Comparison of alternative processes for support decisions

Manuel MARTINEZ-ÁLVAREZ
Instituto Politécnico Nacional, Centro de Investigación en Computación (CIC-IPN), Departamento de Investigación en Ciencias de la Computación. Av. Juan de Dios Bátiz, esquina con Miguel Othón de Mendizábal, Mexico, D.F., 07738

and

Sandra-Dinora ORANTES-JIMÉNEZ
Instituto Politécnico Nacional, Centro de Investigación en Computación (CIC-IPN), Departamento de Investigación en Ciencias de la Computación. Av. Juan de Dios Bátiz, esquina con Miguel Othón de Mendizábal, Mexico, D.F., 07738

and

Graciela VÁZQUEZ-ÁLVAREZ
Instituto Politécnico Nacional, SEPI-Escuela Superior de Ingeniería Mecánica y Eléctrica (SEPI-ESIME-IPN), Av. Luis Enrique Erro S/N, Unidad Profesional Adolfo López Mateos, Zacatenco, Delegación Gustavo A. Madero, C.P. 07738, Mexico, Distrito Federal

ABSTRACT

There are many tasks that revolve around combinatorial analysis problems, same tasks found in Decision Support Systems (DSS) as most of these are responsible for assessing a number of possibilities to deliver the best options. Within the analysis of possible solutions is performed by the DSS there are alternative procedures inside the engine for making decisions that involve them. As part of these alternative procedures today has highlighted the use of metaheuristics, thus in this paper we propose a comparison of some of them trying to broaden the spectrum we have for the applications nowadays.

Keywords: Decision Support System, Combinatorial Optimization, Heuristics, Genetic Algorithms, Particle Swarm Optimization, Tabu Search, Optimization based on Ant Colony.

1. INTRODUCTION

Decision Support System (DSS) are part of the tactical and management levels of companies focus on problems unique nature of rapidly changing and usually there are no predefined processes or methods for solution.

The DSS are usually supported by problems of combinatorial analysis, in which it is not possible to obtain an optimal solution by a specific mathematical model, since the implementation of an exact algorithm would require a lot of computational resources, for this such problems using allowed heuristics efficiently obtain optimal solutions.

Heuristic methods or simply called heuristics seek and obtain the best (near-optimal) solution at a relatively low computational cost. Based on similar to those used for approximation algorithms, this kind of algorithms can be classified as constructive or local search methods, constructive algorithms generate solutions from scratch by iteratively adding components to an initially empty solution until the solution techniques is complete i.e., construct a solution to a combinatorial optimization problem incrementally. Thus, these algorithms step by step without backtracking add solution components until a complete solution is generated. On the other hand, local search algorithms start from an initial solution and repeatedly try to improve the current solution by local changes, i.e. by iteratively scanning sectors seek solutions to improve outcomes through changes locally. Although the construction algorithms are typically faster heuristics from the quality of the solutions they generate is often inferior to the quality found by local search algorithms [9].

Heuristic algorithms have had a big impact in solving allocation problems, task scheduling, transportation and production [12] [22]. When heuristics algorithms require other methods for creating and evaluating alternative solutions are called metaheuristics methods [14][24].

A metaheuristic can be seen as a general purpose heuristic method designed to guide a heuristic specific to regions of very promising solution i.e., spaces of high quality solution [9]. Thus, metaheuristics methods use iterative processes from which sail in solution spaces where specific rules applying methods converge towards increasingly better solutions.

Examples of metaheuristics are simulated annealing, Tabu search, Local Search iterative, Genetic Algorithms, Optimization based on ant colonies, etc.

Therefore, using as a basis some problems such as task scheduling, mapping routes, resource allocation, stable allocations, and this article will explain and make comparisons on some metaheuristics alternatives and thus have a broad knowledge needed when choose how to attack a problem in phase solution involving the DSS.

This paper is organized as follows. Information Systems and particularly the DSS are presented in section 2 and 3. Section 4 to 8 presents some metaheuristics seen as alternative processes of problem solving combinatorial optimization. In Section 9 to 12 some advantages and disadvantages of metaheuristics mentioned in the previous section are presented. Finally, are presented in Sections 13 and 14 respectively, the results and conclusions.

2. INFORMATION SYSTEMS

Information systems in organizations are designed and implemented in them not only to manage their information but also as a means to automate and improve their processes. Depending on the activities of users or the organizations that operate the different information systems is that they may find higher or lower benefits at the time of use. Process automation, DSS and data management are just some of the competitive advantages that converges the correct use of information systems. Consequently, information systems now constitute a key element in organizations, with a major impact on their processes, structure and culture [21]. A system will be more efficient and better the more you can improve business processes and support decision as this will lead to greater profitability in organizations while reducing their costs. [8]

Whether its stability, degree of cooperation or as reflected goal is that there may be different classifications of information systems. Considering the evolutionary logic that have followed different systems is that a perspective is presented from the point of view of the users who manage [18], this classification is identified in terms of the main groups of the organization to serve distinguishing according to levels or hierarchies according to their activities and in turn are classified as (See Figure 1):

- Executive Information System (EIS), are systems designed to provide analysis and comprehensive assessments of the company required by the directors of the organization.
- Decision Support System (DSS), are used by the headquarters of businesses and assist in decision support processes that encompass data on problems or unique nature.
- Management Information System (MIS), designating a service to the immediate management or administrative positions and which generates reports on the current situation of the organization for monitoring and control.
- Transaction Processing Systems (TPS), monitoring the activities and transactions of the organization elemental (collection, storage, processing and retrieval of data) is provided.

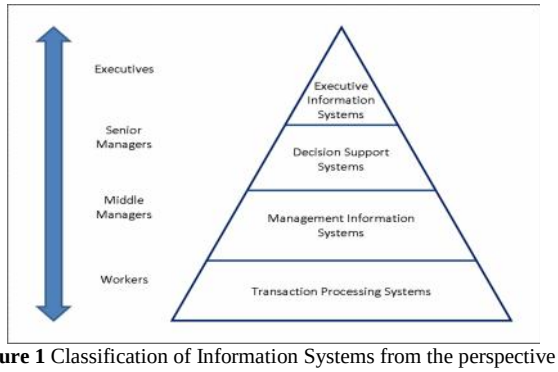

Figure 1 Classification of Information Systems from the perspective of users [18]

Interaction systems

All systems according to their classification can be seen independently, however, higher order systems often use lower systems, because through them, they can obtain the information they need for their work.

Based on the classification system from the user perspective is that the following relationship between the existing types is presented (See Figure 2).

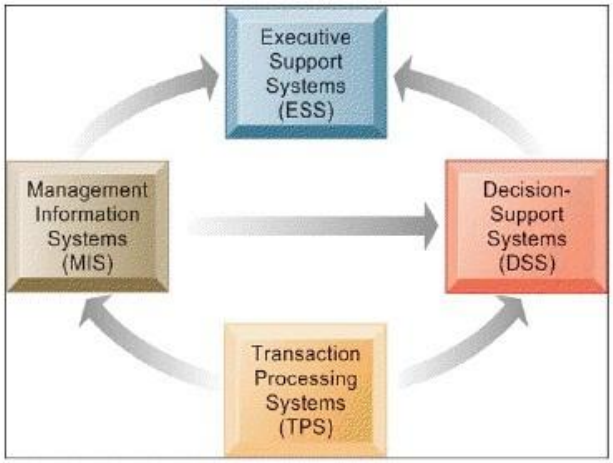

Figure 2 Interaction of information systems [29]

3. ANALYSIS OF DECISION SUPPORT SYSTEMS

The anticipations are part of the structure involving many companies planning tasks, budgeting and support decision. If the forecasts are inaccurate plans, budgets and decisions based on them tend to be ineffective and lead to lost opportunities or unnecessary expenses [19].

The objective of producing more accurate decisions has become the key factor in the industry ever since the higher certainty due to make a bigger process will be the reliability of its results. Consequently it is that the DSS are becoming increasingly a basis for the processes within companies. Being part of the middle management level or administrative DSS focus on issues unique nature and rapidly changing, they also usually do not exist for processes or predefined solution [18] methods.

Traditionally it has been considered that organizations have a predefined and static set of goals. However in order to remain competitive and survive in the field in which they specialize, is that they have chosen to improve the ability to respond and adapt to changes that occur in your business environment [3].

In recent years the DSS, rather than focus on the information the manager needs and risk factors, are now trying to support the overall processes involving the decision to be taken and thus reduce partiality in their results [6].

The first tool used in the DSS was the Data Warehouses, from there two tools, which were in line analytical processing OLAP (On- Line Analytical Processing) and Data Mining were derived [27]. All these tools agree on the same approach, and provide a historical process and unified view of enterprise data.

Today the field of DSS has had to employ new methods that fit the needs of the environment and has included models such as Artificial Intelligence, Expert Systems, Adaptive Systems and Unconventional Computation among others [3].

A basic model for decision making can be divided into three stages: formulation, solution and analysis [27].

- Formulation. Proposal is an alternative solution. From the real problem adapting to the proposed solution is expected to generate all points of optimization.
- Solution. Comprising the step is the algorithm or method previously proposed alternative solution. This is where the reliability, performance and accuracy of the algorithms and techniques used will impact the success or failure of the solution.
- Analysis. Stage at which the result given by the alternative solution proposed should be interpreted and test in order to know the impact and benefits of the system developed.

Components of Decision Support Systems

In an overview of the components of the DSS should have elements of data processing module and finally, with the user interface [18].

- Data Elements. This component consists of the database and the DSS is a set of historical and current data from multiple external sites sources of information, these can come from other systems such as TPS.
- Processing Module. This component consists of all the software tools used for data analysis. Contains OLAP tools, data mining tools, tools for Data Warehouse, various algorithms, etc.
- User Interface. This component comprises presenting the system about the user who will use it. It is the means by which the user can interact with the computer equipment as it is intended to provide information about the processes and tools for the control of applications through which the user usually seen on the screen.

Figure 3 shows how the components of DSS interact.

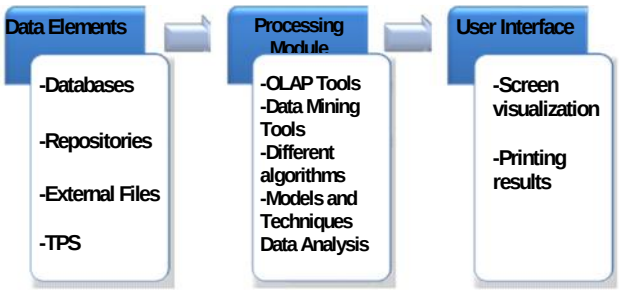

Figure 3 Interaction of the components of a DSS [18]

4. ALTERNATIVE RESOLUTION PROCESSES (METAHEURISTICS)

Founded on the stage of the DSS solution is that the algorithms used to solve different problems in the systems listed. Within the problems of combinatorial analysis are two kinds of algorithms available for their solution: exact algorithms and approximate algorithms [9].

Exact algorithms are guaranteed to find the optimal solution of a problem, plus the ability to prove any instance of finite size. On the other hand, the NP-Hard problems (NP-Hard), in the worst case, the exponential time is considered to find the optimal solution.

If the optimal solution obtained cannot be found efficiently in practice, the only possibility is to try to find the best also results in an efficient manner. In other words, the guarantee to find optimal solutions can be sacrificed to achieve very good solutions in polynomial time. Approximate algorithms often vaguely called heuristics, are then those looking to get acceptable solutions without consuming many resources or spend much time in execution. By consequent, metaheuristics become a general algorithmic framework which can be applied to different optimization problems to be adapted with minor modifications to specific problems [9].

The difficulties related to work within combinatorial optimization problem of great magnitude have contributed to the development of alternative proposals for their solution. Classic to solve problems such as linear programming or dynamic programming techniques have not been sufficiently effective when solving problems of the type NP-Hard to be a large number of variables and complex objective functions. To overcome these problems, researchers have proposed the use of metaheuristics consist in finding nearly optimal results for your solution.

Metaheuristics classified as evolutionary algorithms [12] or as swarm intelligence [16] [26] are stochastic search methods that mimic the metaphor of natural biological evolution or the social behavior of species. Examples include such as ants looking for the best route to reach their food or birds find their destination during migration where the behavior of each species is guided by learning, adaptation or evolution. Efficient imitation of these behaviors of the species have been studied and implemented in systems that seek fast and robust solutions to complex combinatorial optimization problems [12].

There are countless of combinatorial optimization problems in NP type which can be applied metaheuristics [2] [7] [15] [17]. Many of these problems can be considered within the following categories [9]: Routing, Assignment, scheduling and issues subsets problems.

Within each of these categories, there are a variety of problems in each of them, same as to be attacked by some metaheuristic may generate better or worse depending on the solution is carried out.

Routing problems, for example, are those in which one or more agents must visit a number of predefined locations and whose objective function depends on the order in which they are visited, two of these problems considered classics are: The Traveling Salesman Problem (TSP) [23] or Vehicle Routing Problem (VRP) [28].

The task allocation problems are to establish a set of "items" (objects, activities, etc.) to a number of sources (locations, agents, etc.) subject to some restrictions. These mappings can be considered as mappings between entities. Examples of such problems are [9] [15]: Quadratic Assignment Problem, Generalized Assignment Problem, and Frequency Assignment.

The task scheduling problems in a broad sense have to do with the allocation of resources to tasks distributed in a given time. Scheduling problems focus on industries in production and manufacturing. Within these problems are: The SMTWTP (Single -Machine Total Weighted Tardiness Problem) [1] [20], JSP (Job Shop Problem), OSP (Open Shop Problem) [9].

In sub-problems, the solution is immersed in the representation of groups of elements under a number of constraints. That is,

through permutation or combination of components of a given instance and obeying the restrictions involved is likely to be a solution for this type of problems. Some of the problems of this type are set covering, Set Splitting, Set Packing [15]. In general, evolutionary algorithms and Swarm Intelligence share a common approach in applications to solve a problem optimization, for this, first the problem needs to be represented to suit each method, once the algorithm is to be used should be applied iteratively to arrive at an acceptable solution, so at the end only look for the correct interpretation of the results obtained by each method. A brief description of some algorithms of this type is presented in the following sections.

5. GENETIC ALGORITHMS

A genetic algorithm easy part of having a "**P**" population "**p**" in which individuals "$\mathbf{P_x}$" represent solutions to optimization problems. Here an evolutionary process takes place within a population of candidate solutions.

Genetic algorithms are inspired by a physical condition of biological systems through its evolution. The solution given by the problem is represented as a chain called "Chromosome", which consists of a set of elements called "Genes", which maintain the set of values for the optimization variables. Genetic algorithms work with a random population of potential solutions (chromosomes). The fitness of each chromosome is determined by evaluation of performances in front of the objective function.

The purpose of these algorithms is to simulate the natural survival of entities through the exchange of information over the strongest chromosomes (through a crossover or mutation) to produce more offspring chromosomes with better solutions [12] [16]. So the child solutions (solutions from the "offspring") are evaluated and used to develop the new population provided they offer better solutions than the weak members of the population. Usually it's a continuous process of improvement for a large number of generations to obtain the best fit result [13].

The basic genetic algorithm pseudocode is shown in Figure 4 in which four main parameters affecting the performance of the genetic algorithm: the size of the population, the number of generations, the operation of crossing and mutation operation.

```
GeneticAlgorithm(){
    Generate a random population of P solutions (chromosomes)
    For each individual i ∈ P compute its fitness (i)
     For each generation
        Randomly select a crossing or mutation operation
        If Cross
            Select two random parents ia and ib
            Generate one son ic from Cross(ia and ib)
        If Mutation
            Select one chromosome i randomly
            Generate one son ic form Mutation(i)
           Calculate the Aptitude ic
        If ic is better than the worst chromosome that has
            Replace the worst chromosome that resolves ic
        Check for Completion()=true
}
```

Figure 4 Basic Pseudocode of genetic algorithms [12]

6. PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization (PSO) algorithms originated as a simulation of a simplified social system elegant and unpredictable choreography of a flock of birds [11]. These algorithms mimic the communication between the flock at the time of your flight. Each bird look in any direction in particular, to communicate with other birds identified which is in a better location. Consequently each bird accelerates towards the best bird using a rate that depends on your current position. Each bird, then investigates the search space from its new position, eventually the process must be repeated until the desired destination is reached.

In PSO each solution is a "bird" of the flock and is known as a "particle". A particle would be analogous to a chromosome (member of a population) in genetic algorithms. Unlike genetic algorithms on PSO does not create new parent birds, but birds of a population only evolve their social behavior and consequently its movement to a destination [12]. PSO's pseudocode is shows in Figure 5.

```
MetaheuristicPSO(){
Generate a random population of N solutions (particles)
For each individual i∈N calculate Aptitude(i)
Initialize a value of weight factor w
      For each particle
   Assign pBetter the best position of the particle i
   If Aptitude(i) is better than pBetter
              pMejor(i)=fitness(i)
   Assign gBetter as the best fitness of all the particles
    For each Particle
            Calculate the velocity of the particle
            Update the position of the particle
       Update the value of the weight factor   w
            Check for Completion()=true
}
```

Figure 5 Pseudocode of PSO [12]

7. TABU SEARCH

The Tabu search metaheuristic is a procedure used to handle a heuristic local search algorithm to avoid the process stops at a local optimum. Therefore, Tabu search makes a scan through a configuration space appropriately defining the local optima [13].

This type of algorithm does is keep a list of points in the problem space that have been evaluated wrong, away from it as much as you can, in order to make the search for its solution [11]. So while searching for the solution by maintaining the list of bad results is that it prevents the process and return to local optima into repetitive cycle's performed.

The algorithms consider these movements as "Tabu movements" and therefore prohibit a configuration to be accessed again.

The idea is to make moves on the solution space, when a movement has been classified as taboo and after being analyzed produces better chosen reference value (which can be a good solution uncertainty or another previously found) objective function, then a criterion called "rule of aspiration " consisting cancel the prohibition of a state and return to accept the motion for walking on the solution space [13] applies.

More specifically, the Tabu search is based on the premise of solving a problem from an adaptive memory and responsive exploration. The role of adaptive memory Tabu search enables the application of procedures able to search a solution space economically and effectively since local search options are governed by the information gathered during the search in progress. The emphasis on the exploration of a solution using Tabu search with either deterministic or probabilistic implementation is derived from the superposition of a bad strategic choice can provide more information than a good random choice. In a system that uses memory, a bad choice based strategy can provide useful clues about how profitable will change strategy [10].

8. OPTIMIZATION BASED ON ANT COLONY

The Ant Colony Optimization (ACO) is one of the most recent techniques for solving optimization problems. This technique is based on the behavior of ant colonies. The basis of this behavior is the indirect communication between the ants when they performed to random search of their food, leaving its passage chemicals called pheromones, which it then used to make marking routes from the nest to the point where they can collect their food.

ACO is a metaheuristic in which a colony of artificial ants cooperates to find the best solutions to optimization problems. Collaboration is the key design component of ACO algorithms. The idea is to allocate the computational resources to establish relatively simple agents (artificial ants) that communicate indirectly through the environment. Here's how ACO can obtain good solutions as an emergent property of the cooperative interaction of its agents [9].

First generate a finite set of solution components. Then you have to define a set of values of "pheromones" that are nothing more than model values that will be used as probabilistic parameters. The model of the "pheromone" will be used to generate solutions to the problem probabilistically. In general, the ACO approach attempts to solve optimization problems by iteration of the following steps [5]:

- Candidate solutions are constructed using a pheromone model i.e., parameterized probability distribution over the space of solutions.
- The candidate solutions are used to modify the pheromone values in a way that will be considered for future high-quality solutions. Informally, an ACO algorithm can be thought of as an interaction of three methods: *AntBuildSolution*, *PheromoneUpdate* and Actions [9]. (See Figure 6)

AntBuildSolution is the function that manages the colony of artificial ants concurrently and asynchronously, visit the states of problem identified by its movement through the neighboring nodes in the graph generated resulting from the representation of the problem. Ants move by using stochastic policy decisions at the local level using the pheromone trails deposited on different nodes of the graph and the heuristic information you have. Once the ant has built a solution or while the solution is being built, the ant will evaluate the partial solution built for use by the *PheromoneUpdate* procedure for deciding the amount of pheromone deposited.

PheromoneUpdate is the process by which the pheromone trail is modified. The pheromone trail value may increase, when the ant's pheromone deposited on graph nodes, or reduce, this due to the evaporation of the pheromone. Both increases and decreases of pheromones consist of variants specific to the metaheuristic functions.

Actions is the procedure used to implement the actions that have nothing to do with the performance of the ants, but with local optimizations and global management of information pertaining to the metaheuristics.

```
MetaheuristicACO {
        AntBuildSolution PheromoneUpdate
        Actions
}
```

Figure 6 Pseudocode of the ACO [9] metaheuristic

9. ADVANTAGES AND DISADVANTAGES OF GENETIC ALGORITHMS

Advantages

- Allows you to search more intensively on a particular space walking through neighboring solutions [13].
- Although the first solutions are made at random, the following solutions will produce better results to complete the evaluation of the generations of the population [25].

Disadvantage

- Need a quite number of iterations in order to find good results. In case you have not selected a correct number of generations then fall into partial optimal.

10. ADVANTAGES AND DISADVANTAGES OF PARTICLE SWARM OPTIMIZATION

Advantages

- In PSO although it requires some "evolution", unlike genetic algorithms need calculate no mutation or cross in its entirety, but the evolution of which is only concerned with developments in the behavior of its elements and therefore its movement towards a destination [26].
- It has very good performance in multi-target areas, dynamic optimization and constraint handling.

Disadvantages

- The method suffers easily optimal partial drop, making it less precise in regulation of the speed and direction of the elements of the algorithm.
- The method cannot solve problems of scattering (scattering problems), nor can solve problems without coordinate systems such as the field of energy (Energy Field) or the rules of the motion of particles in a field of energy (The rules of moving the particles in the energy field) [26].

11. ADVANTAGES AND DISADVANTAGES OF TABU SEARH

Advantages

- Avoid bad solutions through a list of "Tabu movements" [13].
- Can get solutions in a short time [10].

Disadvantage

- Despite operating in a very short time is not always the best solution [10].

12. ADVANTAGES AND DISADVANTAGES OF OPTIMIZATION BASED ON ANT COLONY

Advantages

- By nature made to the problem, since it uses agents (artificial ants) independently, it is easy to parallelize.
- Positive feedback for finding better solutions [26].
- Adapts to changes in the problem statement in a dynamic way, i.e., in real time if the objective function change, the algorithm could be easily adapted.
- Because the algorithm is based on how ants leave their nest in search of a destination, its food is that it has a greater advantage when problems have a source and destination previously defined [26].

Disadvantages

- The theoretical analysis is very difficult [26].
- Sequences using random functions based decisions.

13. COMBINATORIAL ANALYSIS

An entry for a computational problem will be encoded as a finite binary string s, whose size is given by **| *s* |.** Meanwhile a decision problem will be the one with a number of strings respond with a "Yes" or "No" depending on the input string you have. Will say that an *algorithm A* to solve a decision problem run in polynomial time if there is a polynomial ***p()*** that for any input string ***s*** an *algorithm A* will end on ***s*** at most **p(|s|)** steps.

Also an algorithm can be verified efficiently regardless if efficiently be solved. A verifier algorithm for a given problem has different structure on an algorithm that seeks to solve a problem, because in order to check a solution, only an input string containing evidence that the algorithm will answer "Yes" when it is needed its solution [17].

It is said that an efficient verifier to a problem will be that knowing a string ***s*** can determine in polynomial time whether this belongs to the solution space of the problem at hand. Well defined that a problem is **NP**, if it belongs to the set of problems for which there is an efficient verifier.

An optimization problem consists minimize or maximize the value of a variable, that is, is a problem which seeks to determine or calculate the minimum or maximum value in a function.

In computer science, mainly graphs, we can find optimization problems that whatever the problem you want to solve can be used exhaustive search techniques for their solution, same which by their nature tend to have a large time complexity. Such problems using metaheuristics as your solution, but say not deliver the best results always provide solutions to the expected solution approach.

14. RESULTS

Depending on the type of problems that are evaluated or type of instances that might prove the result of the implementation of metaheuristics vary as a pattern is not preserved in the results they generate. Throughout the article we have explained them in some metaheuristics which is now some evidence will be analyzed.

In [25] a comparison chart between the ACO and Genetic Algorithms, as is shown in Table 1.

Table 1 Performance between Genetic Algorithm and ACO complexity in different environments [25]

Number of nodes	12 nodes		22 nodes		63 nodes	
Algorithm	GA	ACO	GA	ACO	GA	ACO
Time (sec)	72.12 44	9.337	242.7 088	32.566	2144 .84	814.8 19
Iteration	7.1	1.9	9.5	3.1	21.8	3.5

Figure 7 shows how both metaheuristics increases the number of iterations agreement nodes to evaluate augments but seen as ACO clearly indicate better performance in the number of iterations. As to the execution time of the algorithm is also observed how ACO maintains better performance. (See Figure 7 and Figure 8)

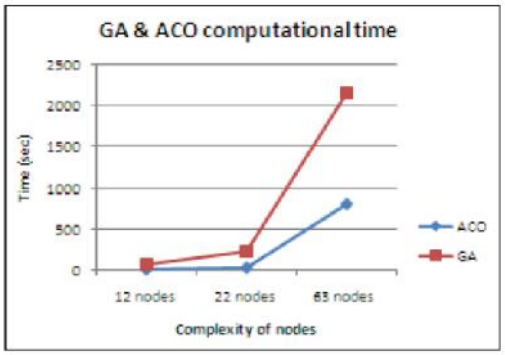

Figure 7 Computational time between ACO and genetic algorithms [25]

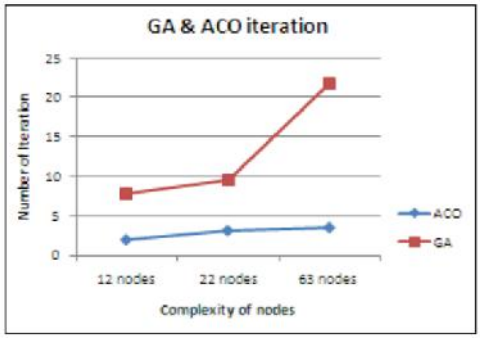

Figure 8 Number of iterations between ACO and Genetic Algorithms [25]

One of the main reasons for improved performance has ACO is due to the behavior of the algorithm, starting how populations (solutions) are generated and hence as is that they prove to be better obtained. Although both algorithms are based on randomized space solutions livelihoods ACO is benefited because the rules in handling deposits pheromone [9].

Additionally, ACO quality is better because the populations in each generation resulting improves outcomes, this due to updates in the pheromone trails in solution spaces. However, in genetic algorithms, crosses or mutations those are made, but always seek better results there is no guarantee that the children resulting solutions quickly improve the solution. So this will affect the number of iterations that occur. In addition to the adjustment of parameters in ACO takes to adapt the problem to the different amounts of nodes is much easier than what you have to adjust on genetic algorithms [25].

Moreover, in [26] mention that ACO has been applied to various optimization problems such as quadratic assignment and routing of vehicles in which have always obtained good results is made.

For his part in [13], an evaluation between 8 cities for Genetic Algorithms, Tabu Search and ACO was made.

When implementing different algorithms showed that the three metaheuristics could obtain the same solution, however the drawbacks involving implementation naturally, such as genetic algorithm requiring specialized operators for both the "cross" to the "mutation" same operations not being conducted properly produce a generation of infeasible results. Mention is also made that the algorithms that offer greater advantages are ACO and Tabu search regarding ease of implementation and its convergence is performed in a smaller number of iterations. In an analysis in [14] shows how the complexity of ACO is lower with respect to genetic algorithms. On the other hand mentioned that genetic algorithms can ensure the evolution of the population by operators "crossover" and "mutation", however because if you try to always select the best individual, if operators are chosen from the solution properly fall into local optima.

In [12] the tests were performed between genetic algorithms, PSO and ACO in discrete optimization problems genetic algorithms showed the poorest results and are observed to increase as the number of variables the successful outcome decreases. Here it is seen as the results between PSO and ACO are very similar however regarding ACO processing time shows a better performance. (See Table 2)

Table 2 Results of a discrete optimization problem to minimize days and costs [12]

Algorithm	Minimum project duration (days)	Minimum cost ($)	Processing time (s)
GAs	113	162,270	16
PSO	110	161,270	15
ACO	110	161,270	10

Reviewing PSO in [26] mentioned that there has been very good implementations to solve multi-objective optimization.

Meanwhile in [4] a method of Tabu search which is completely deterministic so it would generate a better return for their results is presented.

Comparison of metaheuristics for 8-node problem
Table 3 shows the results of a graph with 8 nodes.

As can be seen all techniques obtained the same results, only differed in the use of each algorithm was in the number of iterations, since the required less effort was the ACO metaheuristic. Note that, in essence, each algorithm performs "iterations", but in particular, each of them, name them differently because of the context handlers.

Table 3 Comparison of metaheuristics for 8-node problem [25]

AGORITHM	ITERATIONS	RESULTS (VALUE TO MINIMIZE)
OPTIMIZATION BASED ON ANT COLONY	10 cycles	1490
TABU SEARCH	14 iterations	1490
GENETIC ALGORITHMS	125 generations	1490
PARTICLE SWARM OPTIMIZATION	625 proposals	1490

15. CONCLUSIONS

While it is true that the fact of using an information system in an organization can help increase your productivity, if the decision engine that is used does not provide adequate results, just do not give a chance to solve power system problems that arise.

A system will be better and generate a greater impact with their results whenever it is able to improve business processes and decision making through the support of its processes with the use of appropriate algorithms to the problem. While an algorithm may lead to a solution, there are algorithms that can generate the same results but in less time and with less effort used.

Automation of processes, decision making and data management are just some of the points that can be provided to optimize the algorithms and when to support their applications are indicated.

In assessments were performed with different metaheuristics, all results were always good, the only thing it was used to vary the performance and the difficulty in performing the operations. Note that depending on the problem to be solved is

first necessary to adapt it to find solution method you want to use, you should also make a preliminary assessment to see which method is best may be coupled to the problem.

This research led to the development of a master's thesis on which was based on the comparison of alternative processes for decision making described in this paper, and the subsequent selection and application of ant colony algorithm for implementation an analysis engine, useful information for system support decision that serves and assists the management and optimization of processes of care to hydraulic leaks, and in this way to minimize water wastage and times response needed to carry out repairs of the same, taking as a case study a reseller water agency of a State of the Mexican Republic, where it is being evaluated for probable acquisition and implementation.

16. REFERENCES

[1] A. Allahverdi, C- Ng, T. Cheng & M. Kovalyoc, **A surver of scheduling problems with setup times or costs,** Elsevier, No.48, 2008.

[2] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, & M. Protasi, **Complexity and Approximation. Combinatorial Optimization Problems and Their Approximability Properties,** USA: Springer, 2003.

[3] G.D.Bhatt, & J. Zaveri,, **The enabling role of decision support systems in organizational learning,** Elsevier, 13, 2001.

[4] U. Bilge, M. Kurtulan, & F. Kirac, **A Tabu search algorithm for the single machine total weighted tardniess problem,** Elsevier, 13, 2006.

[5] C. Blum, **Ant colony optimization: Introduction and recent trends**. Elsevier, 21, 2005.

[6] J.Q. Chen, & S.M. Lee, **An exploratory cognitive DSS for strategic decision making**. Elsevier, 14, 2002.

[7] S. Dasgupta, C. Papadimitriou, & U. Vazirani, **Algorithms** (First ed.). New York, United States: McGraw-Hill, 2008.

[8] C. De Pablos, J.J. López-Hermoso, S. Martín-Romo, S. Medina, A. Montero, & J.J. Nájera, **Dirección y Gestión de los Sistemas de Información de la Empresa,** (Second ed.), Madrid, España: ESIC, 2006.

[9] M. Dorigo, & T. Stützle, **Ant Colony Optimization**. Massachusetts: Bradford, 2004.

[10] D.Z. Du, & P. Pardalos, **Handbook of Combinatorial Optimization,** USA: Springer, 2013.

[11] R. Eberhart, & Y. Shi, **Particle Swarm Optimzation. Developments, Aplications and Resources,** IEEE, 6, 2001.

[12] E. Elbeltagi, T. Hegazy, & D. Grierson, **Comparison among five evolutionary-based optimization algorithms,** Elsevier, 11, 2005.

[13] R. Hincapié, C. Ríos, & R. Gallego, **Técnicas heurísticas aplicadas al problema del cartero viajante (TSP),** Scientia et Technica, 6, 2004.

[14] W. Hui, **Comparison of several intelligent algorithms for solving TSP problem in industrial engineering,** Elsevier, 10, 2012.

[15] V. Kann, & P. Crescenzi, **A compendium of NP optimization problems,** Retrieved Agosto 15, 2013, from A compendium of NP optimization problems: http://www.nada.kth.se/~viggo/wwwcompendium/, Publicado 2005, Julio 01.

[16] J. Kennedy, & R. Eberhart, **Swarm Intelligence**. USA: Morgan Kaufmann, 2001.

[17] J. Kleinberg, & E. Tardos, **Algorithm Design,** (First ed.). Ithaca, New York, United States: Addison Wesley, 2006.

[18] K.C. Laudon, & J.P. Laudon, **Management Information Systems: Managing the Digital Firm**, (Tenth ed.), New York, United States: Pearson, 2008.

[19] M. Lawrence, P. Goodwin, & R. Fildes, **Influence of user participation on DSS use and decision accuracy**, Omega, 12, 2002.

[20] J.K. Lenstra, R. Kan, & P. Brucker, **Complexity of machine scheduling problems. Annals of Discrete Mathematics**, 20, 1977.

[21] A. Orero, J.I. López,, & J.L. Arroyo, **Caso Lyma Getafe, S.A.M.: desarrollo e implantación de un plan estratégico de sistemas en una empresa municipal de servicios**, Revista de empresa, 11, 2007.

[22] C. Rajendran, & H. Ziegler, **Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs,** Elsevier, 13, 2002.

[23] C. Rego, D. Gamboa, F. Glover, & C. Osterman, **Traveling salesman problem heuristics: Leading methods, implementations and latest advances**, Elsevier, 15, 2011.

[24] R. Ruiz, & C. Maroto, **A comprehensive review and evaluation of permutation flowshop heuristics**, Elsevier, 16, 2005.

[25] N.B. Sariff, & N. Buniyamin, **Comparative Study of Genetic Algorithm and Ant Colony Optimization Algorithm Performances for Robot Path Planning in Global Static Environments of Different Complexities**, IEEE, 6, 2009.

[26] V. Selvi, & R. Umarani, **Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques**, International Journal of Computer Applications, 6, 2010.

[27] J. Shim, M. Warkentin, J.F. Courtney, D.J. Power, R. Sharda, & C. Carlsson, **Past, present, and future of decision support technology**, Elsevier, 16, 2002.

[28] P. Toth, & D. Vigo, **The Vehicle Routing Problem,** USA: Siam, 2002.

[29] Kenneth C. Laudon, & Jane P. Laudon, **Sistemas de Información Gerencial**, Décimo segunda edición, Pearson Educación, 2012.