ISSN 1870-4069

Job Shop Optimization by Distributed Cooperation Using the PSO Algorithm

Oscar Martínez[1], Luis Carlos Mendez[1], Martín Montes Rivera[2], Alberto Ochoa-Zezzatti[1], Luis Perez-Dominguez[1]

[1] Universidad Autónoma de Ciudad Juárez, Mdxico

[2] Universidad Politécnica de Aguascalientes, Mdxico

adan.martinez@uacj.mx, luis.mendez@uacj.mx, martin.montes@upa.edu.mx, alberto.ochoa@uacj.mx, luis.perez@uacj.mx

Abstract. As an alternative to the search for the best solution for the Job Shop Scheduling Problem, a combination of two known techniques in treating NP-complete problems is proposed. First of all, there is optimization by the Particle Swarm Optimization algorithm (PSO) where it is sought to find a better solution through the cooperation and communication of different particles and secondly the treatment by distributed computation, where a significant problem can be divided into small sections, in this experiment both techniques are applied. Thus, the selected problem was solved by 300 physically distributed computers that worked simultaneously for 5 hours connected only by a network to the Internet. Each computer sends its best solution found to the cloud, the result of each computer is compared with the best found, and each best solution is stored in the database. At the end of the experiment, there are significant differences between the results obtained at each computer.

Keywords: Job shop, PSO, distributed computing, optimization.

1 Introduction

Currently, the allocation problems are too complex to be solved by a computer in a single day. These problems are known as NP-complete and are derived from real problems in industry, medicine, metallurgy, economics, and others [1]. One of the most known problems in the industry is the Job Shop Scheduling Problem (JSSP) workshop organization, which consists of finding the best assignment of tasks for each machine, and the number of possible combinations grows exponentially as the number of tasks increases[2]. More specifically, it can be said that machines perform various tasks and produce a particular product from an input. For this, a series of steps are followed. Each step consists of assigning a specific machine for some time. Each of these steps is

called an operation necessary to produce the product. A job will be associated with its corresponding operations for obtaining different products. A program is an assignment that sets each operation a time interval. The problem is finding a program that minimizes the time required to complete all the operations called makespan [3]. For example, in figure 1, two different programs for three operations of 3 tasks each and five available machines, in the first program, the processes are finished in 14 units of time and program two (optimized) all the activities are carried out in 10 units of time, you can see how downtimes decrease.

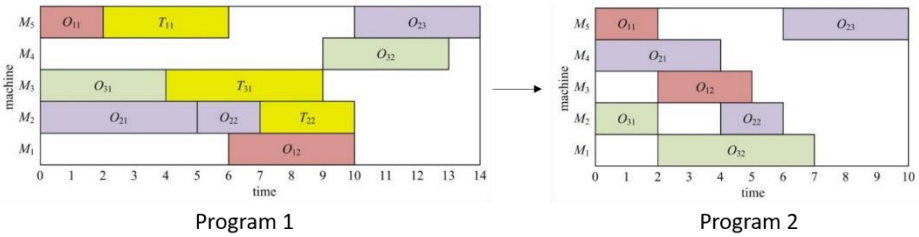

Fig. 1. Programs for the same Job Shop.

To solve exponential growth problems, several researchers have chosen to use optimization algorithms based on a swarm of particles, which aims to imitate the behavior of some animal species that act in coordination to achieve some objective [4]. Each of the algorithms derived from group behavior in the animal kingdom has specific characteristics of behavior and communication. However, all share some general characteristics initially. First, the group is randomly dispersed in space. Then, when a member of the herd detects a good objective, all the group members approach in coordination to focus on a specific point. Some of the most used PSO is based on fish, birds, and bats [5]. Figure 2 shows how a group of fish (shoals) act together to achieve a general purpose. It is necessary to have a communication system and established behaviors to carry out this event.

Fig. 2. Bank of fish.

The behavior above is transformed into a computationally executable algorithm and applied to the JSSP in such a way that each particle represents a possible solution, and by bringing all the particles closer to the best solution found, it is expected to find the best of all, the result of this process is good and is done in a matter of seconds. The

general flow diagram for the PSO is represented in figure 3. All the particles are randomly generated with the necessary parameters, followed by a first evaluation of the results obtained. Once evaluated they can be established global variables such as the best position and the speed and orientation in which the other particles will move from the second evaluation into a loop with each iteration, and the distributed particles move in the direction of the best solution found. So far, stop criteria have been established to terminate the loop. Some of these conditions may be the number of iterations, execution time, or the value of the best result found.

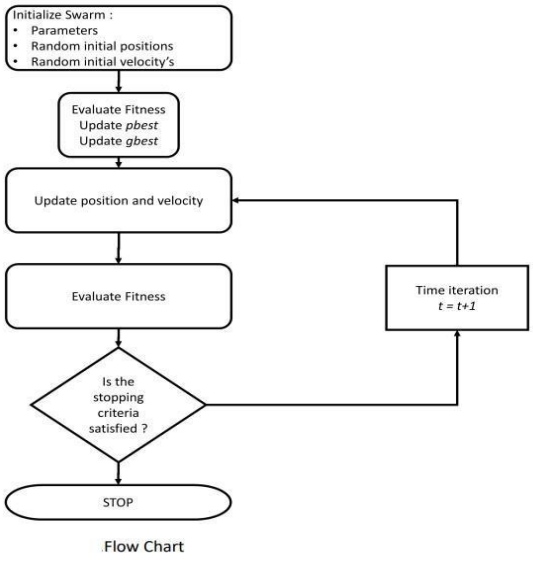

Fig. 3. General flow diagram of the PSO.

In the beginning, all particles are dispersed among the entire universe of possible solutions. As the iterations of the PSO something-rhythm are executed, they all end up grouped in a nearby place or even in the same place. Figure 4 shows the beginning and end of the positions of the particles.

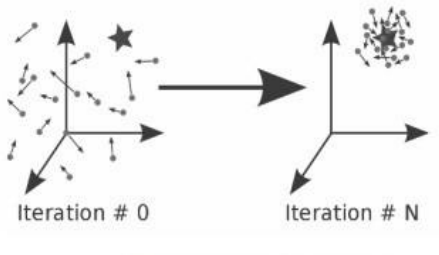

Fig. 4. Particles convergence.

Furthermore, however, the best solution is not always found, and the greater the number of variables involved, the lower the probability of finding the best solution. The only method to find the best solution in NP-complete problems is to solve them by evaluating all possible combinations. However, a computer would take a long time to evaluate each combination.

Therefore, some researchers divide the problem into parts to process the algorithm using multiple computers (distributed computing). Thus, evaluating the different combinations by sections [6] apart from working distributed can also use parallel processing to decrease times.

There are different ways of working in distributed computing both in the architecture and in the way of processing information, one of the most widely used architectures is to connect multiple devices to the cloud (Internet), as shown in figure 5.

Fig. 5. Distributed computing scheme in the cloud.

One of the best-known architectures is client-server, with each device as the client and the cloud as the server.

The processes can be distributed and executed in multiple ways. Although, usually, the server is the one who hosts all the information and the client who makes requests or queries to said server [7], the exact way in which the client and the server interact is defined by the algorithm developed.

However, two forms of interaction can be 1.- To leave the heavy work for the server and use the client as an interface to display data. 2.- Use the server only to store data and process the information on each device or divide the degree of processing and storage between the connected devices and the server.

Figure 6 shows the client-server architecture and the interaction between its parts.

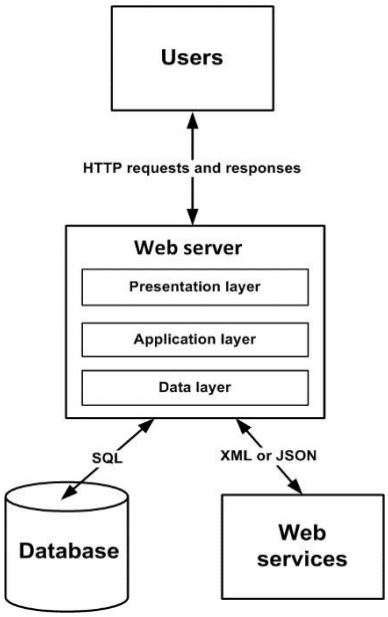

Fig. 6. Client-server architecture.

2 Methodology

The methodology used consisted of developing a generalized particle swarm algorithm to solve only a selected instance. The algorithm was developed in a web environment. Through development using HTML, CSS, PHP, JavaScript, and AJAX languages, a page was created to interact with the client and the server in an automated way. The database generated includes total estimation time, machinery-task assignment, contributing computer IP, recording date, database creation date. Finally, the experiment is carried out to obtain the data generated, and at the end, an evaluation of the data is carried out, obtaining the gap between the best and the worst result and finding the distribution of the results found.

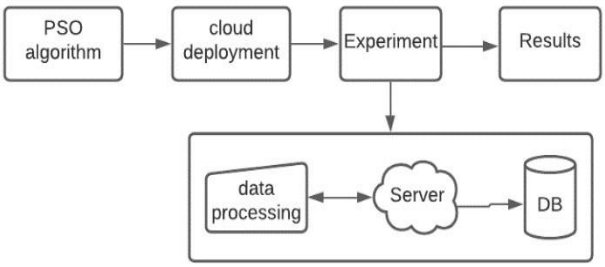

Fig. 7. Methodology used.

2.1 Swarm of Particles

For the creation of the algorithm, no specific animal behavior is considered, the optimization algorithm will be carried out in a generalized way so that the most important thing was to distribute each of the particles proportionally without leaving significant unexplored gaps, to achieve this it is sectioned -will be the universe of possible solutions in n subsets and in each of them a particle is randomly generated, in this way it was tried to control the minimum and maximum possible of unexplored areas, that is to say, that the randomly generated particles are distributed almost uniformly between the universe of possible positions.

Figure 7 graphically shows the distribution generated where each of the particles is found within the sections that limits the maximum distance that can exist between one particle and another, in terms of the minimum it was not relevant to this study.

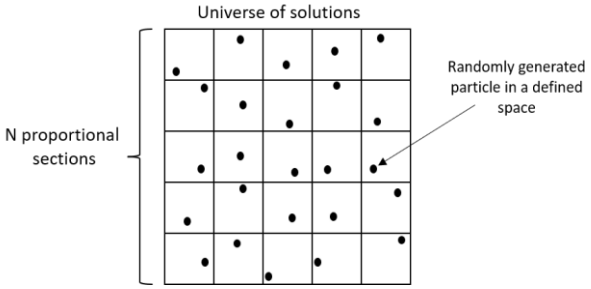

Fig. 8. Generación de partículas en diferentes secciones.

The selected instance includes 20 machines and 50 jobs, which generates a total of 2.02x10919 of possible assignments, of which the generated algorithm will only evaluate 10,000 of them. The first 500 particles are randomly generated within the main sections, depending on the best result.

The 499 remaining particles are transferred to the best section partitioned again, at this point, there are already 1,000 evaluations. The process continues 20 times. Figure 9 shows how this procedure is applied.

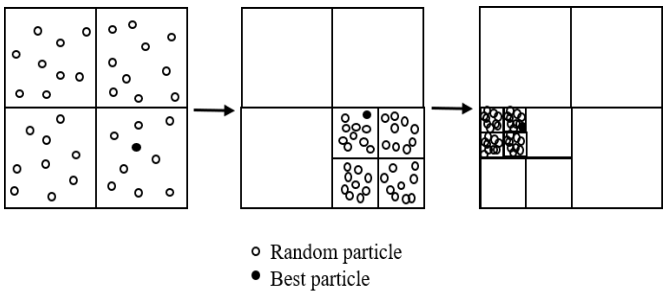

Fig. 9. Search for the best particle by sections.

2.2 Algorithm Distribution

The algorithm in the physically distributed computers, a web server, was used, which was accessed through the domain https://www.jsspo.com, where the particle swarm algorithm was stored. The collaboration continued as the machine had to access the site where it would find the home screen, and automatically, the algorithm was executed indefinitely. During the execution, the system accessed the database to update the best data in units of time.

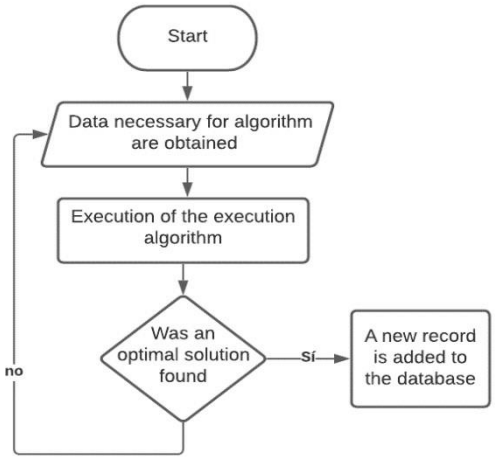

Fig. 10. Web system execution process.

The activity of obtaining the data consists of a client-server interaction where the client will load the variables defined for the problem to be solved when entering the web page. The necessary code to start the second activity executes the algorithm. All the resources for processing are provided to each computer. Additionally, it is possible to execute the algorithm without internet access as long as All the data provided by the first activity that does require internet access is loaded. Accessing the database occurs by adding a new record when finding different data.

3 Experiment

Three hundred computers were used to execute the algorithm simultaneously with their own resources such as processor and RAM memory, each computer connected to the Internet obtained the necessary data from the web page where the algorithm was stored. The information obtained included the initialization variables for the assignment problem used and the code to work with the data automated. The database used was SQL and the MySQL database manager. The hosting plan allowed unlimited bandwidth and storage of up to 50GB to avoid complications of access to the site and data saturation. When entering the page, an interface was shown that shows a message

thanking the cooperation for optimizing the problem. As seen in figure 10, in the lower central part, a counter is shown that increases are indicating that the algorithm is running correctly.

Fig. 11. Web system interface.

The computers used were executed in 5 hours simultaneously. The computers had different hardware characteristics such as processing speed, random access storage memory, and hard disk drive, but our architecture allowed us to work without problems with computers of different characteristics. Table 2 shows a summary of the characteristics of the computer equipment.

Table 1. Used computer equipment.

Number of computers	Processor speed	Memory RAM
30	2.3 Ghz	4 Gb
90	2.4 Ghz	4 Gb
70	2.7 Ghz	4 – 8 Gb
110	2.8 Ghz	8 – 12 Gb

Each computer entered the web page with the system providing the code processed by each PC. The client and server interaction consisted in loading all the processing on the computers and only communicating to perform queries in the database. This process is shown in figure 11.

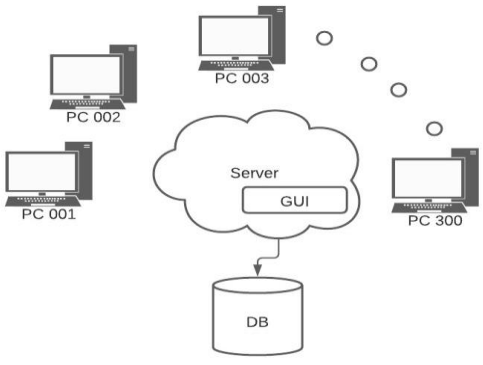

Fig. 12. Working srchitecture.

4 Results

At the end of the experiment, 25,795 records were obtained with different solutions to the proposed problem. Table 2 shows summary data in the experiment.

Table 2. summary of data obtained.

Description	Quantity
Duration of the experiment	5 hr
Total data obtained	25803
Time interval	1159
Best time	15022
Wrost time	16181

The results obtained by the same algorithm executed in different distributed machines obtained a maximum slack of 1159 time units, which shows that the execution of the same optimization algorithm in different computers can yield different results. It was detected that the results obtained are distributed in a non-proportional way.

That is, there is a higher concentration of specific solutions as shown in figure 12, most of the solutions found are between 15800 and 16000, and in lesser quantity the higher than 16000 and finally, in a minor appearance, the results that are close to the most negligible data that is the optimal solution sought by the algorithm.

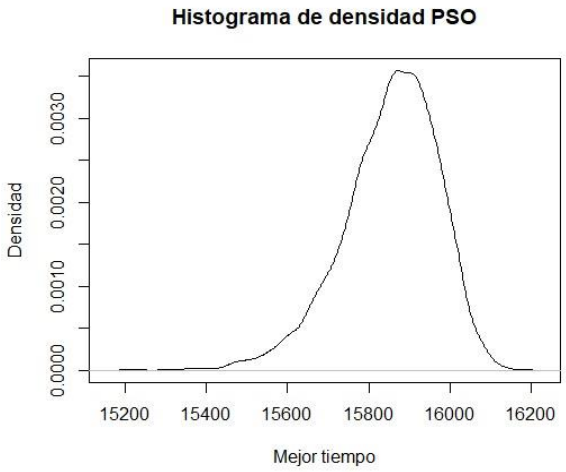

Fig. 13. Distribution of results.

A study was carried out to identify the distribution in which the data obtained were found, performing the KS test (Kolmogorov-Smirnov), an error of only 0.0237 was obtained with the Weibull 0.0507 with the normal distribution and 0.0518 with the gamma distribution. In figure 14, the agreement with the Weibull distribution is shown.

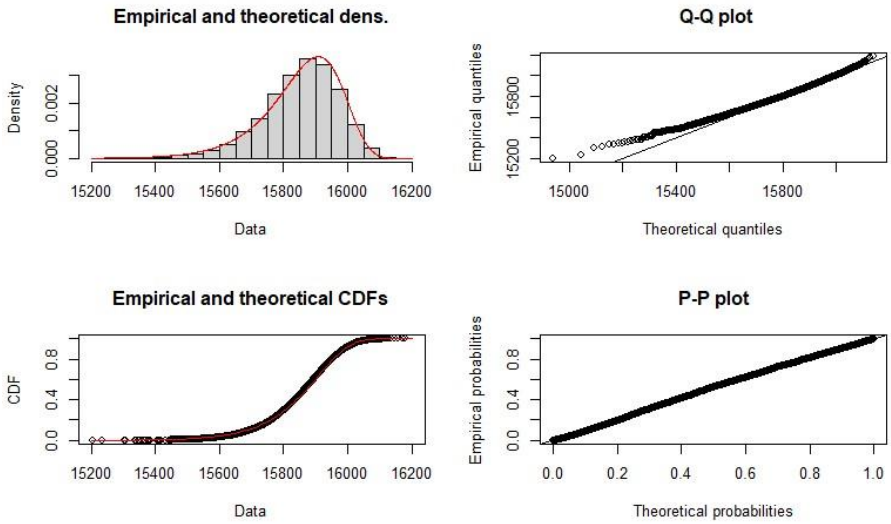

Fig. 14. Weibull distribution.

The best data is at a distance of 831time units, and the probability of solving is less than 1%. Table 3 shows a summary of statistics.

Table 3. Final statics.

Statics	Data
mean	15853.26
median	15866
confidence interval 90%	15851.38-15855.13
confidence interval 95%	15851.03-15855.49
confidence interval 99%	15850.32-15856.20

5 Conclusions

The combined techniques such as distributed computing and optimization algorithms such as the swarm of particles have yielded promising results, showing a significant difference between the different results obtained. Most results obtained are concentrated in a range with slight variation, and the most optimal results take time to appear. That time is due to the algorithm's structure, searching the objective quickly and leaving solutions out of the search.

Of possible solutions, therefore, it can be concluded that the longer the algorithm's execution time and the greater the number of computers used, the greater the probability of finding a more optimal result. However, it can also be indicated that no

matter how long the PSO algorithm is executed to solve JSSP problems, it does not ensure that the best possible solution is found.

6 Future Work

It is proposed to carry out an optimization algorithm consisting of a package of various algorithms such as PSO, simulated annealing, and ant colony for future work. The package will be hosted on the web, where it is expected to have greater cooperation of computers with those used in this experiment. Each of the computers will apply one of the three proposed algorithms and will look for the best solution for a flexible job shop problem. The experiment intends to find differences between the results found by each algorithm and demonstrate that it is convenient to combine distributed computing techniques with various optimization algorithms.

References

1. Ricardo Alvarez, Nick Sims, Christian Servin, Martine Ceberio, Vladik Kreinovich: If Space-Time Is Discrete, It Could Be Possible to Solve NP-Complete Problems in Polynomial Time. Int. J. Unconv. Comput. 15(3): 193–218 (2020)
2. Błażewicz, J., Domschke, W., Pesch, E.: The job shop scheduling problem: Conventional and new solution techniques. European Journal of Operational Research, 93(1), pp. 1–33 (1996)
3. Klaus Jansen, Roberto Solis-Oba, Maxim Sviridenko: Makespan Minimization in Job Shops: A Polynomial Time Approximation Scheme. In: STOC, pp. 394-399 (1999)
4. Wang, D., Tan, D., Liu, L.: Particle swarm optimization algorithm: an overview. Soft Comput, 22, 387–408 (2018) https://doi.org/10.1007/s00500-016-2474-6
5. Bai, Q.: Analysis of Particle Swarm Optimization Algorithm. Computer and Information Science, 3 (2010) doi: 10.5539/cis.v3n1p180.
6. Khan, R. Z.: Distributed Computing: An Overview. Int. J. Advanced Networking and Applications. 07 (2015) 2630–2635.
7. Callaghan, M., Harkin, J., El-Gueddari, M., Mcginnity, T.M., Maguire, L.: Client-server architecture for collaborative remote experimentation. Information Technology and Applications, International Conference on. 2. 125−129 vol. 2 (2005) doi: 10.1109/ICITA.2005.97.