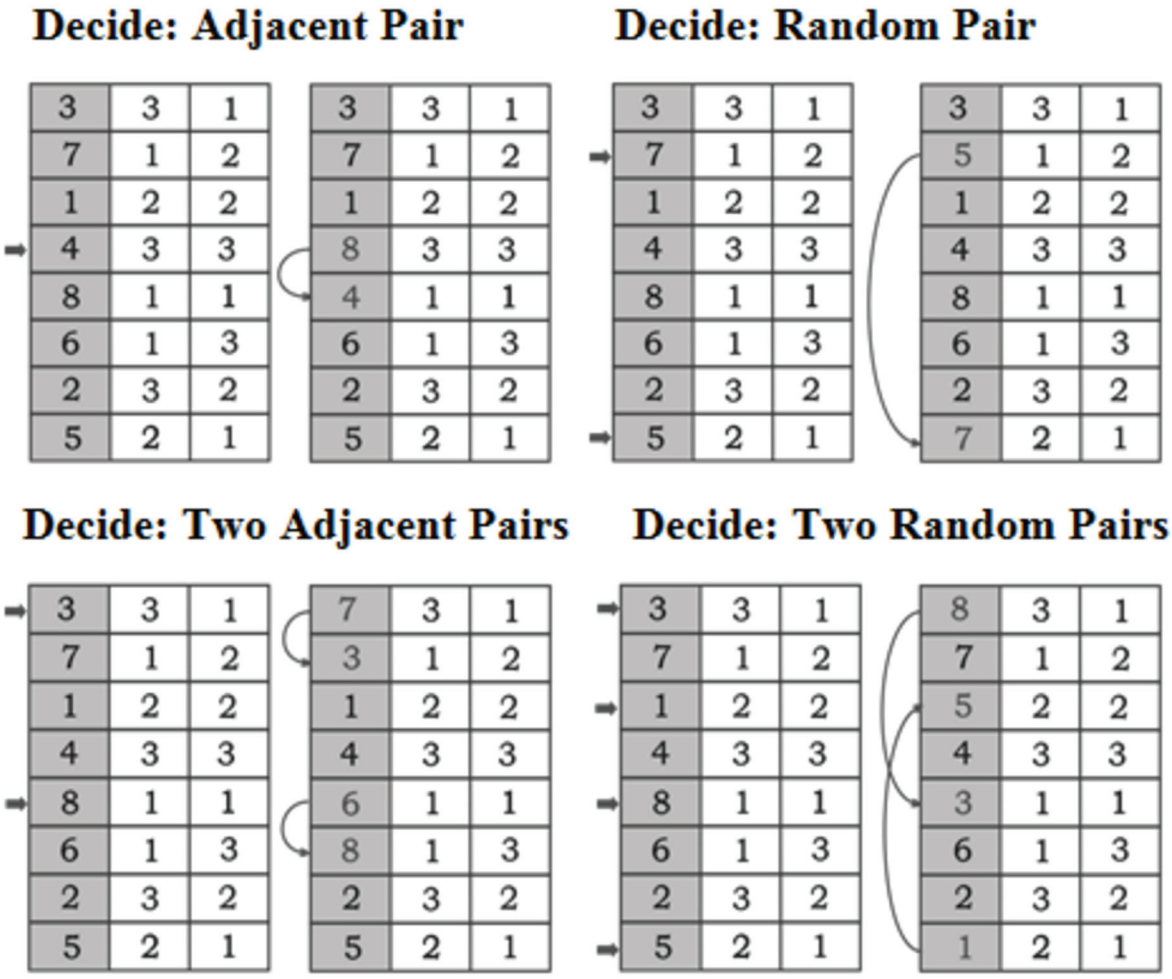


Figure 6. Random movements applied in the VNS for the UPMP



implemented when local search was used, in order to improve the efficacy of the Ant Colony Algorithm, and in CSTSP correspond to a local search.

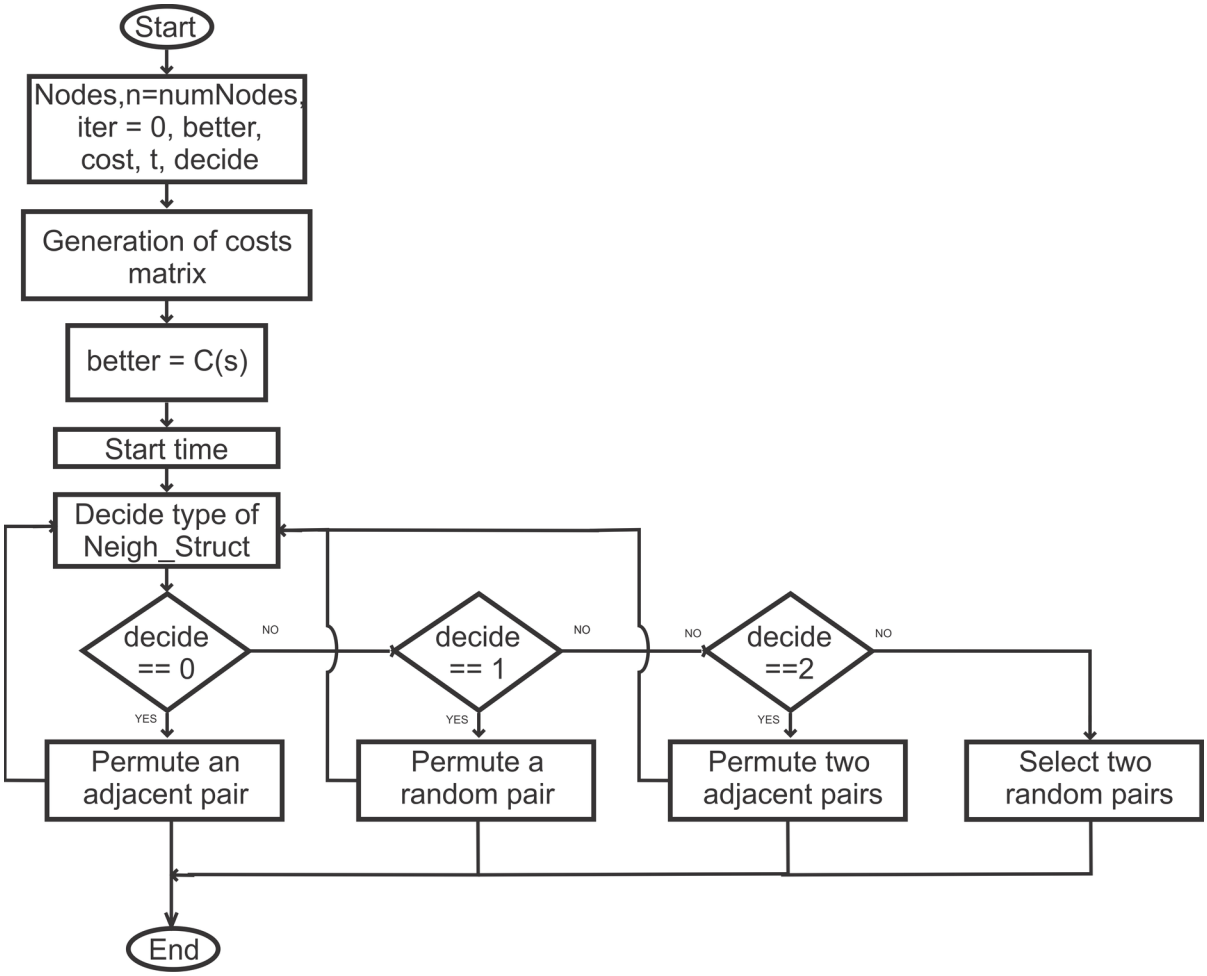
At the end of the iteration, the Ant Colony obtains its best solution so far, which has to be improved. Subsequently, the local search applies the structures randomly, until the stop criterion is met. The stop criterion could be processing time or a specific number of iterations. Local search procedure tries to improve the solution by obtaining neighboring solutions, which are evaluated according to the objective function and the problem constraints. Figure 7 shows the operation of the VNS (Cruz et al., 2010).

Experimental Results

Experimental testing was performed according to guidelines in (Arajy & Abdullah, 2010b; Barr et al., 1995), where the authors explain how to design and report computational experimental results of algorithms, taking into account a reasonable number of test problems executed under the same conditions. Tests were performed on a laptop with a Centrino Core2Duo 2.0 GHz., 3GB RAM, Windows Vista Home Premium.

Neighborhood structures and the Ant Colony algorithm were developed using Visual C 2008 compiler. Test problems were randomly generated for 200, 500, 1000, 2000, 4000, 5000, 6000, 7000, 8000, and 9000 cities for the CSTSP. In the case of UPMP, test problems

Figure 7. VNS for the UPMP and the CSTSP



were randomly generated for 200, 250, 270, and 300 jobs to be scheduled on 12 machines.

This research tested the hypothesis: "The improvement in solution quality depends not only on the applied VNS, but also on the problem complexity." This research undertook the CSTSP and UPMP, both NP problems, but with different hardness. Therefore, CSTSP has a more challenging solution space than UPMP (Papadimitriou & Steiglitz, 1998).

CSTSP: Experimental Results

According to (Arajy & Abdullah, 2010a; Barr et al., 1995), both problems were tested in all the neighborhood structures, including the proposed VNS. Each neighborhood structure performed 30 executions per test problem of 5 minutes each one, obtaining the best found solution, and the total iterations conducted during the specified time.

It is noteworthy, that problems were randomly generated because existing benchmarks are smaller than the instances proposed in this research. The test problems consist of a symmetric matrix of $n*n$, where n is the total number of cities to be visited in a tour. Along with the execution, only the best of the elitist solutions were selected, using time as stop-criteria.

This procedure allows for good performance and direct comparison among the results obtained for the different implemented neighborhood structures, enabling a reliable efficacy and efficiency analysis.

Table 1. Average of the results obtained in experimental testing for different sized problems

Prob_Size	Adjacent Pair	Random Pair	Two Adjacent Pairs	Two Random Pairs	Variable Neighborhood
200	2011.87	1994	1974	1945.13	**1944.96**
500	5161.43	5118.1	5123.6	**5057.3**	5098.13
1000	49661.27	49374.5	49202.43	49166.06	**49128.23**
2000	101074.97	100616.90	100466.86	100856.36	**100231.90**
4000	198292.23	198092.4	197373	197523.46	**197280.03**
5000	52344.83	51885.2	51761.7	51830.53	**51742.43**
6000	61215.9	61172.7	61178.2	61227.93	**61168.23**
7000	72035.5	72100.3	72029.5	72077.73	**72014.30**
8000	82539.57	82617.4	82283.5	82482.17	**82238.10**
9000	973283.4	973009.1	972938	972946.80	**972929.50**

Table 2. Standard Deviation obtained in experimental testing for different sized problems

Prob_Size	Adjacent Pair	Random Pair	Two Adjacent Pairs	Two Random Pairs	Variable Neighborhood
200	57.9	60.89	**48.41**	51.17	63.66
500	57.95	69.86	**57.38**	69.33	70.17
1000	**524.69**	586.61	578.73	788.93	794.61
2000	**519.92**	630.07	704.82	818.71	845.60
4000	**423.03**	575.06	540.39	616.91	874.81
5000	**263.7996**	277.564	271.6555	292.0796	322.302
6000	**171.4664**	196.9481	175.9626	215.913	353.9473
7000	**168.293**	203.0889	209.5244	314.2264	367.5941
8000	377.5554	420.6904	**373.8884**	439.077	971.8005
9000	219.3942	223.218	**207.4521**	252.1876	274.8439

Efficacy Testing

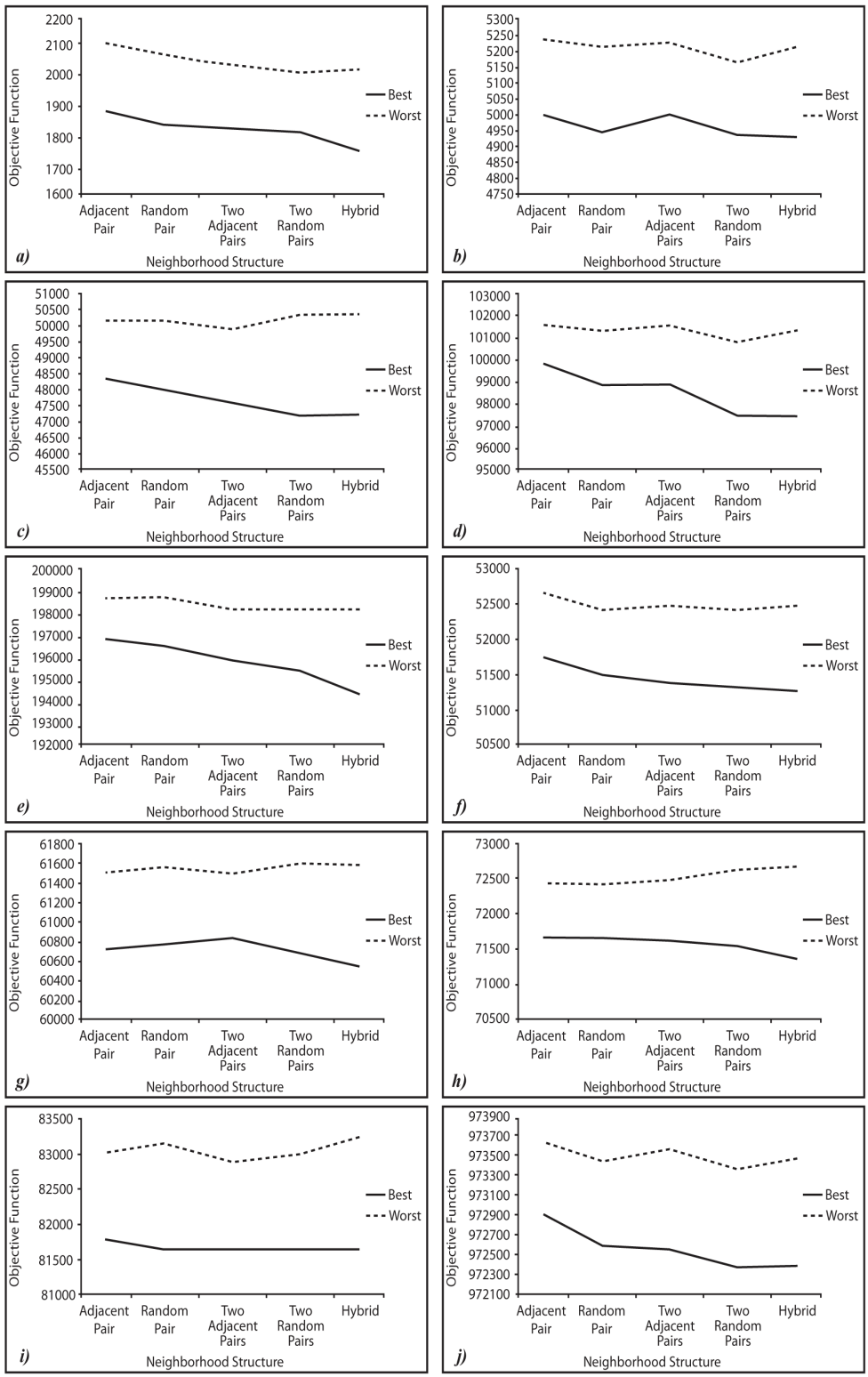
To understand the algorithm behavior, an efficacy analysis was conducted. This allows examination of the dispersion among the obtained solutions. On the other hand, the relative standard error is the percent error between the best-found solution and the best-known solution (in some cases, the optimal solution).

Experimental tests were performed using 10 test problems randomly generated for the CSTSP (200, 500, 1000, 2000, 4000, 5000, 6000, 7000, 8000, and 9000 cities). The problems were evaluated using the neighborhood structures explained in Section 5. Each neighborhood structure conducted 30 executions of 5 minutes each per test problem. Average and standard deviation of obtained results are shown in Tables 1 and 2.

According to calculations presented in Table 1, the most effective neighborhood structure in almost all cases is the VNS. The test problem of 500 cities is an exception; the most effective structure was that of two-random pairs, although the difference between the best obtained averages of both structures was minimal. This behavior is caused because heuristics do not guarantee optimality.

In the case of standard deviation (Table 2), the results obtained by the VNS show greater dispersion than the results obtained by the other structures. This is due to the neighborhood size variations. In this case the VNS allows a better exploitation of the solution neighborhood.

Figure 8. Best/worst solutions obtained for a)200, b)500, c)1000, d)2000, e)4000, f)5000, g)6000, h)7000, i)8000, and j)9000 cities, respectively



The analysis of the obtained results for each test problem is shown in the graphs below (Figure 8a-j), according to each neighborhood structure.

According to the results presented in graphs 10a–10j, there is a clear improvement when implementing the VNS for CSTSP versus using the single straightforward neighborhood structures. The improvement was obtained not only for the best found solution, but for the worst one also, independent of the problem size. These results are consistent with those reported in literature.

Efficiency Testing

The efficiency is a measure of the time required to find a high-quality solution. In this research, efficiency was calculated for 1000 solutions found by each neighborhood structure, and the different sizes of test problems (200, 500, 1000, 2000, 4000, 5000, 6000, 7000, 8000, and 9000 cities) that were studied. Results are shown in Figure 9.

Figure 9 shows the behavior of different structures as they find 1000 solutions, independent of the test problem's size. According to the graph, the behavior of all the evaluated neighborhood structures is very similar for small instances (200-2000 cities). Starting at 4000 cities, some neighborhood structures begin to require more computational effort to find solutions.

Naturally, the interesting cases are the results obtained for large problems (4000-9000 cities). In these cases, there is a clear difference in efficiency under the same conditions, when the size of the instance varies. Thus, Figure 9 demonstrates that the VNS's efficiency is competitive, because it shows better efficiency than the neighborhood structure of two-adjacent pairs and two-random pairs in most of cases, falling in the middle of the efficiency graph. This is a logical behavior because it includes the characteristics of the four neighborhood structures, which are randomly selected.

UPMP: Experimental Results

Experimental tests for UPMP and CSTSP were conducted equally. Therefore, all the instances were tested, and the Ant Colony algorithm was implemented in all of the neighborhood structures, including the VNS. Each neighborhood structure performed 30 executions per instance.

Instances were randomly generated based on a matrix of $n*mn$, where n is the number of jobs that have to be scheduled in m machines with k positions each one ($k=n$). Along with the execution, only the best solution is selected, using time as stop criteria. The Ant Colony algorithm was executed for 3 hours, obtaining the best-found solution, and the total number of iterations performed during the specified time.

This procedure allows for good performance and direct comparison among the results obtained for the different implemented neighborhood structures, enabling a reliable efficacy and efficiency analysis.

Efficacy Testing

Experimental tests were performed using 4 test problems randomly generated for the UPMP (200, 250, 270, and 300 jobs), which have to be scheduled on k positions of 12 machines. The problems were evaluated using the neighborhood structures explained in Section 5, which were implemented into an Ant Colony algorithm, in order to get an improvement in the quality of solution. The enhancement procedure was applied to the neighborhood of the best solution so far. The Ant Colony algorithm conducted 30 executions per test problem on each neighborhood structure. The executing time was almost 3 hours, depending on the input size. Average and standard deviation of obtained results are shown in Tables 3 and 4.

According to calculations presented in Table 3, and contrary to results obtained for CSTSP using the same neighborhood structures, the best solution was found in all cases by the random-pair neighborhood structure, leaving the VNS in second place. With respect to standard deviation, as in the CSTSP, the VNS has the highest dispersion of solutions.

The optimal solution was obtained using the Simplex algorithm, which spent three days finding the optimal. It is noteworthy that the Simplex algorithm had to be executed on a workstation with memory of 20 GB and an Intel Xeón Cuad-Core 3.16 GHz. processor, due to the nature of the method.

Standard relative error (Table 5) was calculated to determine how close the obtained solution was to the optimal.

These results (Figure 10) show clearly that the best option for UPMP is a straightforward random-pair structure, due to its low relative standard error

Figure 9. Time required for five different neighborhood structures to find 1000 solutions for test problems from 200-9000 cities

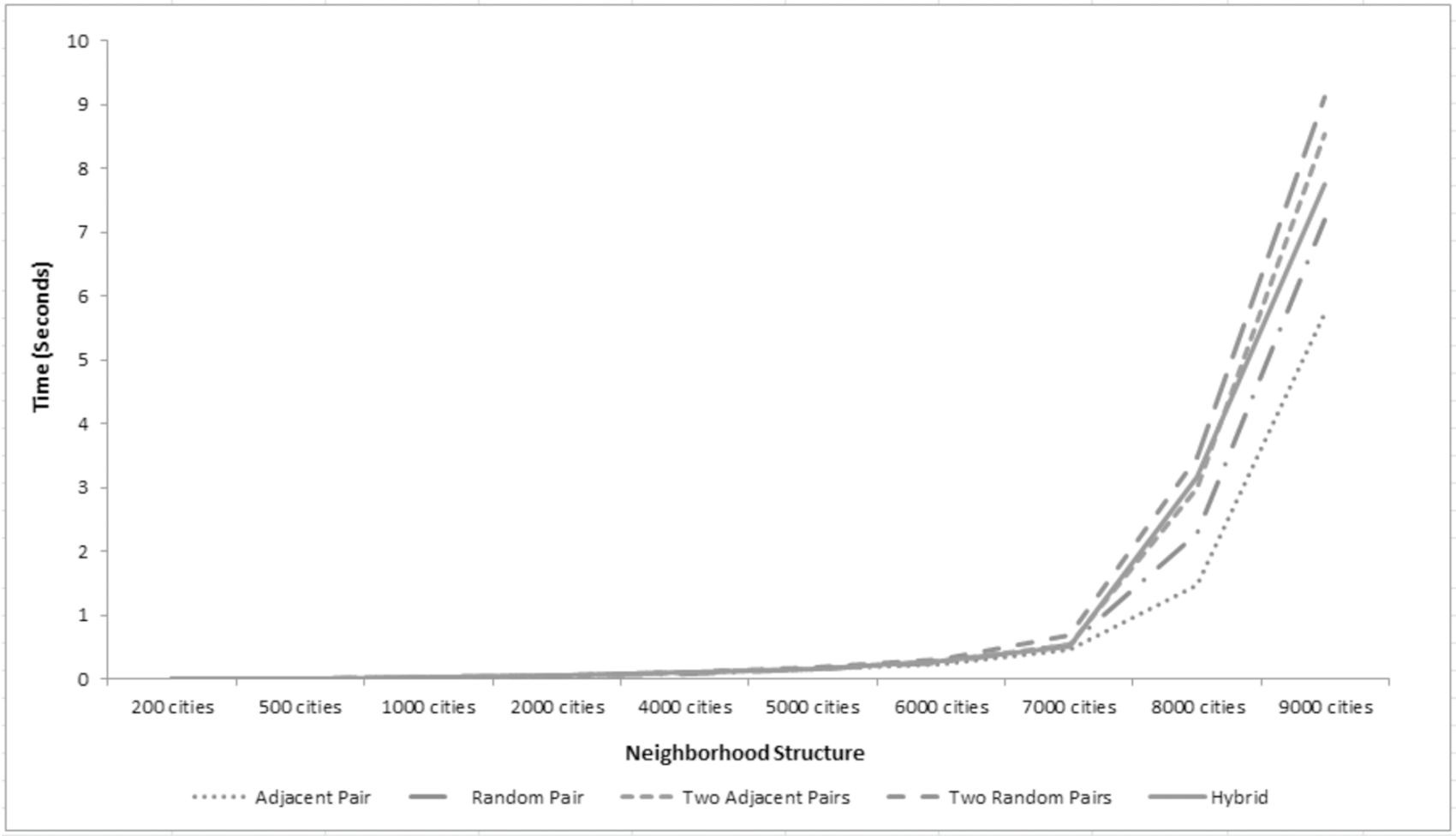


Table 3. Average results obtained in experimental testing for different sized problems

Prob_Size Jobs * Machine	Adjacent Pair	Random Pair	Two Adjacent Pairs	Two Random Pairs	Variable Neighborhood
200*12	2648.00	2316.60	2768.40	2678.00	2320.00
250*12	3928.60	3460.17	3465.00	3997.70	3515.07
270*12	4907.67	4279.00	4389.34	4986.21	4581.9
300*12	5948.07	5139.00	5232.12	5974.98	5387.87

Table 4. Standard Deviation obtained in experimental testing for different sized problems

Prob_Size Jobs * Machine	Adjacent Pair	Random Pair	Two Adjacent Pairs	Two Random Pairs	Variable Neighborhood
200*12	59.90	91.41	82.67	112.60	139.32
250*12	68.32	105.03	97.12	129.78	134.43
270*12	79.95	159.86	118.09	339.12	449.04
300*12	95.34	118.34	175.03	253.90	280.48

Table 5. Relative Standard Error obtained for the evaluated structures using different sized problems

Prob_Size Jobs * Machine	Adjacent Pair	Random Pair	Two Adjacent Pairs	Two Random Pairs	Variable Neighborhood
200*12	2.26%	0.94%	3.81%	2.35%	1.88%
250*12	3.23%	0.97%	4.14%	3.62%	1.92%
270*12	3.77%	0.99%	5.74%	3.99%	1.97%
300*12	4.00%	1.11%	6.71%	4.11%	2.14%

of around 1%, which represents a great improvement in the solution quality. The VNS achieved a relative standard error of 2%, which is considered slightly high within the combinatorial optimization area.

Comparing the obtained results of different neighborhood structures, it can be observed that it is possible to obtain better-quality solutions for this problem by applying a straightforward structure. These results are not consistent with the literature.

This research demonstrates that although CSTSP and UPMP are both NP problems, their hardness is different. CSTSP has a harder solution space than UPMP (Papadimitriou & Steiglitz, 1998). Therefore, the VNS has more benefits for hard problems, as opposed to less-hard problems, where it is easier to find good solutions.

The analysis of the best obtained results for each test problem is shown in Figure 11. The behavior of each neighborhood structure for some specific problems is compared with the optimal solution.

Figure 11 shows the value of the best solutions obtained by different neighborhood structures for different sized test problems, comparing the results with the optimal. According to these results, the high improvement was achieved by the random-pair neighborhood structure, leaving the VNS in second place with 2% standard relative error.

Efficiency Testing

The efficiency of this algorithm was obtained by comparing the time required to find a high-quality solution with the time required by the Simplex algorithm. In the case of Classical Simplex, the time required to find the optimum grows exponentially as the input size increases. The comparison between Simplex and Ant Colony with a VNS is shown in Figure 12.

To obtain an efficiency analysis, some calculations of the required time to find a solution were performed

Figure 10. Relative Standard Error obtained by evaluated neighborhood structures

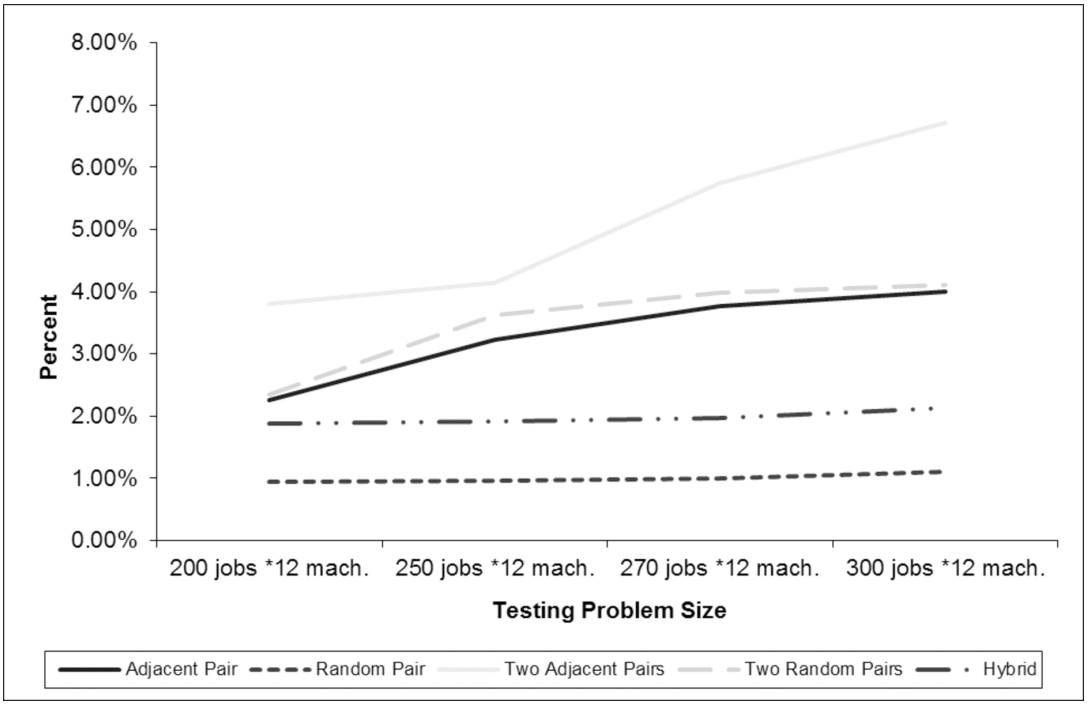


Figure 11. Best Solution obtained by evaluated neighborhood structures for test problems of 200, 250, 270, and 300 jobs with 12 machines

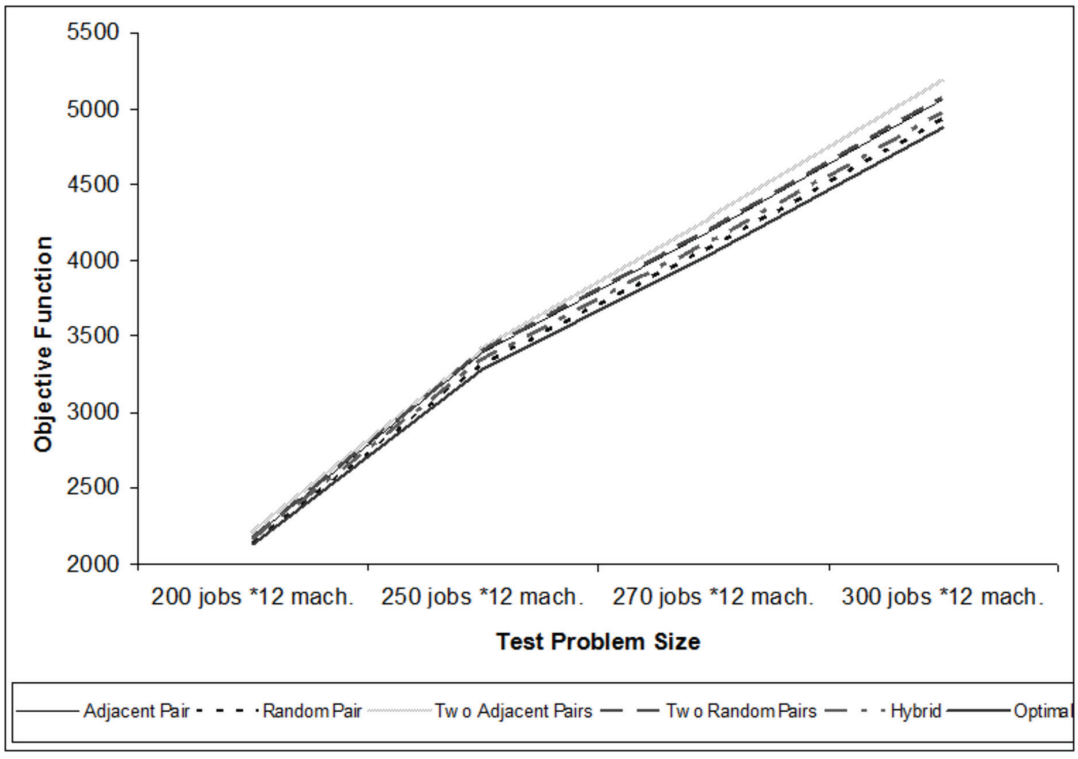
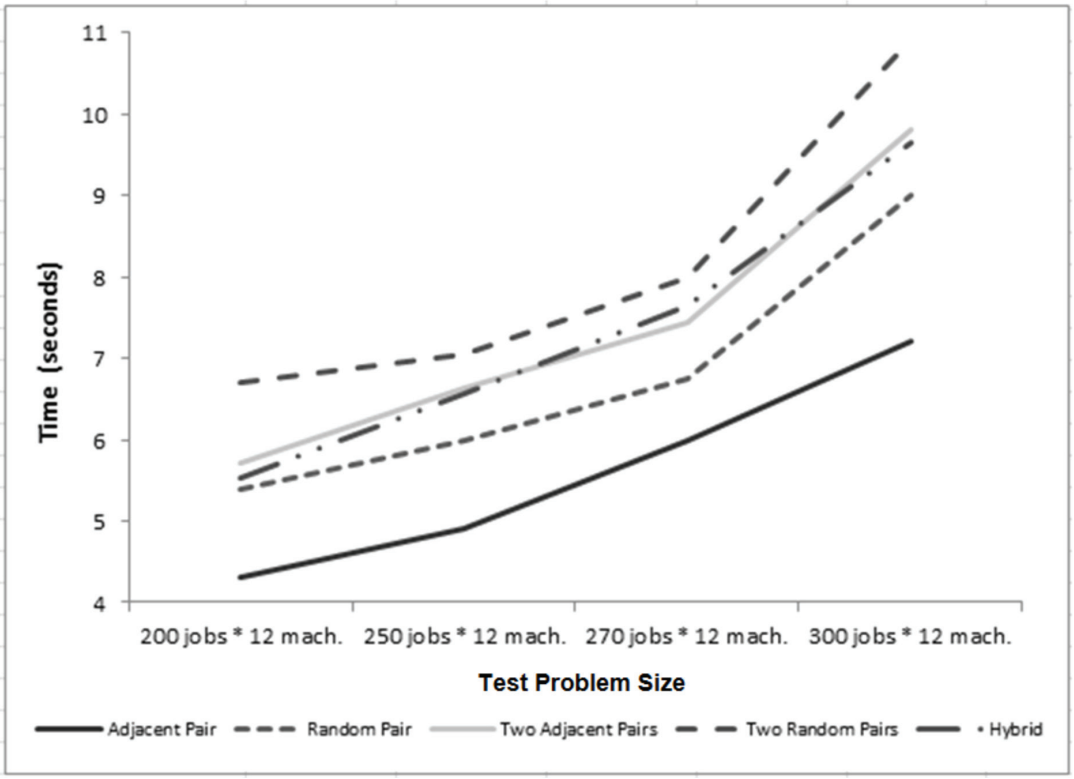


Figure 12. Time required for each neighborhood structure to find a solution



for each neighborhood structure, according to the test problem size. Figure 12 shows graphically the conducted calculations. In the case of CSTSP, it shows that the efficiency of the VNS is competitive with the other structures evaluated for two problems with different complexities. The experimental results were not as expected, because the VNS behaved differently for UPMP than for CSTSP, where the structure demonstrated excellent performance.

CONCLUSION

The most important aspects when evaluating metaheuristics are efficacy and efficiency. Efficacy refers to quality of solutions compared with the best-known bounds. Efficiency refers to the time required for the algorithm to find good-quality solutions.

Many experimental tests were performed for two problems (CSTSP and UPMP) with different hardness. In both cases, test problems were randomly generated, according to the problems' features. Experimental results showed a difference in the VNS performance. In the case of CSTSP, excellent-quality solutions were obtained, but for UPMP the structure did not obtain the best solutions.

According to experimental analysis, the hypothesis presented at this article is confirmed. The contribution of this research is the experimental proof that not all the VNSs work well in all discrete optimization problems versus other straightforward structures. This is seen clearly in the UPMP, where a straightforward structure gets better quality solutions than the VNS.

This research demonstrates that although CSTSP and UPMP are both NP, CSTSP has a harder solution space than UPMP (Papadimitriou & Steiglitz, 1998). Therefore, the VNS has more benefits for harder problems, as opposed to less-hard problems, where it is easier to find good solutions. In industry, this is shown as savings in time, gas and reducing the quantity of vehicles used for delivery, as well as salary of the drivers.

FUTURE RESEARCH DIRECTIONS

This research is the first step before applying the proposed VNS with a heuristic method. In addition, the experimental tests will be performed to different combinatorial problems, taking into account P problems, with the aim of reinforce the hypothesis presented in this work.

REFERENCES

Alba, E. (Ed.). (2005). Parallel Metaheuristics. A New Class of Algorithms (pp. pp.172-174). United States of America: Wiley-Interscience Ed. ISBN-13.978-0-471-67806-9 and ISBN-10.0-471-67806-6.

Arajy Yahya, Z., & Abdullah, S. (2010a). Hybrid Variable Neighborhood Algorithm for Attribute Reduction in Rough Set Theory. In *Proceedings of the* 10th *International Conference on Intelligent Systems Design and Applications, ISDA*. Cairo, Egypt: IEEE.

Arajy Yahya, Z., & Abdullah, S. (2010b). Hybrid Variable Neighbourhood Search Algorithm for Attribute Reduction in Rough Set Theory. In *Proceedings of the 10th International Conference on Intelligent Systems Design and Applications*.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., & Protasi, M. (1999). *Complexity and Approximation: Combinatorial Optimization Problems and their Aproximability Properties*. Springer-Verlag. doi:10.1007/978-3-642-58412-1

Barr, R. S., Golden, B. L., Kelly, J. P., Resende, M. G. C., & Stewart, W. R. (1995). Designing and Reporting on Computational Experiments with Heuristic Methods. *Journal of Heuristics*, *1*(1), 9–32. doi:10.1007/BF02430363

Cortéz, A. (2004) Computational Complexity Theory and Computability Theory, Rev. In *Investig. Sist. Inform* (pp.102-105). INFOSYS. ISSN:1815-0268.

Cruz-Chávez, M. A., Díaz-Parra, O., Juárez-Romero, D., & Martínez-Rangel, M. G. (2008). Memetic Algorithm Based on a Constraint Satisfaction Technique for VRPTW. Lecture Notes in Artificial Intelligence, *5097*, (pp.376-387). Springer-Verlag Pub., Berlin Heidelberg. ISSN:0302-9743.

Cruz Chávez, M. A., Martínez Oropeza, A., & Serna-Barquera, S. A. (2010). Neighborhood Hybrid Structure for Discrete Optimization Problems. *In Proceedings of the IEEE Electronics, Robotics and Automotive Mechanics Conference* (pp.108-113). CERMA. ISBN-13:978-0-7695-4204-1.

Fischetti M., Salazar-González J.J. & Toth P.(1995). The Symmetric Generalized Traveling Salesman Polytope. *Journal NETWORKS, . 26*(2), 113-123. CCC 0028-3045/95/020113-11.

Garey, M. R., Johnson, D. S., & Shethi, R. (1976). The Complexity of Flow Shop and Job Shop Scheduling. *Mathematics of Operations Research*, *1*(2), 117–129. doi:10.1287/moor.1.2.117

González-Velázquez, R., & Bandala Garcés, M. A. (2009). *Hybrid Algorithm: Scaling Hill and Simulated Annealing to Solve the Quadratic Allowance Problem*. México: 3th. Latin-Iberoamerican Workshop of Operation Research.

Hansen, P., & Mladenović, N. (1999). Variable Neighborhood Search: Principles and Applications. *European Journal of Operational Research*, 449–467.

Kenneth, D. B. (1995). *Cost Versus Distance in the Traveling Salesman Problem. Los Angeles, CA*. CiteSeer, USA: UCLA Computer Science.

Lin, S., & Kernighan, W. (1973). An Effective Heuristic for the Traveling Salesman Problem. *Operations Research*, *21*(2). doi:10.1287/opre.21.2.498

Liu, S. B., Ng, K. M., & Ong, H. L. (2007). *A New Heuristic Algorithm for the Classical Symmetric Traveling Salesman Problem* (pp. 267–271). World Academy of Science, Engineering and Technology.

Lourenço, H. R., & Martin, O. C. (2003). Iterated Local Search. In Handbook of Metaheustics. International Series in Operations Research & Management Science, 57, (pp.320-353). SpringerLink.

Martin O., Otto S.W. & Felten E.W.(1991). Large Step Markov Chains for the Traveling Salesman Problem. *Complex Systems*, 299-326.

Martin O., Otto S.W. & Felten E.W.(1992). Large Step Markov Chains for the TSP Incorporating Local Search Heuristics. Operations Reasearch, 219-224.

Michaelewicz, Z., & Fogel, D. B. (2000). How to Solve It: Modern Heuristics (pp.117-125). New York: Springer-Verlag, Berlin Heidelberg. ISBN.3-540-66061-5

Pacheco, J., & Delgado, C. (2000). Different Experiences Results with Local Search Applied to Path Problem. Electronic Journal of Electronics of Comunications and Works, *2*(1), 54-81. ISSN:1575-605X. ASEPUMA.

Papadimitriou, C. H., & Steiglitz, K. (1998). *Combinatorial Optimization, Algorithms and Complexity. Inc*. Mineola, New York, USA: Dover Publications.

Pinedo, M. L. (2008). Scheduling Theory, Algorithms, and Systems. Third Edition. New York University, Ed. Prentice Hall, Springer. ISBN:978-0-387-78934-7, e-ISBN:978-0-387-78935-4.

Voudouris, C. H., & Tsang, E. (1998). Theory and Methodology Guided Local Search and its Application to the Traveling Salesman Problem. [Elsevier Science.]. *European Journal of Operational Research*, 469–499.

ADDITIONAL READING

Cruz-Chávez, M. A., Díaz-Parra, O., Juárez-Romero, D., Barreto Sedeño, E., Zavala Díaz, C., & Martínez Rangel, M. G. (2008). Un Mecanismo de Vecindad con Búsqueda Local y Algoritmo Genético para Problemas de Transporte con Ventanas de Tiempo, ACD, Junio pp. 23-32, 25-27. CICos 2008 6to Congreso Internacional de Cómputo en Optimización y Software.

Cruz-Chávez, M. A., Rodríguez-León, A., Rivera-López, R., Juárez-Pérez, F., Peralta-Abarca, C., & Martínez-Oropeza, A. (2012). Logistics Management and Optimization through Hybrid Artificial Intelligence Systems, Chapter 3: Grid Platform Applied to the Vehicle Routing Problem with Time Windows for the Distribution of Products. Ed. Carlos Alberto Ochoa Ortiz Zezzatti et al., ISBN13: 9781466602977, IGI Global, 422 pages, march 2012.

Hansen, P., & Mladenovi, Ć. N. (2001). *Variable Neighborhood Search: Principles and Applications*. Pp. 449-467. [Elsevier Science.]. *European Journal of Operational Research*, *130*(Issue. 3). doi:10.1016/S0377-2217(00)00100-4

Martínez-Bahena, B., Cruz-Chávez, M. A., Díaz-Parra, O., Martínez Rangel, M. G., Cruz Rosales, M. H., & Peralta Abarca, J. del C. & Juárez Chávez J.Y. (2012) *Neighborhood Hybrid Structure for Minimum Spanning Tree Problem*. Electronics, Robotics and Automotive Mechanics Conference, CERMA2012, IEEE-Computer Society, ISBN: 978-0-7695-4878-4, pp 191 - 196, November 20-23, México.

Polacek M., Hartl R.F. & Doerner K (2004). *A Variable Neighborhood Search for the Multi Depot Vehicle Routing Problem with Time Windows*. Pp. 613-627. Journal of Heuristics. Vol. 10, Issue 6. 10: 613-627. Springer Link. Netherlands.

KEY TERMS AND DEFINITIONS

Combinatorial Optimization: Area of computer science that studies difficult combinatorial problems by means of the applications of algorithms theory trying to solve them bounding ad exploring their search space.

Heuristics: Computational methods based on experience applied to tackle difficult combinatorial problems in a reasonable time without ensure optimality.

Local Search: Method that starts with an initial solution then applies a sequence of local changes in attempt to improve the value of the objective function and obtain the local optima.

Neighborhood: Set of solutions close to a given initial solution that could be reached applying a σ movement.

Neighborhood Structures: Technique implemented with an algorithm in order to exploit the solution neighborhood, with the intent to improve the quality of solutions, according to the objective function of the undertaken problem. Neighborhood structures define the size of the neighborhood.

NP-Complete: Well-known as intractable problems, are the hardest problems classified by the complexity theory. Therefore, there is no known exact algorithm that solve them in all their instances, thus it is necessary to use heuristic methods to find high-quality solutions in a reasonable computational time.