Chapter 3
Grid Platform Applied to the Vehicle Routing Problem with Time Windows for the Distribution of Products

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

Fredy Juárez-Pérez
Autonomous University of Morelos State, CIICAp, Mexico

Abelardo Rodríguez-León
Technological Institute of Veracruz, Mexico

Carmen Peralta-Abarca
Autonomous University of Morelos State, FCQeI, Mexico

Rafael Rivera-López
Technological Institute of Veracruz, Mexico

Alina Martínez-Oropeza
Autonomous University of Morelos State, CIICAp, Mexico

ABSTRACT

Around the world there have recently been new and more powerful computing platforms created that can be used to work with computer science problems. Some of these problems that are dealt with are real problems of the industry; most are classified by complexity theory as hard problems. One such problem is the vehicle routing problem with time windows (VRPTW). The computational Grid is a platform which has recently ventured into the treatment of hard problems to find the best solution for these. This chapter presents a genetic algorithm for the vehicle routing problem with time windows. The algorithm iteratively applies a mutation operator, first of the intelligent type and second of the restricting type. The algorithm takes advantage of Grid computing to increase the exploration and exploitation of the solution space of the problem. The Grid performance is analyzed for a genetic algorithm and a measurement of the latencies that affect the algorithm is studied. The convenience of applying this new computing platform to the execution of algorithms specially designed for Grid computing is presented.

DOI: 10.4018/978-1-4666-0297-7.ch003

INTRODUCTION

The problems of resource assignment based on scheduling are well-known combinatorial problems in the computer science research community. Complexity theory classifies them as a very difficult set of problems to solve; they are the NP-complete problems (Garey & Johnson, 1979). This complexity is due to the combinatorial feature of these problems to handle only discrete variables in their formulation, and because the number of solutions to the problem grows exponentially with increasing size of the instance to solve. At times, the instance of a problem containing a small number of variables is an unknown solution; we have only knowledge of an upper or lower bound close to the solution. This behavior occurs in scheduling problems, for example, in transportation problems with time windows for 1000 clients, as proposed by Homberger and Gehring (1999), there is currently no known solution due to the hardness of this problem (NP-complete), as mentioned by Toth and Vigo (2001). For this type of symmetrical transport problems, the solution space size is bound as $((1+n/r)!)^r$, much like a multiple traveling salesman problem as presented by Mitrovic-Minic and Krishnamurti (2006), where n is the number of customers that need attention and r indicates the number of paths that must be traveled, the symmetry indicates that there are the same number of customers per path.

Due to the hardness of solving problems of resource assignment, it is necessary use nondeterministic Meta heuristics to bind in polynomial time the search for the best possible solution with good performance in both efficacy and efficiency for the algorithm. For this reason, the scientific community has worked to improve the performance of Meta heuristics in several ways. For example, one way is the hybridization of two or more methods, one of which is an exact method, thereby taking advantage of the best features of each. Another way to improve the performance of the Meta heuristic is to improve the neighborhood structure for local search, because several of these methods work with iterated local search, as suggested in Hansen and Mladenovic (2001) and Cruz-Chavez et al. (2010). Another way is the partial or total parallelization of the algorithm. All these alternatives have been successful, resulting in the improvement of the upper/lower bounds of solutions for unsolved instances. The main Meta heuristics that have been applied to VRPTW are genetic algorithms, used to solve constraint satisfaction models, ant colony and simulated annealing.

The industry needs a transport to distribute their products and the savings achieved through the efficient scheduling of distribution paths. This in turn could lead to a decrease in transportation costs, resulting in a substantial savings for the company, making it more competitive nationally and/or internationally, due to decreased price for the same product quality. Good resource assignment will result in savings in several ways: less gasoline, fewer transport units and therefore savings in depreciation of the units, fewer expenses in payment to drivers who have a smaller number of paths to meet (every driver completes their daily work in a single path). In this type of resource assignment, good use of the storage capacity of the units is also involved. Better use of the transport units' space and a better selection of merchandise to be transported lead to the transportation optimum.

The type of infrastructure used to solve the VRPTW has been varied over the years; PCs, workstations, supercomputers and clusters have been used. Considering that this is a hard problem, and attempts are made to find the global optimum solution, a new type of infrastructure is currently being used in the world for application in various areas. It is beginning to be used in the area of combinatorial optimization to minimize the runtime of Meta heuristics as applied to NP-complete problems. Cruz-Chavez et al. (2010b) use the new infrastructure, called grid computing, in their work with the Job Shop Scheduling Problem to reduce communication between nodes on a Grid. Grid

computing is simply the union of several clusters of computers that are geographically distant, using an operating system to work with the cluster of computers as a single supercomputer. The Grid has been used in Fujisawa et al. (2004) to work with optimization problems. The Meta heuristic that has been used in the Grid, due to ease of programming in a distributed environment, is the genetic algorithm, as presented in Chang et al. (2006), Lim et al. (2007), and Moon et al. (2008).

The approach generated in the use of Grid computing to work with NP-complete hard problems in computational science involves the availability of all the processing cores of the Grid, in order to implement a computer program that can launch a number of processes. This allows the tasks of the algorithm to be divided, in this case a Meta heuristics, in order to obtain good solutions to the problem in a very short execution time or to significantly increase the search of the solution space to improve or find the global optimum, while requiring a reasonable execution time.

Working with the VRPTW, or any NP-complete type problem in a Grid environment, requires the development of two important elements. The first element is the generation of a computational Grid infrastructure that will allow running a computer program which can make use of all available processing cores on the Grid. At present, most of the Grids in the world allow the use of a single computing cluster for the execution of a computer program. The second element is the design of a computer program that allows the use of total processing cores in parallel/distributed, and also helps reduce the total latency in the execution of computer program. Latency is the time required to bring and take a data packet between a pair of clusters that belong to the Grid.

This infrastructure was developed as an experimental MiniGrid between two educational institutions; the MiniGrid consists of two computer clusters, a cluster located at the Autonomous University of Morelos State in Cuernavaca, Morelos and the other cluster located in the Technological Institute of Veracruz, in Veracruz, Veracruz. The link to the MiniGrid is Internet 2 which offers a bandwidth of 5 Gbs.

To use all of the MiniGrid cores easily, a genetic algorithm computer program was chosen because the parallelization and distribution of the algorithm in the MiniGrid becomes very simple in relation to the division of the population of the genetic design in the number of cores available. The genetic design includes reducing the total latency generated during the execution of the computer program.

The Grid architecture provides an alternative to the use of computers with parallel processing supercomputing, which is still extremely expensive and therefore available in very few institutions. The use of Grid computing enables the sharing of computing resources among academic and private institutions. This represents an area of opportunity both in research and in industry application. An example of using a Grid for optimization problems is shown, it is a genetic algorithm of the VRPTW, which is of interest to researchers but also has wide application in industry.

BACKGROUND

This chapter presents a model of the vehicle routing problem with time windows and the development of a genetic algorithm for this problem which works in a Grid environment. A description of the characteristics of the Experimental MiniGrid developed to study the VRPTW follows. Next, the tests run using the MiniGrid with a genetic algorithm for the VRPTW are shown. Finally the conclusions of this chapter are presented.

VRPTW MODEL

The vehicle routing problem with time windows, VRPTW, is a model that applies to the supply chain and school transport. The following is the

mathematical model of Integer Linear Programming representing the problem and as presented by Toth and Vigo (2001):

VRPTW Mathematical Model

$$\min f = \sum_{k \in K} \sum_{(i,j) \in A} C_{ij} x_{ijk} \tag{1}$$

s.t.

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1 \tag{2}$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1 \tag{3}$$

$$\sum_{i \in \Delta^-(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{ijk} = 0 \tag{4}$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1 \tag{5}$$

$$w_{ik} + s_i + t_{ij} + w_{jk} \leq \left(1 - x_{ijk}\right) M_{ij} \tag{6}$$

$$a_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq w_{ik} \leq b_i \sum_{j \in \Delta^+(i)} x_{ijk} \tag{7}$$

$$E \leq w_{ik} \leq L \tag{8}$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C \tag{9}$$

$$x_{ijk} \geq 0 \tag{10}$$

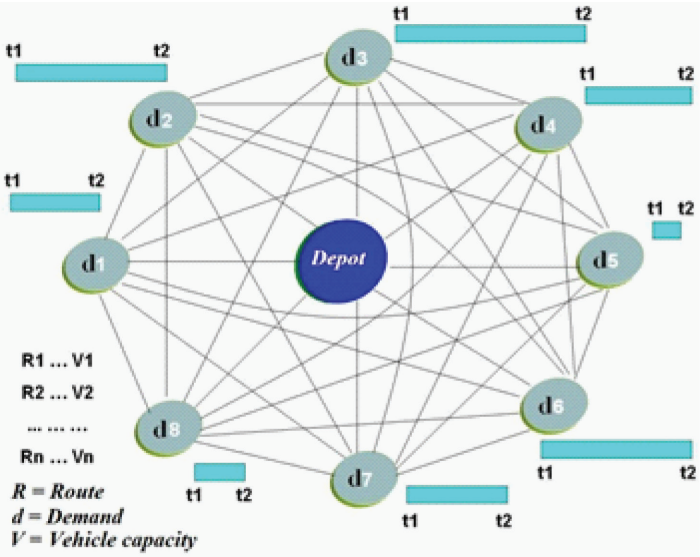
$$x_{ijk} \in \{0,1\} \tag{11}$$

The objective function presented in (1) represents the total cost, which can be interpreted as the travel time, distance traveled or the total cost generated in monetary units. It is required to obtain the minimum cost of the total travel using the smallest number of vehicles. The x_{ijk} variable takes a value of 1, when the k vehicle serves the route from client *i* to client *j*. The depot is represented as $i = 0$ or $i = 1 + n$.

The constraints that must be met are presented in (2) through (11). Constraint (2) restricts the assignment of each customer to a single vehicle route, i.e., customers who belong to the same route are served by the same vehicle. Constraints (3) to (5), stipulate the characteristics of the route to be followed by each vehicle *k*. Constraint (3) defines, for each vehicle *k*, the number of customers *j* that can be reached directly without going through another customer from the depot 0, i.e., for each vehicle *k*, only one client *j* can be reached if it comes from the depot. Constraint (4) indicates that for each vehicle unit, the number of vehicles arriving to a client is the same number of vehicles that exit.

Constraint (5), defines, for each vehicle *k*, the number of customers *i* that can directly reach the deposit $n + 1$. That is, for each vehicle *k*, there is only one client that connects to the depot. Restriction (6) through (8) and (9) guarantee the feasibility of a solution with respect to time conditions and aspects of capacity respectively. Restriction (6) indicates that a service to client *j* cannot start if client *i* has not been served and if the vehicle has not reached customer *j*. Here, *M* is a constant which allows constraint (6) to be satisfied in the special case that vehicle *k* does not address the route from client *i* to client *j*. Constraint (7) indicates that for each vehicle unit and each customer *i*, the initial service time w_{ik} must be started within the time window $[a_i, b_i]$, where a_i is the lower bound time and b_i is the upper bound time for initiation of service of vehicle unit *k* to client *i*. Constraint (8) indicates that for each vehicle unit *k* in the depot *0*, the start time of service w_{ik} must begin

Figure 1. Instance of the VRPTW with 8 customers



within the time window [*E, L*], i.e., there is a time window in which vehicle *k* is allowed out of the depot to serve customers. Constraint (9) indicates that for every vehicle *k*, the sum of the demands of all customers served should not exceed the vehicle capacity. Constraint (10) forbids negative values for the variables *x*. Constraint (11) defines the linear model as a binary integer linear model.

VRPTW Representation by a Disjunctive Graph

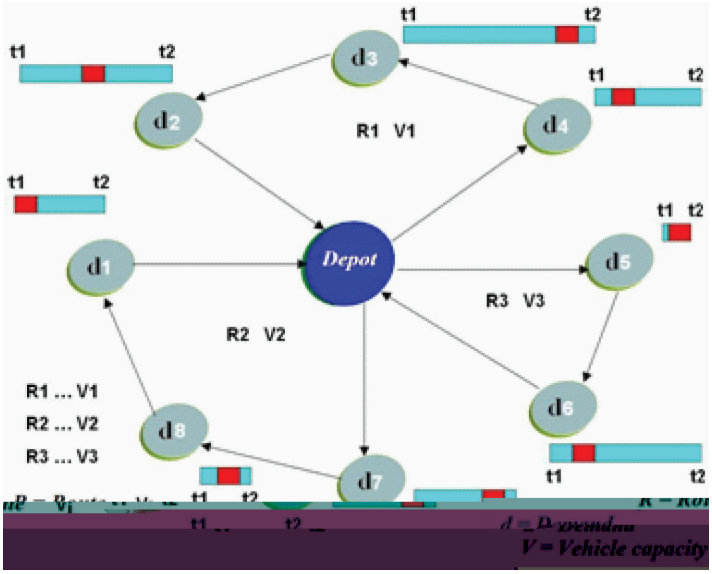
An instance of the VRPTW can be represented by a disjunctive graph which forms a clique. Figure 1 shows an instance for the problem of 8 clients. The graph represents, in a general form, the planning of the problem, each vertex representing a client. Each client has a time window $[t_i, t_j]$ in which it can receive the goods. At the beginning, each client has the ability to connect with any customer and the depot.

One solution to this instance of 8 clients is to define the number of routes to be used (one route per vehicle), so as to comply with the restrictions (2) through (11). A solution is presented in the digraph in Figure 2. The solution graph shows each time window in red, the time of service w_{ik} to the customer. The solution presented defines three routes, thus three vehicles used for delivery. It also shows the order in which each vehicle serves its customers. The restrictions presented explicitly in the solution graph in Figure 2 are (2), (3), (4), (5), (6) and (7). Naturally, some of the restrictions seen in the mathematical model are treated implicitly; these are (8), (9), (10) and (11).

GENETIC ALGORITHM IN THE GRID ENVIRONMENT FOR THE VRPTW

For the genetic design of distributed parallel processing in the MiniGrid, the computer program of the genetic algorithm was structured to reduce the total latency time that is generated in the use of the MiniGrid. In the genetic design, the weakness of this kind of Meta heuristics was addressed. The algorithm applied population genetics work efficiently in exploring the solution space to find the best solution, but they have a deficiency. The exploitation they do in the solu-

Figure 2. Solution for the VRPTW instance with 8 clients



tion space in search of local optima is very weak because they do not worry about finding the best solution in the neighborhoods that would generate each individual in population genetics. That is, they do not find the local optimum in a solution neighborhood as defined by each individual in the population. This is important because a local optimal solution can also be the global optimal solution. To improve the search operation, genetic hybridization is performed in the mutation phase by applying two types of iterative mutations, using a local search such as that defined by Papadimitriou and Steiglitz (1982). The first type of mutation is a restrictive one, as presented by Cruz-Chavez et al. (2007), which is a mutation iteratively solving the constraint satisfaction model of the VRPTW that is presented in equations (2) through (11) of the mathematical model. The second type of mutation is an intelligent mutation, as presented by Díaz-Parra and Cruz-Chavez (2008), which develops an iterative intelligent mutation for each individual to be mutated. The type of mutation used in genetic execution is random. This type of genetic mutation applied in an iterative genetic procedure, known as iterated local search, is usually referred to as a memetic algorithm. Moscato and Cotta (2010) give a comprehensive explanation of the features of a memetic algorithm.

With the advantages provided by the MiniGrid that can use all the cores, the parallelization of the computer program is structured so that the execution of the mutation is sent iteratively to the MiniGrid nodes as a cooperative process for each individual in the population to be mutated. Individuals are distributed in all MiniGrid nodes through a communication scheme presented in Rodriguez-Leon et al. (2010). This is done in each generation of the genetic. The mutation process is the most time consuming portion of the algorithm. The efficiency of the algorithm depends on the number of nodes used in the MiniGrid and total latency of the algorithm. With more nodes and greater communication between MiniGrid nodes, the total latency is higher, so it is necessary to minimize communication between nodes. On the other hand, with a larger number of nodes, the population that can withstand the algorithm will be larger. This provides a better solution to the problem with greater efficacy, because it is possible to explore and exploit a larger solution

space, increasing the possibility of finding the global optimum of the problem.

Genetic Algorithm in a Grid Environment

Figure 3 presents the genetic algorithm which contains a selection operator "The best" and a crossover operator "Crossover-k," in addition to the operator "Mutation." This mutation operator selects an individual to mutate, then randomly chooses between two mutation operators, "mutation-intelligence" and "mutation-restrictive." The Genetic works with multi populations generated at the beginning by the k-means technique (MacQueen, 1967), commonly used in data mining techniques to generate clusters. The genetic algorithm creates populations through k-means clustering where the individuals are feasible according to their geographical distribution. The operator "the best," selects the best individuals from each population according to their fitness percentage, and performs a crossover of these individuals. With the crossover-k, the size of each population is maintained, and exploration of the solution space is allowed. The iterative mutation operator improves the fitness function but also allows individuals to choose mutations that do not improve the fitness function; the mutation operator performs a local search to find a local optimum. The above procedure is repeated for each multi-population generation. The stop criterion for the algorithm is defined by the maximum number of generations G_n. The population size Pz is defined by the number of nodes present in the MiniGrid. $Pz = Pz_1+Pz_2+Pz_3+...+Pz_n$. It is Pz_i, where Pz_i is the population size present in each MiniGrid node and n is the number of nodes present in the MiniGrid.

The distributed processing applied to the algorithm is presented by a flowchart in Figure 4. It is noted that each MiniGrid node executes a genetic with a feasible population of size Pz_i. Each node independently performs the selection, cross-

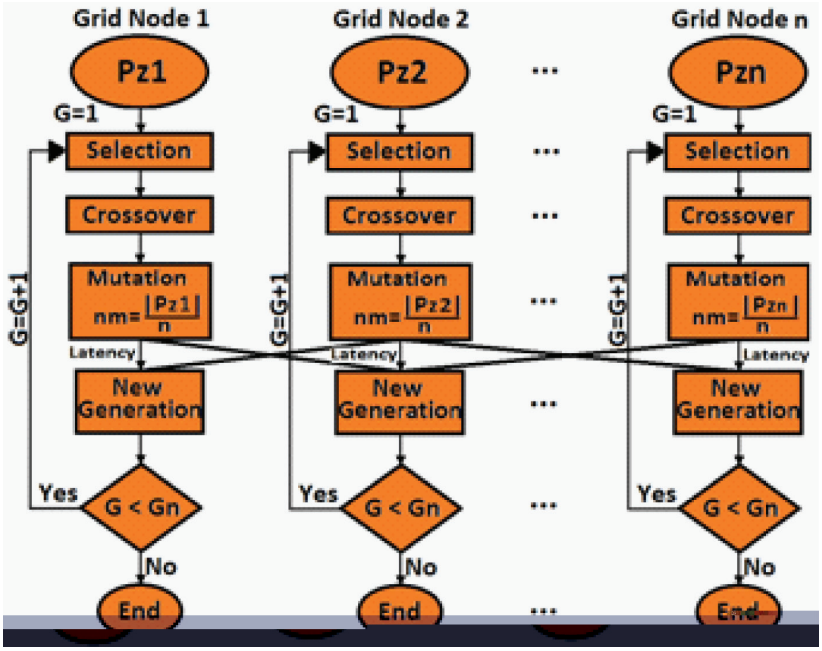
Figure 3. Multi population genetic algorithm

```
1  Multi_population=Init_Pop_k_MeansClustering (Pz);
2  G=0;
3    while(G != G_n)
4    {
5          aptitude_cost (Multi_population);
6          selection= TheBest (Multi_population);
7          Multi_population = Crossover_K(selection);
8          Multi_population = Mutation(Multi_population);
9          G=G+1;
10   }// end while
```

over and mutation. The portion that requires the cooperation of every process running a genetic is the mutation. First the population of each genetic is divided into *nm* sets of individuals, a randomly selected set of each population Pz_i is taken, and the mutation is applied to this set. The processes in each MiniGrid node communicate to exchange part of their population. The set *nm* of mutated individuals migrate to another population Pz_i. This is a representation of the reality that happens to immigrant groups, in this case there is migration from one population to another. In the communicative processes between MiniGrid nodes, a large latency is generated, which is higher when nodes are farther apart geographically. This can affect the efficiency of the genetic algorithm, so it is important to study this latency. The experimental results section presents an analysis of the latency generated by the genetic algorithm. Figure 4 is a description of the two mutation operators used in the genetic algorithm.

The process of selection and crossover of the genetic algorithm is very fast; it does not depend on the processes running on other MiniGrid nodes and thus requires a very short time. It defines an algorithm structure where mutations take place in the CPUs of the MiniGrid, which makes the algorithm work efficiently because it does not require communication between nodes in order to execute mutations, i.e., it does not affect the

Figure 4. Execution of a genetic algorithm for VRPTW using the MiniGrid UAEM-ITVER via Internet 2



latency in the process or later in the algorithm. The algorithm only requires communication once the mutation has taken place in MiniGrid nodes; this prevents latency from negatively influencing the execution of the algorithm. The genetic algorithm in Figure 4 can use all the resources of the MiniGrid independent of its size. If the MiniGrid has more high performance CPU resources it may lead to greater exploration and exploitation for the VRPTW with growth in polynomial time of the algorithm. This is because the population size *Pz* is directly proportional to the number of nodes used in the MiniGrid.

Restrictive Mutation Operator as a Model of Constraint Satisfaction

An adaptation of the Precedence Constraint Posting (PCP) method to the vehicle routing problem with time windows as a constraint satisfaction problem (CSP) using the VRPTW model can be made without taking into account the objective function. PCP was proposed by Cheng-Chung and Smith (1995, 1997). In Cruz-Chavez et al. (2007), the PCP method is proposed for the relaxed VRPTW model of constraint satisfaction where the size of each time window is infinite. The PCP method has been applied in a genetic algorithm for the VRPTW in Cruz-Chávez and Díaz-Parra (2010), only to build the initial population, but has not been implemented as part of an iterative mutation.

The restrictive mutation is a local search that involves the use of the shortest path that begins and ends in the depot and passes by all customers. It also requires finding the shortest path between pairs of nodes and the evaluation of each one of the cases that form the PCP procedure. To work with restrictive mutations, it is necessary to use an incidence matrix that represents the graph model of each individual of the VRPTW, as presented in Figure 2. The application of these techniques allows the evaluation of a solution that result in the constraint satisfaction VRPTW model. If this is the case, the individual (solution) is a feasible individual of the genetic population. To generate a

mutation in the feasible individual, a route change is made for one or more of the randomly chosen customers, to one or more routes selected by the PCP procedure which also evaluates the feasibility of the new solution. In this type of mutation, the quality of the new individual does not matter as long as it is feasible. This means that the restrictive mutation operator in the population accepts individuals who may have a lower fitness.

To evaluate an individual of the VRPTW using the PCP algorithm, the set Ω of client pairs (i, j) is defined, where i is a customer looking to switch to route k and j is a customer that belongs to route k. Then the shortest path (sp) algorithm for each pair (i, j) of the set Ω is applied. The values of sp are evaluated for cases of PCP in each pair (i, j) to set the direction and route for a pair of customers (i, j). It checks whether the new solution is feasible, i.e., if the solution meets the constraints in the mathematical model. If it is not a feasible solution, backtracking is performed, (Cormen et al., 2001), returning to the previous nodes and the PCP procedure is repeated.

To apply the mutation process to CSP as presented in Chung-Cheng and Smith (1997) and Christodoulou et al. (1994), the restrictive mutation is based on the PCP algorithm. To find a feasible solution to the VRPTW model, it is necessary to identify the tuple $\{V, D, C\}$. The set V is formed by *Orderingij* variables; each pair of customers (i, j) defines one of these variables. The set D is the set of values for the variables in V, each *Orderingij* can take two possible values, or. That is, customer i is visited followed by j, or customer j is visited followed by i (a precedence is defined among customers i and j). The set C of constraints on V is given for each pair (i, j) representing a variable *Orderingij*. This restricts the feasible values for the distance between a pair (i, j). PCP defines the set of restrictions based on four cases which are explained in the PCP algorithm. The basic search procedure for the VRPTW as a CSP is composed of six steps which are presented for various applications in Chung-Cheng and Smith (1995, 1997) and Cruz-Chavez et al. (2007).

Step 1. Apply constraints propagation in order to establish the current set *VD* of values for each *Orderingij* variable not assigned. *VD* is obtained by PCP.
Step 2. If *VD = 0* for any *Orderingij* variable, backtrack.
Step 3. If there are not variables without assignment or if the assignment is not consistent for all the variables end. Otherwise,
Step 4. Select an *Orderingij* variable not assigned.
Step 5. Select a value of *VD* in order to assign it an *Orderingij* variable.
Step 6. Go to step 1.

PCP Algorithm

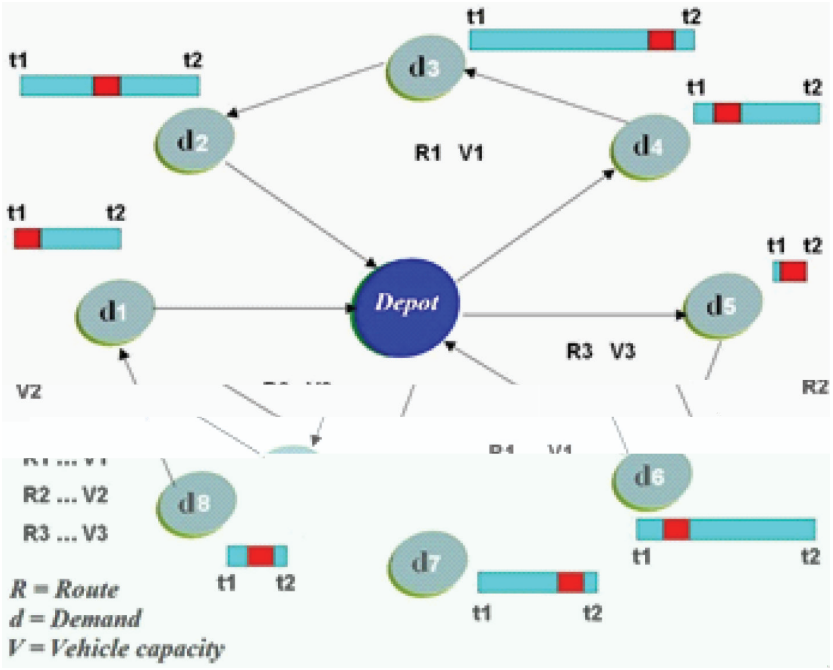
Within the search procedure, PCP builds the solution through Depth First using partial assignments of Ω. The PCP algorithm carries out a pruning of the search space early on and provides a heuristic for the value assignment of the *Orderingij* variables.

PCP consists of a series of cases in which it is necessary to prove whether the shortest path sp between a pair of nodes (i, j) that represents the *Orderingij* variable has a value that fulfills some of the PCP cases. According to the result obtained upon evaluating the shortest path, the value of *Orderingij* is designated. The evaluation of sp is calculated from i to j (sp_{ij}) and from j to i (sp_{ji}).

For the Restrictive mutation, the PCP algorithm is applied to a partial solution of the VRPTW. PCP obtains a new feasible solution by assigning one or more clients to the same routes in a different sequence, or to other routes. The PCP algorithm applied to the VRPTW for the CSP consists of the four steps that are presented below:

Step 1.- Find the shortest path for each unordered pair of nodes sp_{ij} and sp_{ji}.
Step 2.- Classify the decision of ordination of the pairs unordered with four cases

Figure 5. Partial solution with customer d7 without route assignment



Case 1. If $sp_{ij} >= 0$ and $sp_{ji} < 0$ then $O_i \prec O_j$ should be selected.

Case 2. If $sp_{ji} >= 0$ and $sp_{ij} < 0$, then $O_j \prec O_i$ should be selected.

Case 3. If $sp_{ji} < 0$ and $sp_{ij} < 0$, then the partial solution is inconsistent.

Case 4. If $sp_{ji} >= 0$ and $sp_{ij} >= 0$, then no relationship of order is possible

Step 3.- Existence of cases Does either case 1 or case 2 exist?

If one exists, go to step 4. If neither exists, go to step 1

Step 4.- Fix new precedence for unordered pairs.

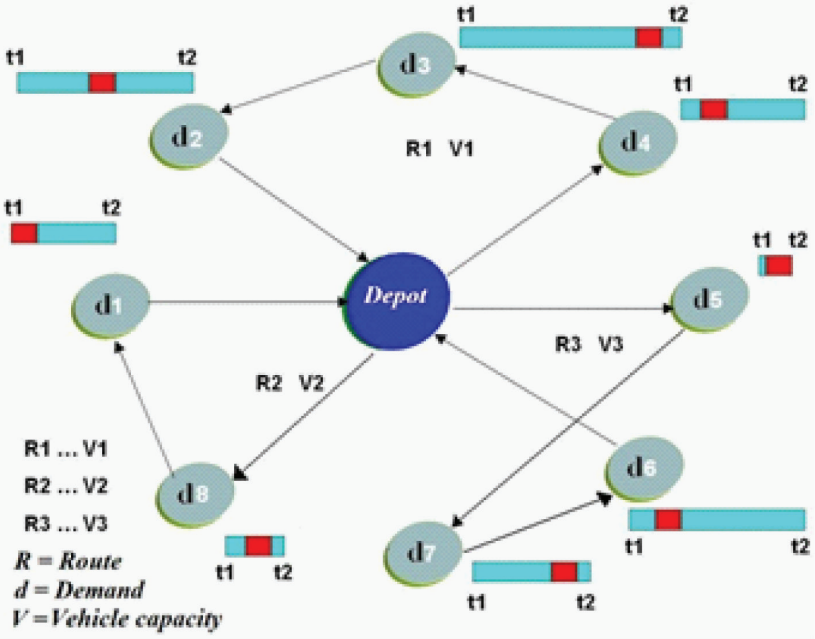
To better understand the PCP algorithm, an example of the 8 clients instance is presented in Figure 1. From the solution presented in Figure 2, the customer d7 is randomly selected to change course or sequence. It is removed from the path to which it belongs (Figure 5). The PCP algorithm is applied to find a route, and a feasible solution is generated. The new solution is presented in Figure 6. In this figure, we see that the client d7 becomes part of the route R3.

Intelligent Mutation Operator

The intelligent mutation operator proposed in Díaz-Parra and Cruz-Chavez (2008), takes a mutation customer candidate (gene-candidate) from the individual that minimizes the objective function presented in (1). Unlike the restrictive mutation that does not pay attention to the quality of the individual, the intelligent mutation operator always tries to improve the quality of the individual.

The procedure starts when the intelligent mutation operator searches for the gene (client) with the greatest distance from another gene, called the gene-candidate, and finds the gene that produces this distance, called gene-mutation-1. The gene-mutation-2 should be the one with the lowest

Figure 6. New solution with restrictive mutation



distance from the gene-candidate. Once they are identified, the pair of genes is swapped; the change is made only if the change does not violate time window and capacity constraints in the vehicle. In case of violation of any restriction, the process starts again. With these iterative gene mutations in an individual, the intelligent mutation operator minimizes the objective function of the VRPTW.

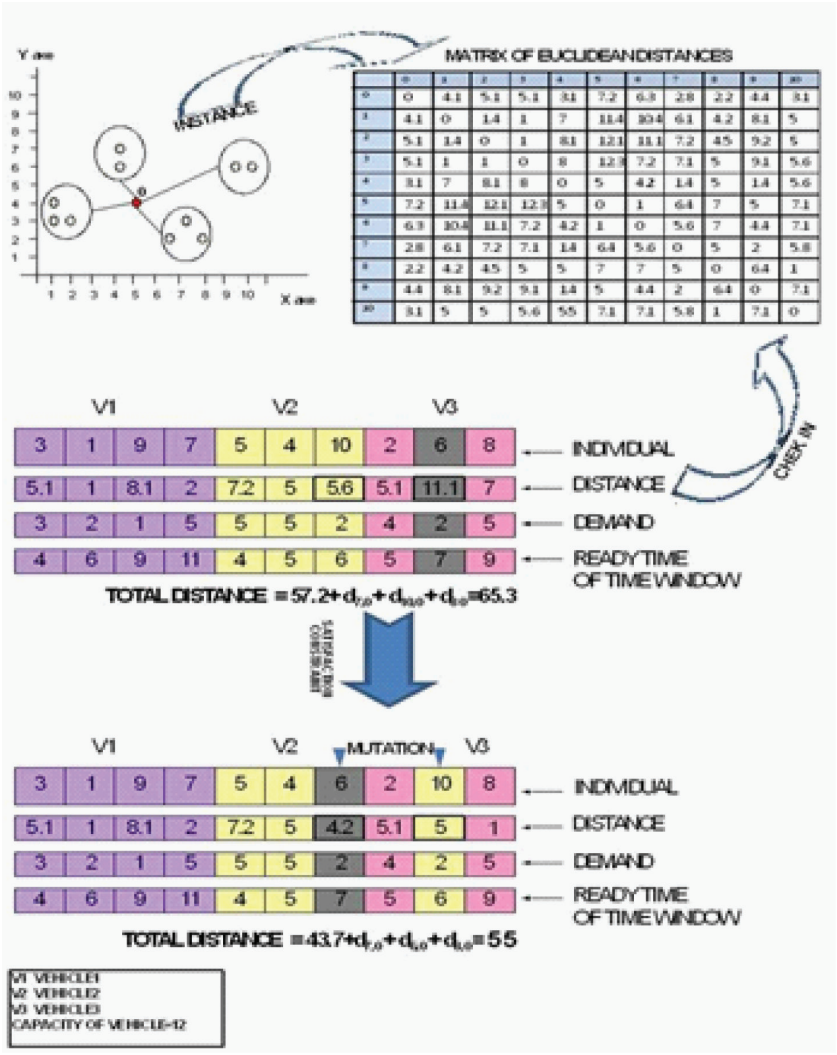
Figure 7 shows an example of a case in which the intelligent mutation operator is applied for an individual made up of 10 genes. The value of the objective function of the individual is 65.3. In the beginning of the procedure, the matrix of Euclidean distances from the gene is used to find the greatest distance. In this case gene-mutation-1 = 6 and gene-candidate = 2 are farthest apart; the distance from gene 2 to gene 6 is 11.1. In this case, the two genes are present in route V3. Next, the Euclidean distance matrix for the entire individual is searched to find the genes with a distance of less than 11.1 from the gene-candidate. The gene that has the shortest distance from the gene-candidate is the gene-mutation-2 = 10 which has a distance of 5 from the gene-candidate. In this case the gene is present in route V2. Because this distance of 5 is less than 11.1, a permutation between gene-mutation-1 and gene-mutation-2 is performed. Figure 2 presents the new mutated individual. It verifies that demand does not exceed the vehicle capacity, which is 12. It also verifies that the ready time of service of each gene (customer) is within the time window for each client. Finally, the value of the objective function of the individual mutation is 55, which improves the value of the objective function of the individual before mutation.

EXPERIMENTAL MINIGRID INFRASTRUCTURE

The experimental technique used to connect computing infrastructure performance through a virtual private network is based on software; this is the way to create the MiniGrid.

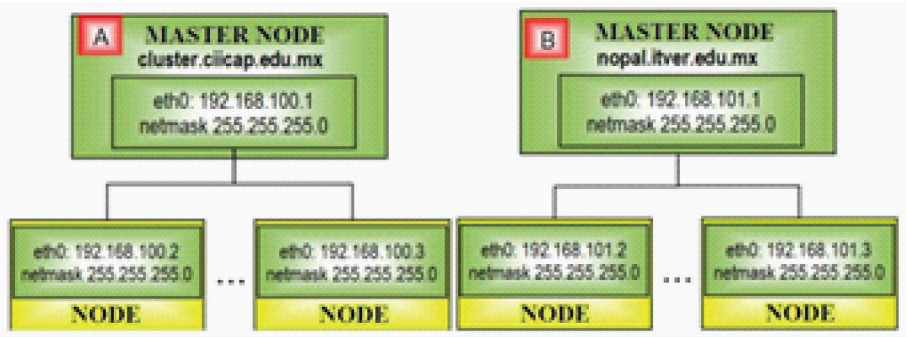
Distributed systems provide mechanisms to help make it easier to manage databases,

Figure 7. Intelligent mutation operator. Taken from Díaz-Parra and Cruz-Chávez (2008)



Web systems, email, and phones, among other services. These distributed systems have been developed under the client-server scheme. Over many decades, great progress has been made with respect to the integration of resources, but with an approach based on local networks where they reach the lowest latencies, leaving only the Internet to consult their data, which does not require large bandwidths. Distributed systems are defined as a coordinated, transparent and secure way to share information resources across geographically distributed sites. Recently, distributed computing has given way to Grid computing, more specifically to the Computational Grid. Foster and Kesselman (2004) present everything related to the Grid phenomenon, mainly due to an increase in connection capacity of the Internet and private networks of wide coverage. The Grid takes these clusters and super computers to the next level by making a connection between localized clusters in different geographical areas, thus achieving collaborative resource sharing between institutions, companies and others.

Figure 8. Two geographically separated clusters on two different subnets A and B



The Grid is an infrastructure for sharing computing resources (numeric processing and storage), using the Internet as a medium. The Grid is not simply communication between cluster computers; its goals are much more ambitious.

Currently in Mexico there are joint efforts to develop and implement a Grid National Laboratory. Corporación Universitaria para el Desarrollo de Internet 2 (CUDI, 2011), is one of the main promoters of these efforts. CUDI is the agency managing the Internet 2 project in Mexico. Internet 2 is a high-speed academic network connecting several universities in Mexico, with interconnection to high-speed university networks in the United States and Canada.

The Grid is developed in accordance to standard protocols and reference platforms based on open source for its development. Widely used by researchers and research centers, grid computing is rapidly emerging as the means used by corporate companies to collaborate, share data and software, store more information than in existing networks, and access large amounts of processing power without investing significant sums on expensive computers.

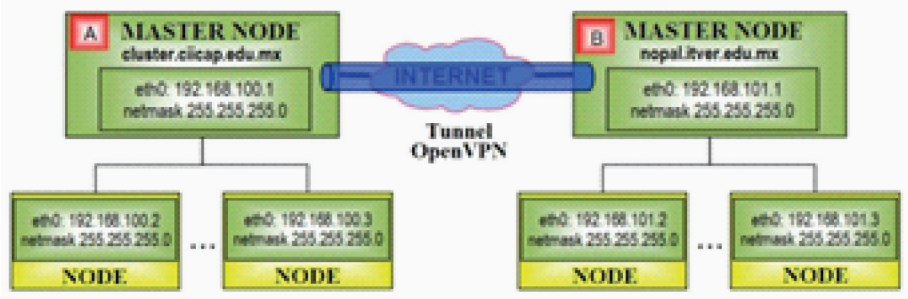
The experimental MiniGrid developed for the implementation of genetic algorithms to study the VRPTW consists of two high-performance clusters, which are geographically distant but are united through a Virtual Private Network network-to-network using OpenVPN (software) instead using routers (hardware). In principle, clusters contain a different subnet. The cluster CIICAp located at the Autonomous University of Morelos State (UAEM) was configured as a subnet 192.168.100.0, the subnet that is identified as Cluster A. The cluster NOPAL located at the Technological Institute of Veracruz (ITVER) was configured as a subnet 192.168.101.0, the subnet that is identified as B (Figure 8).

Configuring the OpenVPN Virtual Private Network

OpenVPN is a connectivity solution based on software that provides point to point connectivity via the Internet to validate users and hosts connected remotely. OpenVPN is an implementation of software used to build a virtual private network via routers without hardware; it supports various authentication means such as certificates, usernames and passwords. OpenVPN configuration can be summarized in three types:

- **Machine to Machine.** It is the simplest method for encrypting communications between two computers which can send packets directly to each other either because they are connected to the same local network or because both are connected to the Internet and are accessible to each other.

Figure 9. Two geographically separated clusters in different sub networks A and B linked via VPN



- **Road Warrior.** This is one of the most used and requested ways to connect to a network. It allows a machine outside the local network to communicate with the OpenVPN server and once authenticated, to access local network resources.
- **Network to Network.** Through this type of configuration, two separate and geographically distant networks can come together and share each other's resources. Communication between the two networks is encrypted outside the OpenVPN servers until it reaches the other end (destination).

The idea for these configurations is based on being able to use a single channel of communication, called tunnel (Figure 9), to send all communications back and forth from one extreme to another, which allows one to see all the machines connected, as if they were physically attached via cables. It follows that, to join two clusters, it is necessary to join two OpenVPN configuring networks, using Internet 2 as the media.

To configure a network to network connection it is necessary to define who will be the server and who will be the client. Here the cluster ciicap.edu.mx is the server and nopal.itver.edu.mx is the client. The only two machines that are connected are the master nodes from each cluster. The implementation of routing packets is required to achieve the master nodes that are at the ends of the VPN.

The infrastructure deployed in the high-performance clusters of the ITVER and UAEM that comprise the MiniGrid is made with the same software, but not with the same hardware. The result is two high-performance heterogeneous clusters, due to differences in processors, memory and communication equipment, both in the Master nodes and in the processing nodes. Thus, the first overview of this infrastructure is that each entity has its own HPC Cluster. The computer hardware and software that integrate MiniGrid are specified in Tables 1 and 2. Table 1 presents the hardware and software used by the CIICAp cluster at UAEM. Table 2 presents the hardware and software used by the NOPAL cluster at ITVER.

EXPERIMENTAL TESTS

The results of the genetic algorithm with message passing (MPI) in the VRPTW model are presented, as implemented using the experimental MiniGrid. Evidence of latency and bandwidth between clusters of the MiniGrid with Internet 2 communication is also reported.

A Measure of Bandwidth on the Grid

The bandwidth in the MiniGrid was calculated, this covers only the master nodes between the points cluster.ciicap.edu.mx and nopal.itver.edu.mx. Because traffic on the network directly

Table 1. Hardware and software infrastructure of the CIICAp cluster at UAEM

ELEMENT	HARDWARE	SOFTWARE
Communication	Switch Cisco C2960 24/10/100 Internal Cable level 5	S.O. Red Hat Enterprise Linux 4 gcc compiler version 3.4.3 OpenMPI 1.2.8 MPICH2-1.0.8 Ganglia 3.0.6 NIS ypserv-2.13-5 NFS nfs-utils-1.0.6-46 OpenVPN
Master node	Pentium 4, 2793 MHz 512 MB RAM 80 GB Hard disk 2 cards 10/100 Mb/s	
Processing node 01	Pentium 2, 266 Mhz 256 MB RAM 4 GB Hard disk 1 card 10/100 Mb/s	
Processing node 02	Pentium 2, 133 Mhz 128 MB RAM 4 GB Hard disk 1 card 10/100 Mb/s	
Processing node 03 Processing node 04 Processing node 05	Intel® Pentium® Dual Core, 2800 MHz, 1GB RAM, 160GB Hard disk, 1 card 10/100 Mb/s	
Processing node 06 Processing node 07 Processing node 08	Intel® Celeron® Dual Core, 2000MHz, 2 GB RAM, 160GB Hard disk, 1 card 10/100 Mb/s	

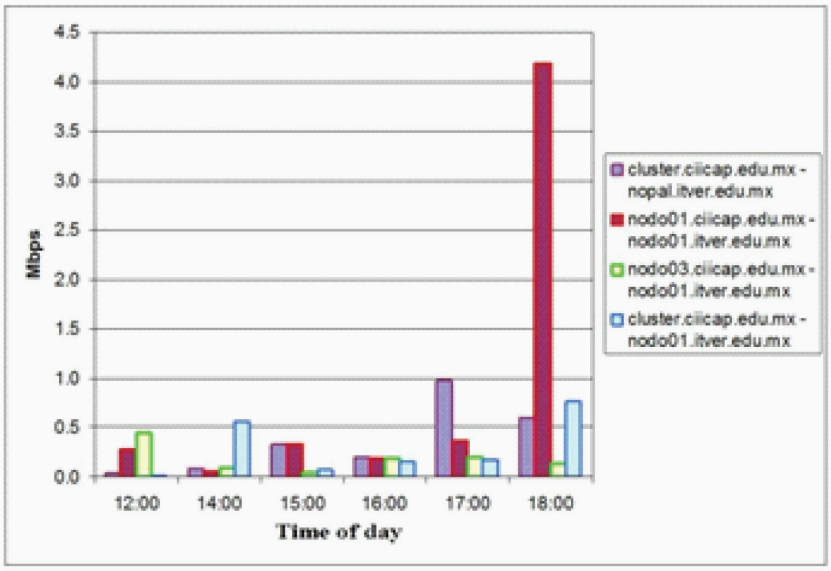
Table 2. Hardware and software infrastructure of the NOPAL cluster at ITVER

ELEMENT	HARDWARE	SOFTWARE
Communication	Switch 3com 8/10/100 Internal Cable level 5	S.O. Red Hat Enterprise Linux 4 Compilador gcc versión 3.4.3 OpenMPI 1.2.8 MPICH2-1.0.8 Ganglia 3.0.6 NIS ypserv-2.13-5 NFS nfs-utils-1.0.6-46 OpenVPN
Master node	Pentium 4, 2394 Mhz 512 GB RAM 60 GB Hard disk 2 cards 10/100 Mb/s	
Processing node 01	Pentium 4 Dual Core, 3200 Mhz 1 GB RAM 80 GB Hard disk 1 card 10/100 Mb/s	
Processing node 02	Pentium 4 Dual Core, 3201 Mhz 1 GB RAM 80 GB Hard disk 1 card 10/100 Mb/s	

affects communications, several measurements were made per day, and these tests were repeated for three days.

Figure 10 shows the experimental measurement of the bandwidth between the CIICAp and NOPAL clusters. Specifically, the measurement is made between various pairs of nodes, one corresponding to the CIICAp cluster and the other corresponding to NOPAL cluster, as shown in Figure 10. It can be seen that the bandwidth taken at different times on the first day of testing varies between 0.1 and 1 Mbps, with reference to several pairs of nodes in the experimental MiniGrid. Most measurements are between 0.1 and 0.5 Mbps. Figure 10 shows a peak between 4 and 4.5 Mbps at 18 hours. One can see that the bandwidth for

Figure 10. Bandwidth on March 23 (Mbps), monitored on Day 1



any pair of nodes is never stable, and is well below the reported maximum bandwidth of 16 Mbps of Internet 2 for connection of the MiniGrid. An increase in the bandwidth can be observed as the day progresses, this may be because there is less use of Internet 2 in the two institutions (UAEM-ITVER) in the afternoon.

Figure 11 shows the experimental measurement of the bandwidth between the CIICAp and NOPAL clusters with Internet 2 connection. The measurement is made between various pairs of nodes, one for the CIICAp cluster and another for the NOPAL cluster, as shown in Figure 11. It can be seen that the bandwidth taken at different times during the second day of testing varies between 0.1 and 0.4 Mbps, with reference to several pairs of nodes in the experimental MiniGrid. The figure shows a peak between 1.2 and 1.4 Mbps at 18 hours. It can also be seen that the bandwidth in any pair of nodes is never stable, and is well below the reported maximum bandwidth of 16 Mbps of Internet 2 for connection of the MiniGrid. A decrease in bandwidth can be observed from 12 to 14 hours, and an increase observed between 14 and 18 hrs. This may be due to the use of Internet 2, which may have had more traffic that day from 13 to 15 hours.

Figure 12 shows the experimental measurement of the bandwidth between the NOPAL and CIICAp clusters with an Internet 2 connection. Specifically, the measurement is made between various pairs of nodes, one corresponding to the CIICAp cluster and the other corresponding to the NOPAL cluster, as shown in Figure 12. It can be seen that the bandwidth taken at different times on the third day of testing varies between 0.1 and 0.5 Mbps, with reference to several pairs of nodes in the experimental MiniGrid. The figure shows a peak between 3 and 3.5 Mbps at 14 hrs. It can also be seen that the bandwidth in any pair of nodes never is stable, and is well below the reported maximum bandwidth of 16 Mbps of Internet 2 for connection of the MiniGrid. It can be observed that the width of the band has a tendency to increase as the day passes. This may be because of less network traffic on Internet 2 in the two institutions (UAEM-ITVER) in the afternoon.

Comparing the three measured days shown in Figures 10, 11, and 12, it can be seen that the

Figure 11. Bandwidth (Mbps), monitored on Day 2

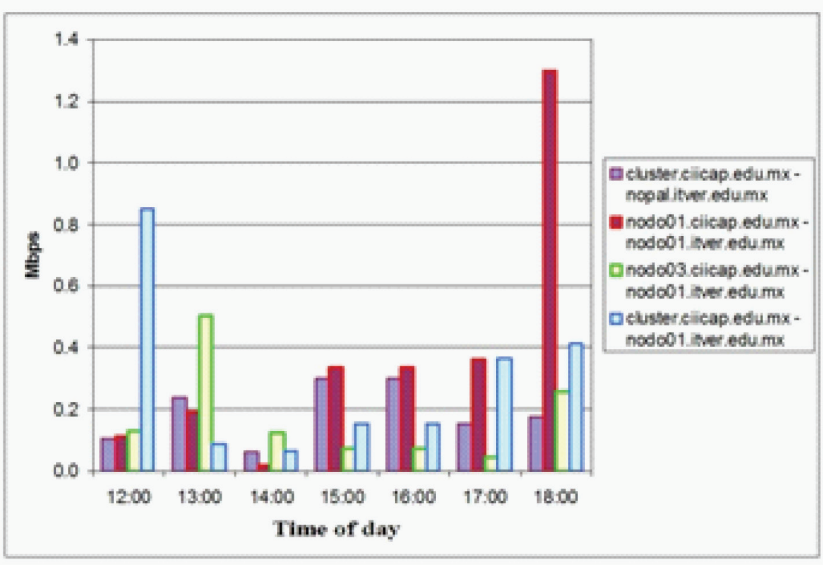
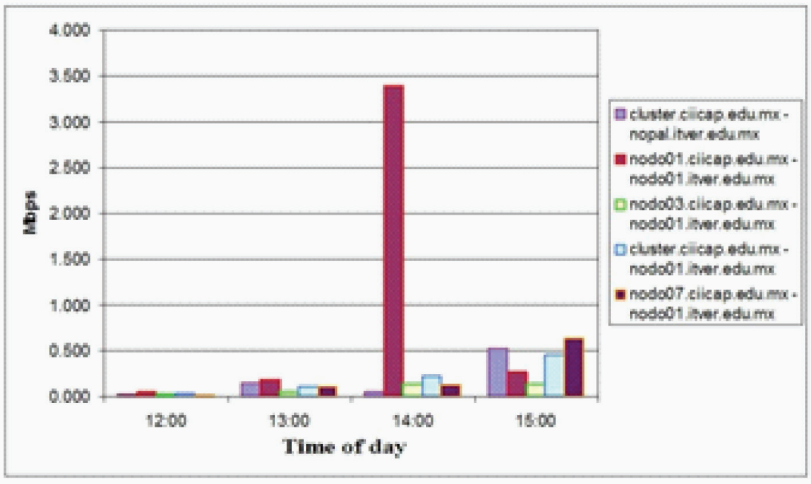


Figure 12. Bandwidth (Mbps), monitored on Day 3



major peaks of bandwidth occur during the afternoon, starting around 14 hrs. It can also be seen that the measures of bandwidth are smaller in the morning, before 14 hrs.

Figure 13 presents the average bandwidth on the first day of testing between pairs of nodes. One can see that the pair of nodes that has the lowest average bandwidth is the pair nodo03.ciicap.edu.mx - nodo01.itver.edu.mx, and the pair of nodes that has the highest average bandwidth is the pair nodo01.ciicap.edu.mx - nodo01.itver.edu.mx. The bandwidth ranges from 0.2 to 0.9 Mbps.

Figure 14 presents the average bandwidth on the second day of testing between pairs of nodes. One can see that the pair of nodes that has the lowest average bandwidth is the pair of master nodes cluster.ciicap.edu.mx - nopal.itver.edu.mx and the pair of nodes that has the highest average

Figure 13. Average bandwidth (Mbps), monitored on Day 1

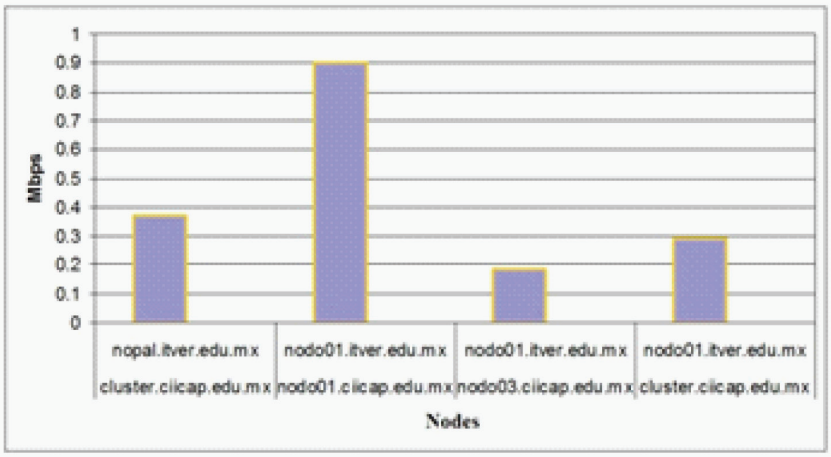
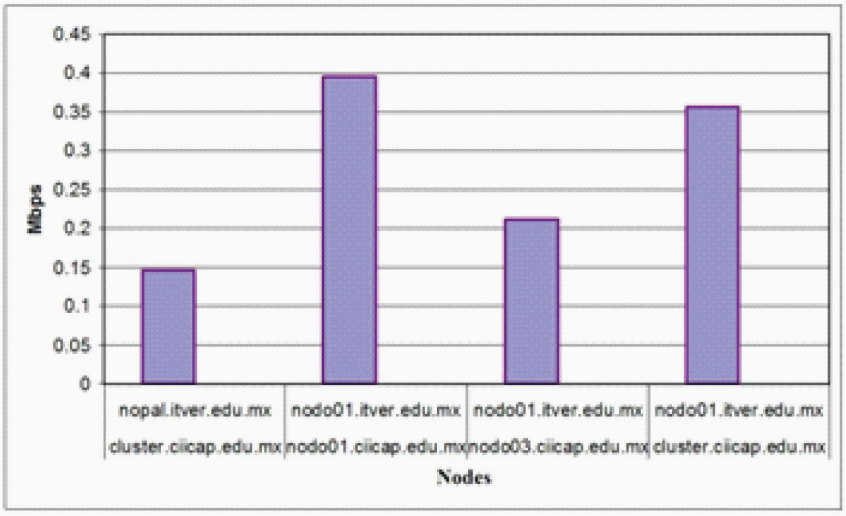


Figure 14. Average bandwidth (Mbps), monitored on Day 2



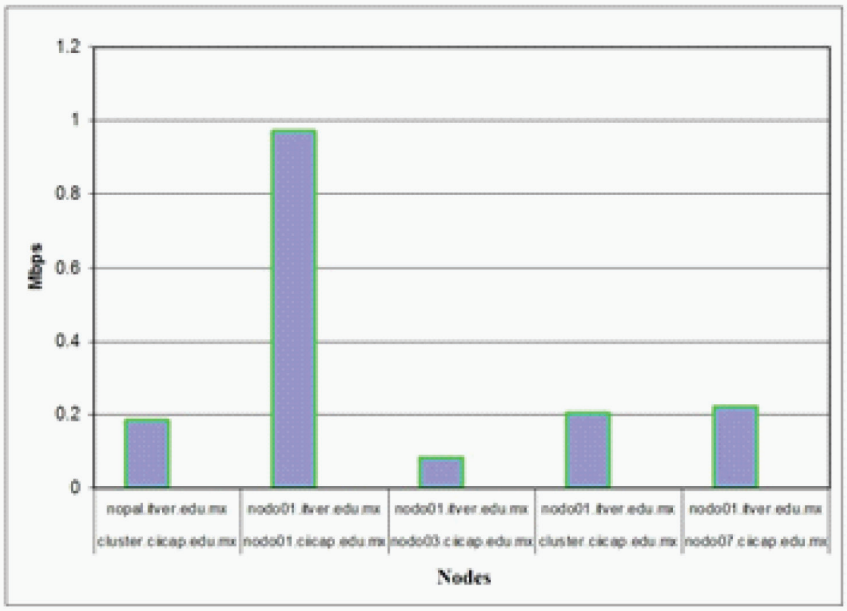
bandwidth is the pair nodo01.ciicap.edu.mx - nodo01.itver.edu.mx. The bandwidth ranges from 0.15 to 0.4 Mbps.

Figure 15 presents the average bandwidth on the third day of testing between pairs of nodes. One can see that the pair of nodes that has the lowest average bandwidth is the pair of nodes nodo03.ciicap.edu.mx - nodo01.itver.edu.mx and the pair of nodes that has the highest average bandwidth is nodo01.ciicap.edu.mx torque - nodo01.itver.edu.mx. The bandwidth ranges from 0.05 to 0.98 Mbps.

It can be observed in Figures 13, 14, and 15 that the pair of nodes nodo01.ciicap.edu.mx - nodo01.itver.edu.mx throughout the three days, consistently had a higher bandwidth and the pair nodo03.ciicap.edu.mx - nodo01.itver.edu.mx had the lowest bandwidth in two of the three days of testing.

Latency measurement in the Grid. The calculation of latency is based on transfer rates, which

Figure 15. Average bandwidth (Mbps), monitored on Day 3



adjusts to the speed of the VPN between the nopal.itver.edu.mx and cluster.ciicap.edu.mx points, so the latency between nodes of clusters are consistent with the latencies of the master nodes. To test the genetic algorithm for VRPTW with the problem of 100 clients, only 4 individuals are sent by each process. According to the type of data used in the algorithm, which is a 4-byte integer, the flow of data sent in each process depends on the size of the structure. In this case, the size of the structure that represents an individual is 12,800 bytes, so each CPU of the CIICAp cluster will be sent a flow of 51.200 bytes. The calculation of latency in the experimental MiniGrid involved launching the algorithm from the CIICAp cluster, and required the transmission via Internet 2 of a stream of data to the NOPAL cluster of 204,800 bytes (200kb), during each iteration of the genetic algorithm. According to this data stream, the latency is calculated in the MiniGrid and is presented in Figures 16 through 19.

Figure 16 presents experimental proofs of the latency obtained on the first day of testing from 12 to 18 hours. It is observed that the latency decreases as the day progresses. It starts at 12 hours with higher latency measured between 40 and 45 sec, and ends at 18 hours with the lowest latency measure of less than 0.4 sec. It can be seen that for each pair of nodes, the latency generally tends to decrease as the day progresses. This is consistent with what was observed in the figures of bandwidth, apparently the use of Internet 2 decreases in the afternoon, causing the transfer of data to be more efficient.

Figure 17 presents experimental proof of the latency obtained on the second day of testing from 12 to 18 hours. It is observed that the latency behavior neither increases nor decreases as the day progresses, not at 14 hours or at 16 hours, which is when most latency exists in all pairs of nodes on average. The increased latency is measured at 16 hours for most pairs of nodes, this being between 60 and 70 sec. It has the lowest latency at 18 hours for all pairs of nodes. As in Figure 16, the lowest latency occurs at 18 hours.

Figure 18 presents the experimental proof of latency obtained on the third day of testing from 12 to 18 hours. It is observed that the latency

Figure 16. Latency monitored on the first day of testing

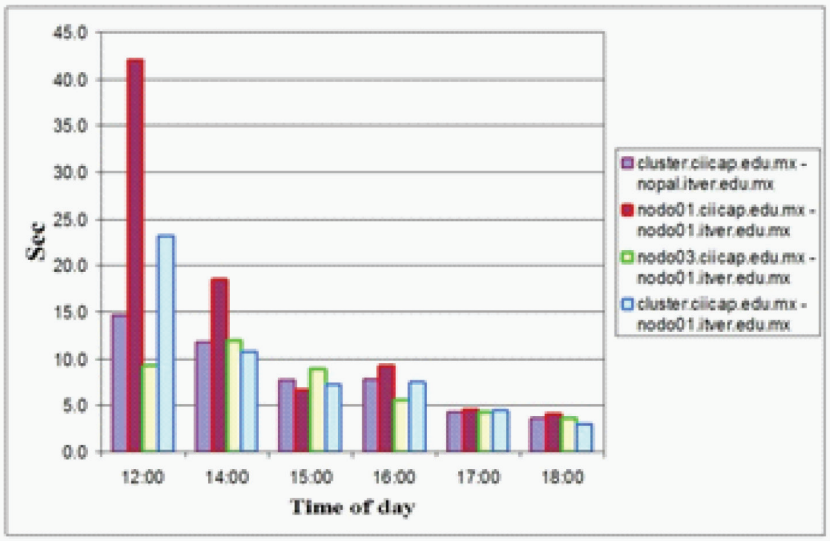
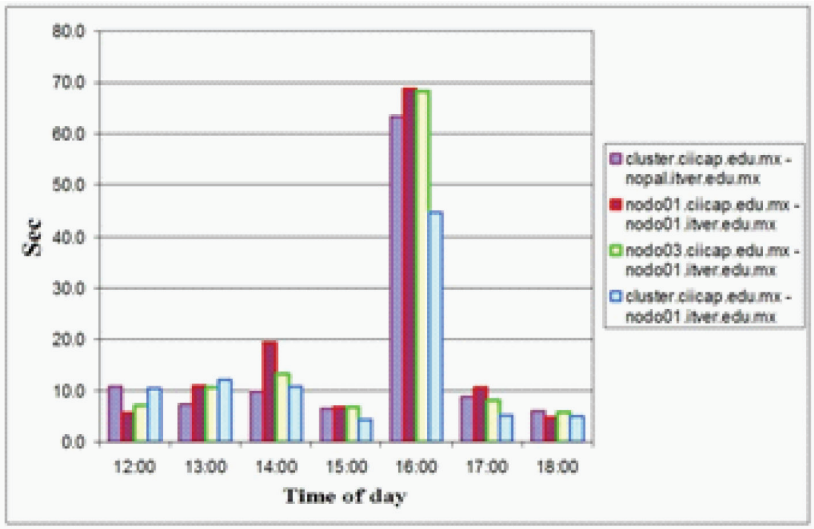


Figure 17. Latency monitored on the second day of testing



decreases as the day progresses. It starts at 12 hours with higher latency measuring between 70 and 75 sec and ends at 18 hours with the lowest latency measuring less than 0.8 sec. It can be seen that in each pair of nodes, latency generally tends to decrease as the day progresses. This is consistent with what was observed in the figures of bandwidth, apparently the use of Internet 2 decreases in the afternoon, causing the transfer of data to be more efficient.

Figure 19 presents the average latency of the first day of testing between pairs of nodes. One can see that the pair of nodes that has the lowest average latency is the pair nodo03.ciicap.edu.mx - nodo01.itver.edu.mx and the pair of nodes that has the highest average latency is the pair nodo01.

Figure 18. Latency monitored on the third day of testing

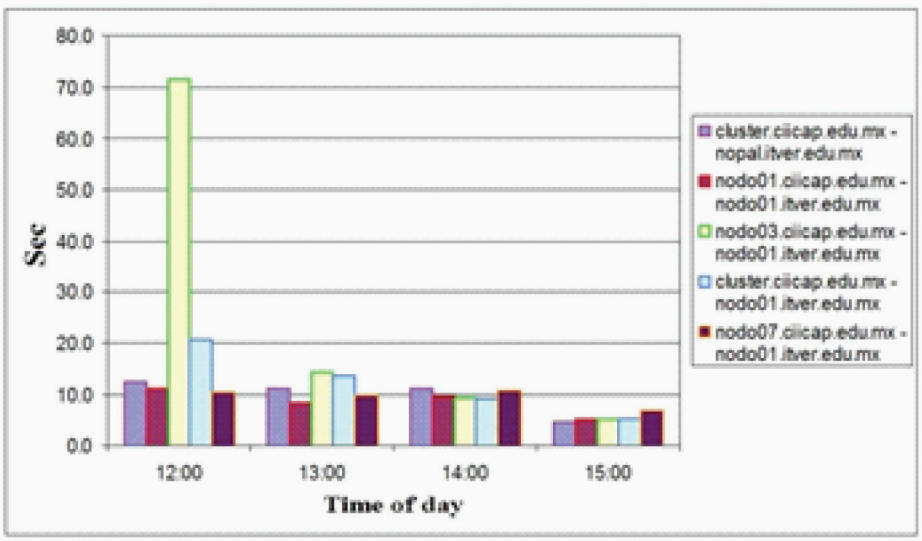
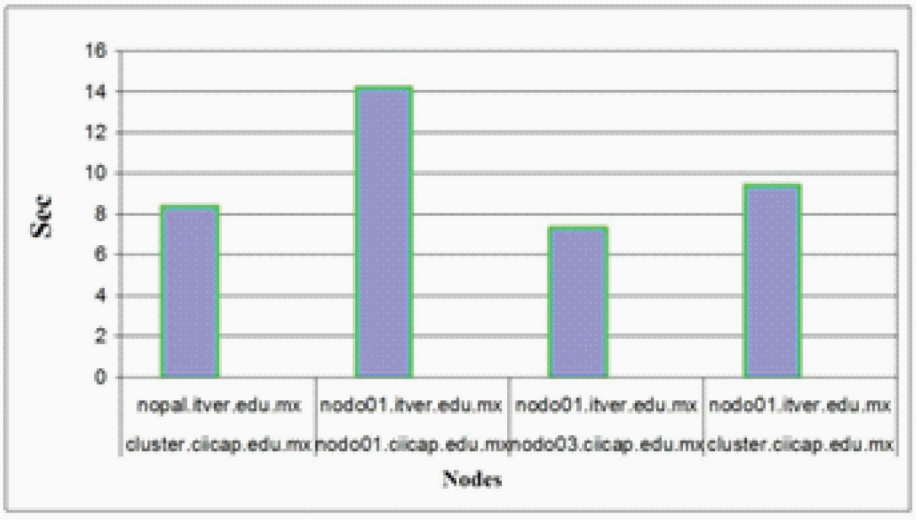


Figure 19. Average Latency (sec), monitored on the first day of testing



ciicap.edu.mx - nodo01.itver.edu.mx. The latency on the first day is between 6 and 14 sec.

Figure 20 presents the average latency of the second day of testing between pairs of nodes. It can be seen that the pair of nodes that has the lowest average latency is the pair cluster.ciicap.edu.mx - nodo01.itver.edu.mx and the pair of nodes that has the highest average latency is the pair nodo01.ciicap.edu.mx - nodo01.itver.edu.mx. The latency on the second day is between 12 and 18 sec.

Figure 21 presents the average latency of the third day of testing between pairs of nodes. It can be see that the pair of nodes that has the lowest average latency is the pair (nodo01.ciicap.edu.mx - nodo01.itver.edu.mx) and the pair of nodes

Figure 20. Average Latency (sec), monitored on the second day of testing

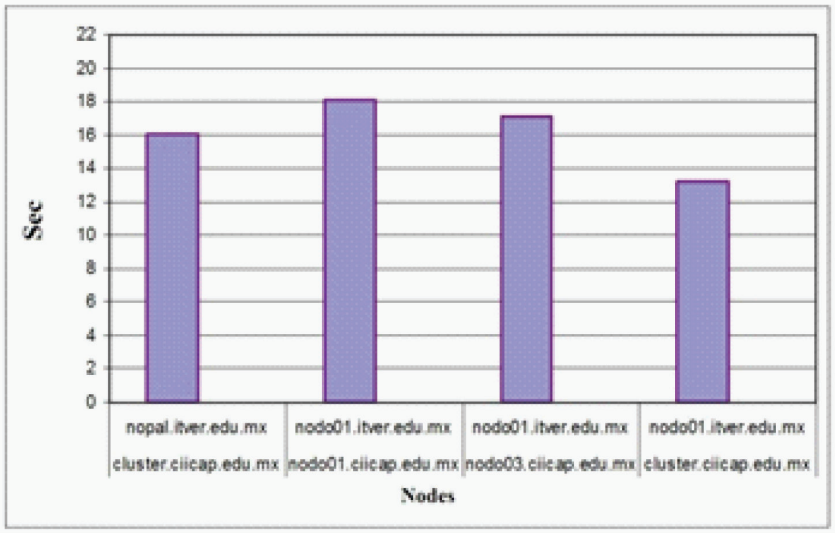
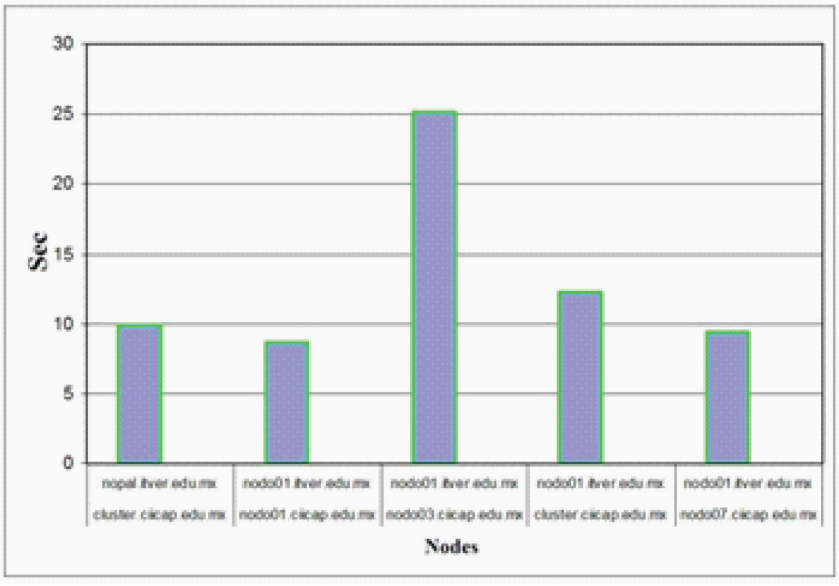


Figure 21. Average Latency (sec), monitored on the third day of testing



that has the highest average latency is the pair nodo03.ciicap.edu.mx - nodo01.itver.edu.mx. The latency of the third day is between 8 and 25 sec.

It can be seen in Figures 19, 20 and 21, that the pair of nodes nodo01.ciicap.edu.mx - nodo01. itver.edu.mx, during two of the three days, had the highest latency. With respect to lower latency, there were no distinctive pairs of nodes.

There was also a measurement recorded of the time required to answer when sending a file of 200kb from node to master node CIICAp and NOPAL (frontends), and back. The completion time for writing the file was also recorded. These

Figure 22. Latency (sec), monitored on the first day of testing

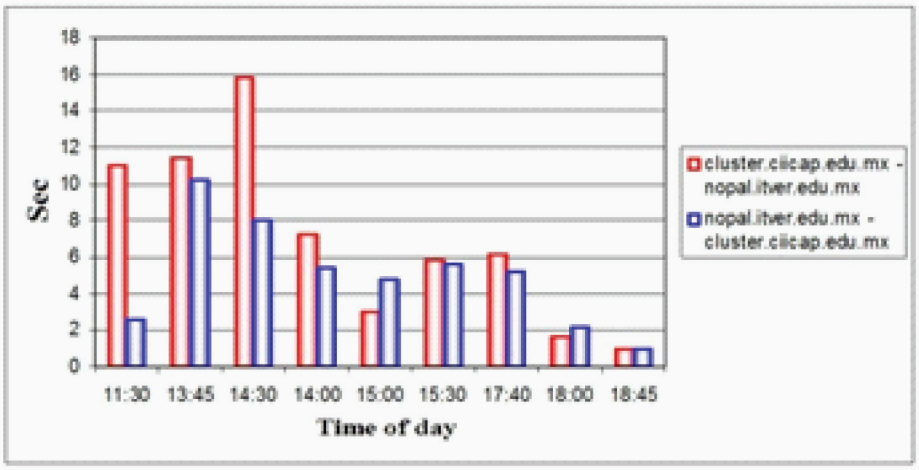
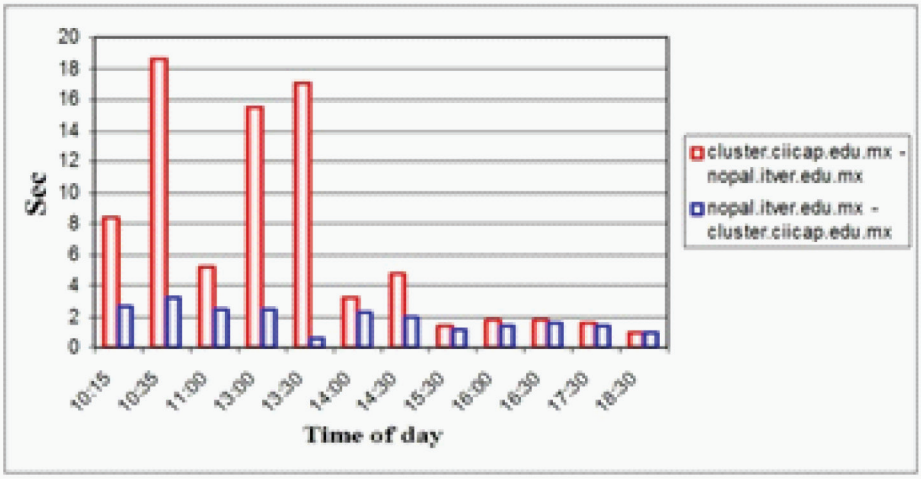


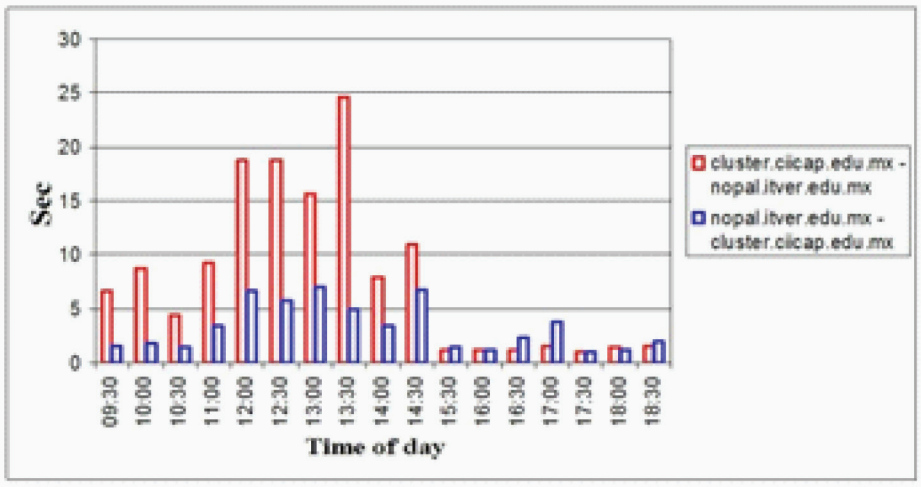
Figure 23. Latency (sec), monitored on the second day of testing



measurements were performed for three consecutive days between 9 and 18 hours. Figures 22 through 24 shows the time required to transfer a file of 200kb. The same behavior can be observed in all the figures. Latency is the largest from the CIICAp cluster to the NOPAL cluster. Latency is more perceptible in the morning and decreases in the afternoon.

To take advantage of the resources of the experimental MiniGrid and accelerate the multi-population evolutionary algorithm, reducing latency was attempted. A mutation was applied to half of the population Pz (for a pair of nodes in the MiniGrid), then the exchange of the mutated population was performed with an additional node. The exchange took place by sending the information via a temporary file with the half of the population missing from the Grid node. Once the population was complete, the fitness evaluation process of each individual was performed and a new generation was started.

Figure 24. Latency (sec), monitored on the third day of testing



The parallel genetic algorithm tests were performed as follows:

- Between nopal.itver.edu.mx cluster nodes (10 Mbs Ethernet communication).
- Between cluster.ciicap.edu.mx cluster nodes (100 Mbs Ethernet communication).
- Between the frontend of the clusters cluster.ciicap.edu.mx and nopal.itver.edu.mx (Internet 2 communication).

The two instances used to measure the efficiency of the parallelized algorithm were of 100 customers, C102-100 and C103-100, taken from the set of 56 test problems proposed by Solomon. These problems can be found in Gehring and Homberger (1999). These problems have a common number of customers and demands. Each customer always makes a demand and the maximum number of vehicles that can be used is 25. In addition to the topological distribution, the differences are given by the demand quantities and different time windows. To test the genetic algorithm, a population of 1000 individuals and 20 generations was used.

Different numbers of nodes were used, as they were available on each platform:

Figure 25 presents the speedup measurement obtained for the two benchmarks used. The ITVER cluster is very limited, mainly because there are few processors and the internal interconnection network of the cluster is 10 Mbps.

In the CIICAp cluster, tests were performed with 8 processors. As shown in Figure 26, the scalability is good and the behavior is very close to ideal. This is because the machines have a great amount of RAM (2 GB per node) and there is a 1 Gbs network, resulting in better use.

In the MiniGrid, tests were performed with the benchmark C102 for 2, 4, and 8 processors. These runs sought the same number of processor cores in both clusters so that the system was balanced.

In Figure 27 it can be seen that the behavior is similar to that observed in the CIICAp cluster presented in Figure 26, which means that good results using the MiniGrid are obtained when data traffic is not too intense.

Figure 28 shows the average efficiencies obtained in all test configurations used in relation to Speedup for the two problems C102 and C103 of VRPTW. This figure shows that the efficiency of the MiniGrid is above 70% when using 8 processors. This efficiency is higher than that of the

Figure 25. Speedup of testing C102 and C103 in the NOPAL cluster

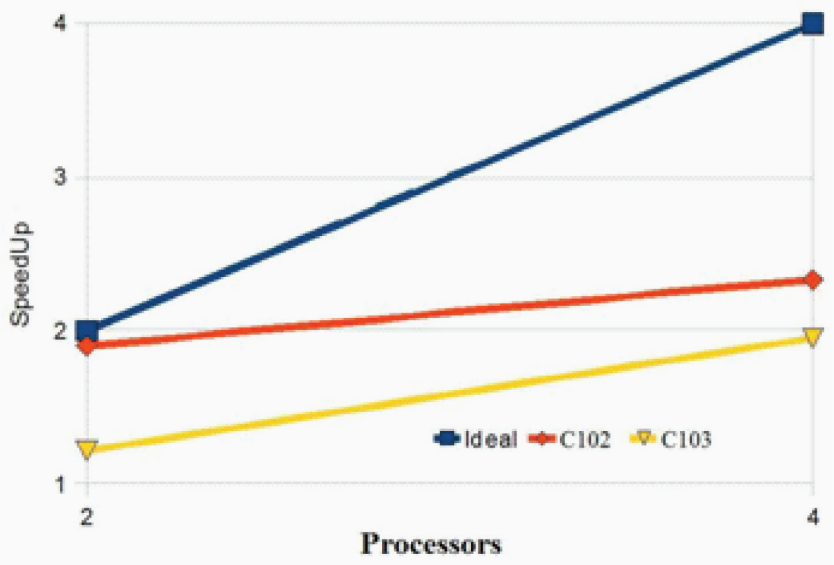
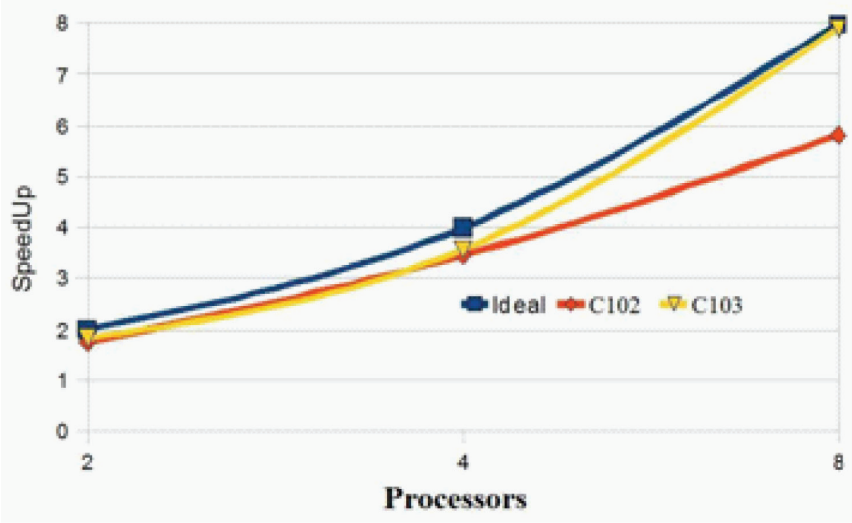


Figure 26. Speedup of testing C101 and C106 in the CIICAP cluster

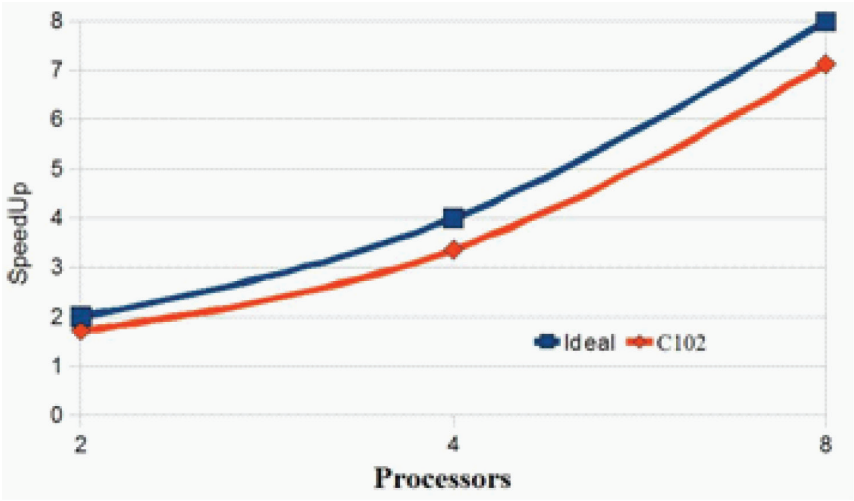


cluster CIICAp. When using 2 and 4 processors in the MiniGrid, efficiency is very similar to that obtained in the CIICAp cluster and better than that obtained in the NOPAL cluster. When using the MiniGrid, an increase in the efficiency of the algorithm is noted by increasing the number of processors from four to eight. When using each cluster separately, a decrease in efficiency is suffered.

With regard to the algorithm efficacy, the optimum of the two problems reported by Solomon for C102-100 and C103-100 was obtained in all executions by the genetic algorithm.

The efficacy of the algorithms can be increased by increasing the exploration and exploitation of

Figure 27. Speedup of testing C102 in the MiniGrid



the solution space. This is possible if the number of CPUs available increases in the MiniGrid, which in turn is possible due to the design of the genetic algorithm. The genetic algorithm suffers an increase in population size Pz if the number of CPUs increases (Figure 4), while the execution time remains relatively constant.

FUTURE WORK

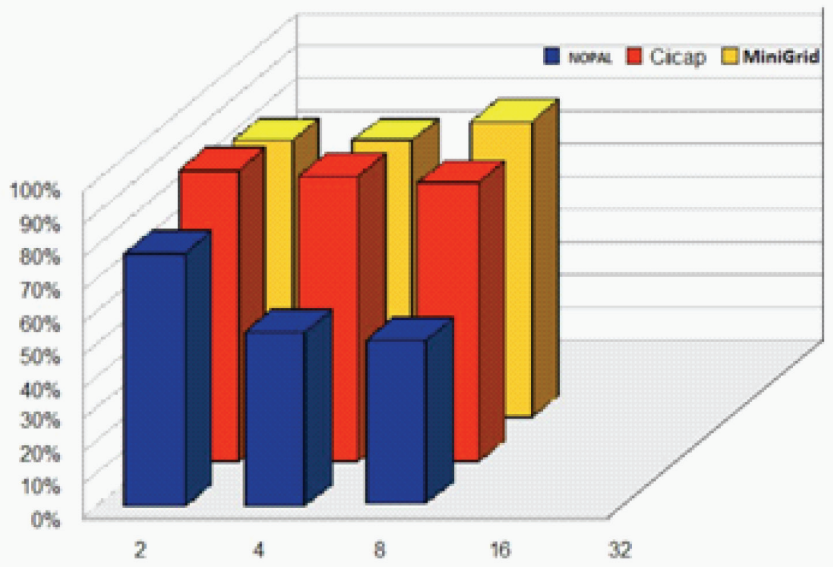
Future work will attempt to improve the infrastructure MiniGrid. The infrastructure used as a test scenario works, but for a production infrastructure, its performance is poor because the computer equipment and communications are not suitable. Because of this, the number of nodes in each cluster and the number of participating institutions will be increased, in order to increase the number of clusters in the MiniGrid. Future work will also attempt to make each cluster independent of local network traffic on Internet 2 in each of the participating institutions, to increase bandwidth and decrease the latency existing in the MiniGrid. When the MiniGrid is conditioned for production, real applications related to the logistics of product transport to companies will be sought. The theoretical study of the VRPTW for instances of 1000 customers will also be continued.

CONCLUSION

This chapter presents preliminary results of the execution of an iterative genetic algorithm, using an experimental MiniGrid. It involves mutation combined with two types of local search for solving the VRPTW model, which is NP-complete.

It can be concluded that the use of a grid platform for such problems is highly recommendable when searching for a better solution bound for problems of large instances in much shorter times. The reduction in time for finding better bounds is presented in terms of the number of CPUs available in the MiniGrid. The efficacy of the algorithms becomes larger because the exploration and exploitation in the solution space increases as the number of CPUs available in the MiniGrid increases, while the algorithm execution time remains relatively constant. When using the MiniGrid, an increase in the efficiency of the algorithm is observed when the number of processors is increased from four to eight. When using

Figure 28. Average efficiencies



each cluster separately, a decrease in efficiency is suffered.

The bandwidth is constantly changing and tends to improve in the afternoon, so it is recommended that the execution of algorithms that require extended communication time be performed in the evenings.

It is always advisable to keep communication between processes minimal when using a Grid platform for the execution of genetic algorithms. A structure of the algorithm should be defined that allows a reduction in the communication time between processes during its implementation. This paper presents a structure in which the mutation process is effective and independent of other processes and requires a great deal of CPU time. The process of selection and crossing is very fast, and does not depend on other processes running in the MiniGrid. It is important to define an algorithm structure where the iterative mutation takes place in the CPUs of the MiniGrid. This enables the algorithm to work efficiently because it does not require communication between nodes while performing the mutation, i.e., it does not affect the algorithm. The algorithm requires communication only once the mutation has taken place in MiniGrid nodes. This prevents latency from negatively influencing the implementation of the algorithm.

The genetic algorithm presented in this chapter can use all the resources of the MiniGrid, independent of the size of its platform. The more CPU resources the MiniGrid has, the greater exploration and exploitation it can perform for NP-complete problems in polynomial time. Thus, the Grid generally benefits the field of study of the models classified by complexity theory as the most difficult models in the world to solve. One of these models that benefits from study using this type of platform is the VRPTW model.

ACKNOWLEDGMENT

This work was supported by project FOMIX MOR-2009-C02-120102 and project I0101/131/07 C-229-07, CUDI-CONACYT, 2008-2009.

REFERENCES

Chang, Y., Cheng, W., Liu, X., & Xie, X. (2006). Application of grid technology in multi-objective aircraft optimization system. In *Proceedings of the 10th International Conference on Computer Supported Cooperative Work in Design* (pp. 1-5).

Cheng-Chung, C., & Smith, S. F. (1995). *A constraint satisfaction approach to makespan scheduling*. Pittsburgh, PA: Carnegie Mellon University.

Cheng-Chung, C., & Smith, S. F. (1997). *Applying constraint satisfaction techniques to job shop scheduling*. Pittsburgh, PA: Carnegie Mellon University.

Christodoulou, N., Wallace, M., & Kuchenhoff, V. (1994). Constraint logic programming and its application to fleet scheduling. *Information and Decision Technologies*, *19*(3), 135–144.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). *Introduction to algorithms*. Cambridge, MA: MIT Press.

Corporación Universitaria para el Desarrollo de Internet (CUDI). (2011). *A. C. Internet 2, México*. Retrieved July 17, 2011, from http://www.cudi.mx/index.html

Cruz-Chávez, M. A., & Díaz-Parra, O. (2010). Evolutionary algorithm for the vehicles routing problem with time windows based on a constraint satisfaction technique. *Computación y Sistemas, IPN*, *13*(3), 257–272.

Cruz-Chávez, M. A., Díaz-Parra, O., Hernández, J. A., Zavala-Díaz, J. C., & Martínez-Rangel, M. G. (2007, September 25-28). Search algorithm for the constraint satisfaction problem of VRPTW. In *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference*, México (pp 746-751).

Cruz-Chávez, M. A., Martínez-Oropeza, A., & Serna Barquera, S. A. (2010, September 28-October 1). Neighborhood hybrid structure for discrete optimization problems. In *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference*, México (pp. 108-113).

Cruz-Chávez, M. A., Rodríguez-León, A., Ávila-Melgar, E. Y., Juárez-Pérez, F., Cruz-Rosales, M. H., & Rivera-López, R. (2010). Gridification of genetic algorithm with reduced communication for the job shop scheduling problem. *International Journal of Grid and Distributed Computing*, *3*(3), 13–28.

Díaz-Parra, O., & Cruz-Chávez, M. A. (2008). Evolutionary algorithm with intelligent mutation operator that solves the vehicle routing problem of clustered classification with Time Windows. *Polish Journal of Environmental Studies*, *17*(4), 91–95.

Foster, I., & Kesselman, C. (Eds.). (2004). *The Grid 2, Second Edition: Blueprint for a new computing infrastructure* (The Elsevier Series in Grid Computing). San Francisco, CA: Morgan Kaufmann.

Fujisawa, K., Kojima, M., Takeda, A., & Yamashita, M. (2004). Solving large scale optimization problems via grid and cluster computing. *Journal of the Operations Research Society of Japan*, *47*(4), 265–274.

Garey, M. R., & Jonson, D. S. (1979). *Computers and intractability a guide to the theory of NP-completeness*. New York, NY: W. H. Freeman.

Gehring, H., & Homberger, J. (1999). *A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with Time Windows* (pp. 57-64). Jyvaskyla, Finland: University of Jyvaskyla. Retrieved from http://www.sintef.no/Projectweb/TOP/Problems/VRPTW/Homberger-benchmark/

Hansen, P., & Mladenovic, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, *130*, 449–467. doi:10.1016/S0377-2217(00)00100-4

Lim, D., Ong, Y.-S., Jin, Y., Sendhoff, B., & Lee, B.-S. (2007). Efficient hierarchical parallel genetic algorithms using grid computing. *Future Generation Computer Systems*, *23*(4), 658–670. doi:10.1016/j.future.2006.10.008

Luna, F., Nebro, A., Alba, E., & Durillo, J. (2008). Solving large-scale real-world telecommunication problems using a grid-based genetic algorithm. *Engineering Optimization*, *40*(11), 1067–1084. doi:10.1080/03052150802294581

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, CA (Vol. 1, pp. 281-297).

Mitrovic-Minic, S., & Krishnamurti, R. (2006). The multiple TSP with Time Windows: Vehicle bounds based on precedence graphs. *Operations Research Letters*, *34*(1), 111–120. doi:10.1016/j.orl.2005.01.009

Moscato, P., & Cotta, C. (2010). A modern introduction to memetic algorithms . In Gendreau, M., & Potvin, J.-Y. (Eds.), *Handbook of metaheuristics, international series in operations research & management science* (*Vol. 146*, pp. 141–183). New York, NY: Springer.

Papadimitriou, C. H., & Steiglitz, K. (1982). *Combinatorial optimization: Algorithms and complexity*. Upper Saddle River, NJ: Prentice Hall.

Rodriguez-León, A., Cruz-Chávez, M. A., Rivera-López, R., Ávila-Melgar, E. Y., Juárez-Pérez, F., & Díaz-Parra, O. (2010, September 28-October 1). A communication scheme for an experimental grid in the resolution of VRPTW using an evolutionary algorithm. In *Proceedings of the Electronics, Robotics and Automotive Mechanics Conference*, México (pp. 108-113).

Toth, P., & Vigo, D. (2001). *The vehicle routing problem*. Philadelphia, PA: Society of Industrial and Applied Mathematics.

ADDITIONAL READING

Baptista-Pereira, F., & Tavares, J. (Eds.). (2010). *Bio-inspired algorithms for the vehicle routing problem, studies in computational intelligence*. Berlin, Germany: Springer-Verlag.

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

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. In L. Rutkowski, R. Tadeusiewicz, L. A. Zadeh, & J. M. Zurada (Eds.), *Proceedings of the 9th International Conference on Artificial Intelligence and Soft Computing* (LNCS 5097, pp. 376-387).

Díaz-Parra, O., & Cruz-Chávez, M. A. (2008). General methodology for converting sequential evolutionary algorithms into parallel algorithms with OpenMP as applied to combinatorial optimization problems. *Polish Journal of Environmental Studies*, *17*(4), 240–245.

Golden, B., Raghavan, S., & Wasil, E. (Eds.). (2010). *The vehicle routing problem, latest advances and new challenges*. New York, NY: Springer.

KEY TERMS AND DEFINITIONS

Genetic Algorithm: Search type Meta heuristic that works in a non-deterministic manner, which mimics the natural process of evolution using techniques such as inheritance, mutation, selection and crossover. It is applied to find solutions to optimization problems.

Grid Computing: Set of geographically dispersed computing clusters, united by a virtual private network and considered to be a super virtual computer.

Latency: Measured time delay experienced by a Grid. The latency between two computing clusters, geographically distant but belonging to a common computational Grid, is the time the round trip takes, of sending a data packet from one cluster to another and back again.

Local Search: Heuristic method that seeks to improve the starting solution in a problem's solution space, through neighborhoods defined by a neighborhood structure. This generates a small perturbation in the starting solution, resulting in a neighboring solution with similar characteristics to the starting solution, which may be a better or worse solution.

Speedup: Relates to parallel computing and refers to a comparison of speed (faster) between the parallel form and the sequential form of the same algorithm.