COLLISION OF SIMILAR PARTICLES FOR THE PROBLEM OF SCHEDULE OF PRODUCTION IN MACHINERY WORKSHOP

Tatiana Balbi Fraga (UFPE)
tatiana.balbi@ufpe.br
Antonio Jose da Silva Neto (IPRJ-UERJ)
ajsneto@iprj.uerj.br

In this work, a new heuristic is proposed to solve the Machine Shop Production Scheduling Problem (PEPOM), called Similar Particle Collision. This heuristic is an adaptation of the Particle Collision Algorithm, originally developed for solving continuous optimization problems.
To test the application of the heuristic Similar Particle Collision to PEPOM, a new algorithm is presented, called Multi Similar Particle Collision with Exploration by Tabu Search, where a Tabu Search algorithm is used as a local exploration operator and a mutation operator is used to perturbation of the solution.
As a result, this work demonstrates that this new algorithm is able to attenuate the sensitivity of parameter adjustments and avoid cyclical processes, typical of Busca Tabu.

Keywords: Production Scheduling Problem, Particle Collision, Tabu Search, combinatorial optimization

1. Introduction

The machine shop production scheduling problem (PEPOM) has received in recent decades a strong attention from researchers around the world. One of reasons for this intense interest is due to its broad and relevant applicability as this problem appears in the day to day of industries of the most different segments and the way it is treated usually has a strong impact on production costs. The other reason is in its high complexity. PEPOM is classified as NP-difficult (LENSTRA *et al.,* 1977) and regarded as one of the most difficult combinatorial optimization problems so far. studied. Briefly, the PEPOM can be presented as follows: considered a set of machines m $\{M_i\}_{i=1}^{m}$ and a set of tasks n $\{J_j\}_{j=1}^{n}$ which must be processed once, and only once, on each element of the set of machines in a known order, which may vary from one task to another. In this way, each task can be understood as a sequence of operations, $J_j = \left(O_{k_i j}\right)_{i=1}^{m}$, where $O_{k_i j}$ is the operation of the task J_j that must be processed by the machine M_{k_i} during processing time known and deterministic $t_{k_i j}$, $t_{k_i j} \geq 0, k_i = 1, \ldots, m$. Additionally: each machine can process a single task at a time; each task cannot be processed for two or more machines at the same time; and preemption is not allowed. The purpose of this problem consists of determining the order in which jobs should be processed on each machine. in order to minimize the total time required to process the group of tasks, in English referred to as *makespan,* without however disrespecting the restrictions imposed by the problem.

Since the beginning of the 1950s, a series of exact and approximation algorithms have been proposed in the literature to solve the PEPOM (BÿAÿEWICZ *et al.,* 1996; JAIN and MEERAN, 1999). Among the exact algorithms, those that obtained the best results were the algorithms built based on the Branching and Pruning methods (BRUCKER *et al.,* 1994; CARLIER and PINSON, 1989). However, despite technological advances and obtained in the evolution of these algorithms, even today, its application to larger problems is computationally infeasible. Among the approximation algorithms, the algorithms built based on Local Search heuristics, such as Tabu Search (DELL'AMICO and TRUBIAN, 1993; TAILLARD, 1994; NOWICKI and SMUTNICKI, 1996; ZHANG *et al.,* 2007), Simulated Annealing (VAN LAARHOVEN *et al.,* 1992; STEINHÖFEL *et al.,* 2003; AZIZI and ZOLFAGHARI, 2004), Genetic Algorithms (CHENG *et al.,* 1996 and 1999; PARK *et al.,* 2003; QING-DAO-ER-JI and WANG, 2012) and, more recently, Crowd Particle Optimization (LIAN *et al.,* 2006; SHA and HSU, 2006; LIN *et al.,* 2010). The problem with these algorithms is that, although they are capable of producing good solutions quickly, it is not guaranteed that optimal solutions will be found for all test problems presented in the literature. Additionally, these algorithms are very sensitive both to the initial solutions and to the adjustment of their parameters.

In this work, a new heuristic for PEPOM solution called Collision of Similar Particles, inspired by the Collision of Particles Algorithm created by Sacco and Oliveira (2005) and until then applied to the solution of optimization problems of a continuous nature (SACCO *et al.,* 2006; SACCO *et al.,* 2008; KNUPP *et al.,* 2009). For heuristic testing here proposal is presented a new algorithm for PEPOM solution, called Multi Collision of Similar Particles with Exploration by Tabu Search (MCPS-BT), where the Search method Tabu by Nowicki and Smutnicki (1996), without backpropagation, is applied as a local exploration, and the mutation operator defined by Lian *et al.* (2006) as M7 with repositioning of only one operation, M7 (1opt), is used for perturbing the solution. The performance of this algorithm is tested through the well-known test problems of the families FT (FISHER and THOMPSON, 1963) and AZT (ADAMS *et al.,* 1988).

The next sections are organized as follows: in the next section the Collision algorithm of Particles, as figured by Sacco *et al.* (2006) for continuous problem solving is presented, as well as the proposed adaptations for the PEPOM solution; the section 3 introduces the MCPS-BT algorithm and introduces the M7 (1opt) and Tabu Search operators that are used, respectively, for pertubation and local exploration of the solution in that same algorithm; in section 4 some of the computational results are presented as well as an analysis of these results; and, finally, section 5 presents the considerations final results and some proposals for future work.

2. Similar Particle Collision

By observing the physics of the collision process between nuclear particles inside a reactor nuclear power it is possible to verify that among the colliding particles, those that reach the nuclei, i.e., regions of high fitness, are absorbed and explore the surroundings while that those that reach regions of low fitness can be absorbed or spread to other regions according to some probability function. Analyzing these interactions, Sacco and Oliveira (2005) observed that the succession of absorption and scattering events allowed the movement of particles to promote an exploration of the search space complete while a deeper exploration of the most promising. Thus, based on this observation, the authors proposed a new algorithm for search which they named the Collision of Particles Algorithm, or PCA (from the English *Particle Collision Algorithm).* In this algorithm initially a solution is chosen and reserved as current solution. Then, this current solution is perturbed through a Perturb() function, generating a new solution. If the new solution is “better” than the current solution (of according to some previously defined criteria), the last solution is replaced by the first and on this a local search procedure defined by a function is applied Exploration(). If the new solution is “worse” than the current solution, then a function Scatter(), where a probability function is used to define whether the current solution will be explored or replaced by a random solution. Note that this algorithm has a similar structure to Simulated Annealing except for the fact that the algorithm can be taken to new search spaces not directly related to the initial solution. Additionally, it is not necessary to define an initial temperature. A Fig. 1 presents a pseudo-code for the PCA, as proposed by Sacco *et al.* (2006) for solving continuous maximization problems. Note that in this algorithm the function of probability is defined by the criterion of Metropolis (METROPOLIS *et al.,* 1953). Figs. 2 and 3 present, respectively, the functions Disturbance() and Small_Disturbance() applied in the same algorithm. In these figures it is possible to observe that, when working with continuous variable problems, perturbations are generated through variations random changes in the value of each variable within previously established limits, with the small perturbations similar to the perturbations, differing only in the limits that are more narrow. In the case of discrete problems like PEPOM, it is necessary to define how these disturbances will take place.

Fig. 4 presents a general scheme for the Similar Particle Collision heuristic where the functions, Disturbance() and Exploit() are respectively replaced by operators of

Disturbance and Local Exploration. As Disturbance operators, whose purpose is to take the algorithm to new search spaces, the same operators are suggested as mutation used in Genetic Algorithms, such as operators M1 to M10 presented by Lian *et al.* (2006). Alternatively, if the Particle Collision algorithm

Similar is applied simultaneously to all individuals of an initial population, it can be

Figure 1 – Pseudo-code for the Particle Collision Algorithm

```
Gere uma solução inicial Solução_Atual
Melhor_Aptidão = : Aptidão (Solução_Atual)
De n = 1 até # iterações
        Pertubação ()
        Se Aptidão (Nova_Solução) > Aptidão (Solução_Atual)
                Se Melhor_Aptidão > Aptidão (Nova_Solução)
                        Melhor_Aptidão = : Aptidão (Nova_Solução)
                Fim Se
                Solução_Atual = : Nova_Solução
                Exploração ()
        Se não
                Espalhamento ()
        Fim Se
Fim De

Exploração ()
        De n = 1 até # iterações
                Pequena_Pertubação ()
                Se Aptidão (Nova_Solução) > Aptidão
(Solução_Atual)
                        Solução_Atual = : Nova_Solução
                Fim Se
        Fim De
retorna

Espalhamento ()
        Probabilidade_Espalhamento = 1 – Aptidão (Nova_Solução) / Melhor_Aptidão
        Se
                Probabilidade_Espalhamento > valor ranômico (0, 1)
                Solução_Atual = : solução randômica
        Se não
                Exploração ()
        Fim Se
retorna
```

if you use the same crossover operators as in Genetic Algorithms, such as example operators C1 to C4 also presented by Lian *et al.* (2006), about peers of solutions. In the case of Local Exploration operators, whose purpose is to explore the surroundings of a given solution, it is suggested the various algorithms built based on in local search heuristics, such as Simulated Annealing, Genetic Algorithms,

Figure 2 - Disturbance function ()

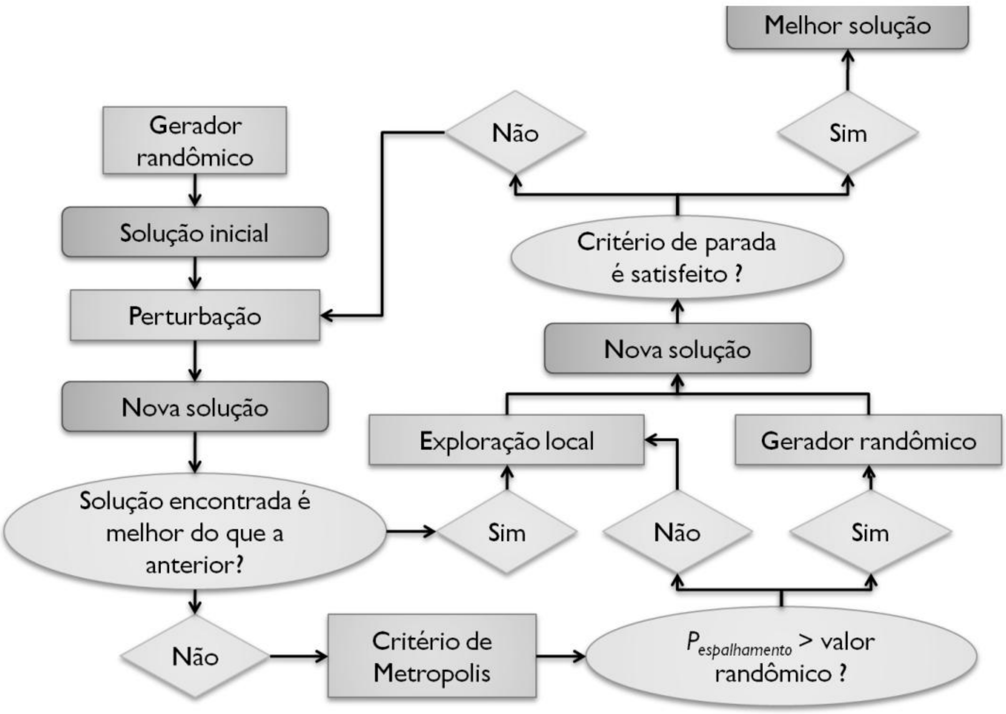
```
disturbance ()
          From i = 0 to (Dimension – 1)
                    Sup = : Upper Limit [i]
                    Inf = : Lower Limit [i]
                    Rand = : Radomic (0.1)
                    New_Solution = : Current_Solution [i] + ((Sup - Current_Solution [i])*Rand) –
                                        ((Current_Solution [i] – Inf)*(1-Rand))
          End of
          If New_Solution [i] > Sup
                    New_Solution [i] = : SupLimit [i]
          If not
                    If New_Solution [i] < Inf
                              New_Solution [i] = : InfLimit [i]
                    End if
          End if
Returns
```

Particle Collision and Tabu Search.

```
Small_Disturbance()
          From i = 0 to (Dimension – 1)
                    Sup = : Radomic (1.0,1.2)* Current_Solution [i]
                    If (Sup > Upper Limit [i])
                              Sup = : Upper Limit [i]
                    End if
                    Inf = : Radomic (0.8,1.0)* Current_Solution [i]
                    If (Inf > Lower Limit [i])
                              Inf = : Lower Limit [i]
                    End if
                    Rand = : Radomic (0.1)
                    New_Solution = : Current_Solution [i] + ((Sup - Current_Solution [i])*Rand) –
                              ((Current_Solution [i] – Inf)*(1-Rand))
          End of
Returns
```

Figure 3 – Small_Disturbance function.

Figure 4 – Scheme for the Similar Particle Collision heuristic.



3. Multiparticle Collision with Exploitation by Tabu Search

To test the Similar Particle Collision heuristic (Fig. 4), this section presents a new algorithm for PEPOM solution called Similar Multiparticle Collision with Exploration by Busca Tabu (MCPS-BT). In this algorithm, the particle collision heuristic Similar is applied on a set of randomly generated initial solutions and in that case it is defined as Multi Particle Collision Similar. As a disturbance operator the mutation operator M7 is applied with repositioning of just one operation, M7 (1opt) (LIAN *et al.,* 2006), and as a local exploration operator, a version is applied deterministic of the Tabu Search method by Nowicki and Smutnicki (1996), without backpropagation. In sections 3.2 and 3.3 these two operators will be presented in more detail. One since the PCA was originally developed for maximization problems (Fig. 1), some changes are necessary for the treatment of PEPOM, these changes are presented in section 3.4. The next section discusses how to represent the solutions adopted in this work.

3.1. Representation of solutions

During the development of an algorithm to solve the PEPOM, one of the questions keys is the choice of how the solutions will be represented. Among the various forms that appear in the literature, the Task Processing Code (CPT) has the advantage of representing only viable solutions and for this reason it was adopted in this work. In CPT a solution is represented by a sequence containing $n \times m$ digits. In this sequence, the index corresponding to each task, $(i, i = 1 \ldots n)$ is repeated m times, that is, by the number of operations of the respective task. Thus, a given operation is defined by the index of its respective task and by the position in which this index appears. Per example, in Fig. 5 presents a solution represented by a CPT for a PEPOM with 3 tasks and 3 machines. In this sequence, the first digit represents the first operation of the third task, the second digit represents the first operation of the second task, the third digit represents the second operation of the third task, and so successively.

Figure 5 – CPT for a PEPOM with 3 tasks and 3 machines.

CPT = (3,2,3,1,2,2,1,3,1)

In the CPT, the position of each operation determines the order of priority in which it must be processed so that the sequence in which tasks are to be processed in each machine is determined by taking the operations to their respective machines in the same order in which they appear in the solution given by the CPT. One of the inconveniences of this form of representation is that two apparently distinct solutions, when traced by the CPT, can portray the same solution (sequence of tasks per machine).

3.2. M7 mutation operator (1opt)

The mutation operators were introduced by the various Genetic Algorithms in order to to simulate the changes in genes that occur from time to time in certain individuals, defined in Darwin's model of natural evolution as mutations. Those operators have as main purpose to promote a significant disturbance in the solutions, forcing the algorithms to scour new search spaces, thus fusing of local minima. Therefore, these operators present themselves as good substitutes for the Disturbance() function used in the PCA originally developed to solve ongoing problems. Among all the mutation operators that are applied in PEPOM solution algorithms, the operator M7 (1opt) was chosen in this work in function of its simplicity. In this operator, initially two positions are chosen randomly along a solution represented by the CPT, the first position being defined as the movement point and the second as the insertion point. then the operation placed at the point of movement is removed from this point and repositioned at the point insertion point (Fig. 6).

Figure 6 – M7 mutation operator (1opt).

insertion point

current solution CPT = (3,2,3,1,2,2,1,3,1)

movement point

Solution after CPT mutation = (3,2,1,2,2,1,3,3,1)

3.3. Tabu Search Operator

Among the various procedures based on local search heuristics, the algorithm Busca Tabu presented by Nowicki and Smutnicki (1996), with neighborhood structure generated by method N5, is still considered one of the techniques that present the best results, in terms of efficiency and effectiveness, for the PEPOM solution, and for this reason the local exploration operator applied in this work was developed based on this algorithm. This operator can be briefly described as follows: initially the initial solution is defined as the current solution and the generation operator is applied to it. neighborhood N5, generating small disturbances that will give rise to a set of new solutions defined as neighborhood. Among the neighborhood elements, the best one is chosen. solution (solution with lowest *makespan)* not belonging to a tabu list (list containing references of solutions that should not be chosen). So this solution is defined as current solution, on which new disturbances are performed generating a new neighborhood, it is

A new selection is made, and the process continues until a previously specified stopping criterion is met. established is reached. In the next sections the N5 neighborhood generation operator, the process of selecting the best solution along with the tabu list, and the stopping criterion applied in this algorithm will be described in more detail.

N5 neighborhood generation operator

Given any solution (sequence of tasks per machine), a critical path can be defined as a sequence of operations where:

- the last operation in the order of this sequence is the operation that is finished last and which, therefore, has final processing time equal to *makespan;*
- The operation preceding every operation, except the first, in the order of this sequence is the operation that precedes it in the machine order or in the job order, being between the two the one with the longest final processing time;
- the starting time of processing the first operation in the order of this sequence is equal to zero.

It is important to note that a solution can contain more than one critical path, such as in the case where there are two or more operations that are finalized last, or when the operations that precede a given operation in machine and job orders are finished at the same moment. In their article, Nowicki and Smutnicki (1996) state that the choice of the critical path does not have a strong influence on the final result and the authors suggest that the even if chosen randomly. In this work, how do you want the operator to Tabu Search presents itself as a deterministic method, the critical path is selected from according to the following priorities:

- if two operations are finished last, the operation chosen will be the one belonging to a lower index machine;
- if the operations that precede a given operation in the machine and service orders are completed at the same time, the operation chosen will be the one preceding the Order of Service.

The critical path can also be subdivided into blocks of operations where each block is composed of the subsequence of successive operations that must be processed by a same machine. By the N5 neighborhood generation method, a neighborhood is formed by swap between the first two and the last two operations of each block of operations in the critical path, and in the case of the first block, only the last two operations are swapped and in the case of the last block only the first two operations are swapped. Case any block is formed by only one operation, no exchange will be performed and, if the even if it consists of two operations, only one exchange will be performed.

Selection of the best non-tabu solution

In the Tabu Search operator applied in this work, the tabu list consists of a vector containing the movements inverse to those used in the generation of the t latest current solutions (where t is a parameter whose value can be adjusted). Thus, the best neighbor (solution chosen as the new current solution) will be the solution with the smallest *makespan* whose movement generator is not in the tabu list. Additionally, an aspiration criterion is considered here. that allows the replacement of the current solution by a solution whose generating movement belongs to the tabu list if the *makespan* of this solution is smaller than the *makespan* of the best solution hitherto found.

stopping criterion

In this algorithm, the iterative process is interrupted when during $Nite$ iterations the best solution found is not updated, that is, no new current solution presents less *makespan* than the *makespan* of the best solution found. Being $Nite$ a parameter whose value can be adjusted.

3.3. probability function

As previously mentioned, the Particle Collision Algorithm was originally developed for solving maximization problems. In the case of PEPOM, as seeks to minimize the value of the objective function defined by *makespan,* the Metropolis criterion should be changed giving the scattering probability function the following form:

$$\text{Probability\_Scatter} = 1 - \frac{\text{Melhor\_Aptidão}}{\text{Aptidão (Nova\_Solução)}}$$

Being the value of the variable Melhor_Aptidão defined by the *makespan* from the best solution to then found and the value of the variable $\text{Aptidão (Nova\_Solução)}$ defined by the *makespan* of analyzed solution. In this way the probability of scattering will increase the measure by that the *makespan* of the analyzed solution deviates from the *makespan* of the best solution and will be null if these two values are equal.

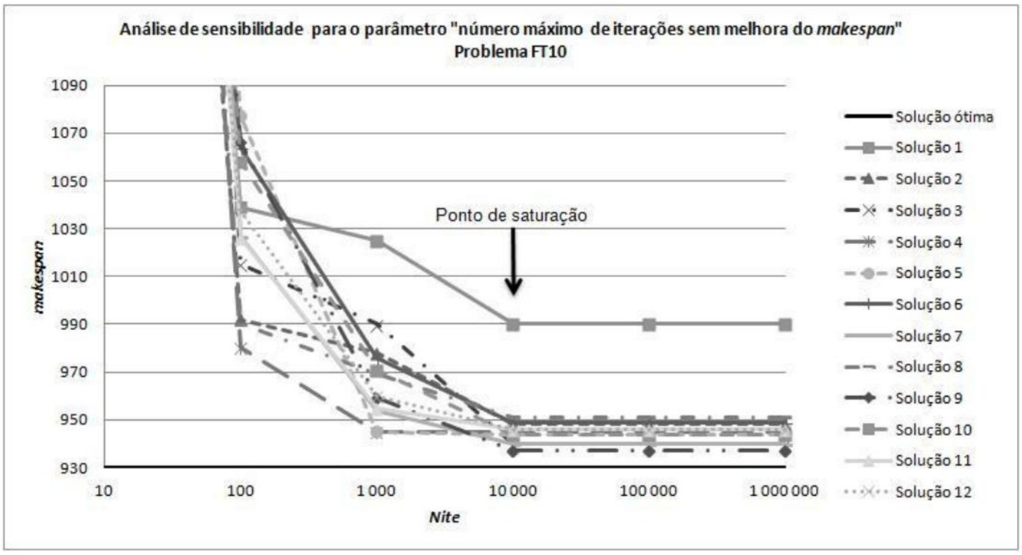
4. Results and discussions

To test the MCPS-BT algorithm, proposed in this work, the problems FT6 (with 6 machines and 6 tasks), FT10 (with 10 machines and 10 tasks) and FT20 (with 5 machines and 20 tasks) proposed by Fisher and Thompson (1963) and problems ABZ5 and ABZ6 (with 10 machines and 10 tasks), and ABZ7, ABZ8 and ABZ9 (with 15 machines and 20 tasks) proposed by Adams, Balas and Zawack (1988). For each problem were generated randomly 12 initial solutions and they were used as initial solutions for all applied tests. The test problems of the FT and ABZ families as well as the tables containing the results of all the tests discussed here can be consulted at Fraga's doctoral thesis (2010).

Initially, a sensitivity analysis of the Busca Tabu algorithm was performed, used as a local perturbation operator, with respect to the parameters: maximum number of iterations no *makespan* improvement $(Nite)$; and tabu list size. From this (t) analysis it was observed that this algorithm, when applied alone, is not robust, since its effectiveness depends on the stipulated value for the parameter t, whose adequate value will depend on each problem specific. Different values measured a result in t different results. It was also verified that, if this parameter is not properly adjusted, the algorithm goes into cycles and, from a certain value defined here as saturation point (PS), no parameter increase $Nite$ is able to improve the result. It was also verified that the order of the value of this saturation point can be directly associated with the size of the problem.

In Fig. 7 presents an analysis of the variation of the *makespan* with the increase in the value of parameter $Nite$ for problem FT10 where it is verified that for this problem test $\mathrm{PS} \sim 10.000$. The same saturation point is checked for problems ABZ5 and ABZ6, the which feature the same number of machines and tasks as the FT10. For the FT20 problem it was verified that $\mathrm{PS} \sim 1.000$ and for problems with 15 machines and 20 tasks (ABZ7, ABZ8 and ABZ9) it was verified that $\mathrm{PS} \sim 100.000$. In the case of the FT6 problem, all solutions converged to global optima with very few iterations, so that this problem was used only for validation of the algorithms presented in this work.

Figure 7 – Sensitivity analysis of the *makespan* to the value of the parameter "maximum number of iterations without improvement of the *makespan"* of the Tabu Search algorithm for the FT10 problem (tabu list size = 8).



The inconveniences of the Tabu Search algorithm highlighted here, ie related to the point of saturation and the sensitivity of the algorithm to the value of the parameter "size of the tabu list", were previously commented by Nowicki and Smutinicki (1996) in the presentation of their algorithm Tabu search with neighborhood structure generated by the N5 technique. To overcome this difficulty the authors proposed the use of a backpropagation technique through which the cycles are identified, causing the algorithm to exit the infinite *loop* and return to a of the best solutions so far visited. The disadvantage of this technique is the significant increased algorithm complexity and memory usage. Additionally this technique does not guarantee that optimal solutions are found and does not significantly decrease the sensitivity of the algorithm to the adjustment of the "tabu list size" parameter.

For convergence analysis of the MCPS-BT algorithm, a sensitivity analysis was generated from *makespan* to the variation of the "number of cycles" parameter. In the case of FT10 problems, FT20, ABZ5 and ABZ6, it was verified that the algorithm converged to the optimal solution in all rounds with this conversion being a bit slower for ABZ5 issues and ABZ6. In the case of problems ABZ7, ABZ8 and ABZ9, the convergence test extrapolated the established maximum processing time (1 week) without even finishing the first round results. These results reflect the complexity that exists in the solution of the PEPOM discussed in the introduction to this paper.

Finally, to analyze the robustness of the MCPS-BT algorithm, an analysis of parameter sensitivity of the Tabu Search local operator parameter. Once

algorithm has random nature, each test was performed 5 times. As a result verified that, regardless of the value set for this parameter, almost all solutions generated were better than the best solution found by the simple search algorithm application Tabu, demonstrating the superiority of the developed algorithm. Despite being repaired a small fluctuation of the values found, it was verified that this fluctuation is due to more to the random nature of the algorithm than the variation of the analyzed parameter value.

Through the tests previously described, it was possible to verify that the random component of the Similar Particle Collision heuristic allows the algorithm to be driven to new search spaces, overcoming the difficulties encountered by the Tabu Search algorithm and thereby producing better and more robust results at a computational cost equivalent. Tab. 1 presents a comparison between the results found by the algorithm by Nowicki and Smutinick (1996) with backpropagation (TSAB), and the best results found for the MCPS-BT, for each of the considered test problems. As can be seen in this table, MCPS-BT was able to find solutions optimal for problems FT20, ABZ5 and ABZ6, outperforming the TSAB algorithm. The authors do not carried out tests for problems ABZ7, ABZ8 and ABZ9 so that in this case it was not comparison possible.

Table 1 – Comparison between the results presented by Nowicki and Smutnicki for their TSAB algorithm and the best results obtained by the MCPS-BT algorithm, for the FT and ABZ family problems

Problema	Solução ótima	TSAB	MCPSEBT
FT6	55	55	55
FT10	930	930	930
FT20	1165	1178	1165
ABZ5	1234	1238	1234
ABZ6	943	945	943
ABZ7	656	***	660
ABZ8	645	***	671
ABZ9	661	***	684

5. Conclusions and Future Work

In this work, a new heuristic for solving the problem of Production Scheduling in Machine Shop called Similar Particle Collision (CPS). To test this heuristic, a new algorithm called Multi Collision of Similar particles with Exploration by Tabu Search where the mutation operator M7(1opt) and Nowicki and Smutnicki's Tabu Search algorithm, without backpropagation, were applied as perturbation and local exploration operators, respectively. As a result it was demonstrated that the perturbation and scattering techniques of CPS lead the algorithm to new search spaces, preventing the solution from running aground in local minima as it happens with the Algorithm Search Tabu considered in this work, when applied alone. For this reason, the CPS heuristic was able to present better and more robust results for all analyzed test problems, for an equivalent computational cost.

Since all tests were performed with a set containing 12 initial solutions fixed it would be necessary to test the efficiency of the algorithm for different initial solutions like this how to perform sensitivity analyzes for the initial population size. additionally new perturbation and exploration operators can be tested seeking to improve the algorithm efficiency. These tests remain as recommendations for future work.

Acknowledgments: This work was possible thanks to the financial support offered by

CAPES, Coordination for the Improvement of Higher Education Personnel.

REFERENCES

ADAMS, J., BALAS, E., ZAWACK, D. **The shifting bottleneck procedure for job-shop scheduling.** Management Science, 34, p. 391-401. 1988.

AZIZI, N., ZOLFAGHARI, S. **Adaptive temperature control for simulated annealing: a comparative study.** Computers & Operations Research, 31, p. 2439-2451. 2004.

BÿAÿEWICZ, J., DOMSCHKE, W., PESCH, E. **The job shop scheduling problem: Conventional and new solution techniques.** European Journal of Operational Research, 93, p. 1-33. 1996.

BRUCKER, P., JURISCH, B., SIEVERS, B. **A branch and bound algorithm for the job shop scheduling problem.** Discrete Applied Mathematics, 49, p. 107-127. 1994.

CARLIER, J., PINSON, E. An **algorithm for solving the job-shop problem.** Management Science, 35 (2), p. 164-176. 1989.

CHENG, R., GEN, M., TSUJIMURA, Y. **A tutorial survey of job-shop scheduling problems using genetic algorithms – I. Representation.** Computers & Industrial Engineering, 30 (4), p. 983-997. 1996.

CHENG, R., GEN, M., TSUJIMURA, Y. **A tutorial survey of job-shop scheduling problems using genetic algorithms: Part II. Hybrid Genetic Search Strategies.** Computers & Industrial Engineering, 37, p. 51-55. 1999.

DELL'AMICO, M., TRUBIAN, M. **Applying tabu search to the job-shop scheduling problem.** Annals of Operations Research, 41, p. 231-252. 1993.

FISHER, H., THOMPSON, GL **Probabilistic learning combinations of local job-shop scheduling rules.** Muth, JF, Thompson, GL (eds.). Industrial Scheduling, Prentice Hall, Englewood Cliffs, New Jersey. 1963.

FRAGA, TB **Proposition and analysis of hybrid models for the production scheduling problem in a machine shop.** Doctoral thesis. Nova Friburgo, RJ: State University of Rio de Janeiro - Polytechnic Institute. 2010.

JAIN, AS, MEERAN, S. **Deterministic job-shop scheduling: Past, present and future.** European Journal of Operational Research, 113, p. 390-434. 1999

KNUPP, DC, SILVA NETO, AJ da, SACCO, WF **Radiative properties estimation with the particle collision algorithm based on a sensitivity analysis.** High Temperatures – High Pressures, 38, p. 137-151. 2009.

LENSTRA, JK, RINNOOY KAN, AHG, BRUCKER, P. **Complexity of machine scheduling problems.** Annals of Discrete Mathematics, 1, p. 343-362. 1977.

LIAN, Z., JIAO, B., GU, X. **A similar particle swarm optimization algorithm for job-shop scheduling to minimize makespan.** Applied Mathematics and Computation, 183, p. 1008-1017. 2006.

LIN, T.-L., HORNG, S.-L., KAO, T.-W., CHEN, Y.-H., RUN, R.-S., CHEN, R.-J., LAI, J.-L., KUO, IH. **An efficient job-shop scheduling algorithm based on particle swarm optimization.** Expert Systems with Applications, 37, p. 2629-2636. 2010.

METROPOLIS, N., ROSENBLUTH, AW, ROSENBLUTH, MN, TELLER, AH, TELLER, E. **Equations of state calculations by fast computing machines.** J.Chem. Phys., 21, p. 1087-1092. 1953.

NOWICKI, E., SMUTNICKI, C., **A fast taboo search algorithm for the job shop problem.** Management Science, 42 (6), p. 797-813. 1996.

PARK, BJ, CHOI, HR, KIM, HS **A hybrid genetic algorithm for the job shop scheduling problems.** Computers & Industrial Engineering, 45, p. 597-613. 2003.

QING-DAO-ER-JI, R., WANG, Y. **A new hybrid genetic algorithm for job shop scheduling problem.** Computers & Operations Research, 39, p. 2291-2299. 2012.

SACCO, WF, LAPA, CMF, PEREIRA, CMNA ALVES FILHO, H. **A metropolis algorithm applied to a nuclear power plant auxiliary feedwater system surveillance tests policy optimization.** Progress in Nuclear Energy, 50, p. 15-21. 2008.

SACCO, WF, OLIVEIRA, CRE de. **A new stochastic optimization algorithm based on a particle collision methaheuristic.** 6th World Congresses of Structural and Multidisciplinary Optimization, Rio de Janeiro, Brazil. 2005.

SACCO, WF, OLIVEIRA, CRE de, PEREIRA, CMNA **Two stochastic optimization algorithms applied to nuclear reactor core design.** Progress in Nuclear Energy, 48, p. 525-539. 2006.

SHA, DY, HSU, C.-Y. **A hybrid particle swarm optimization for job shop scheduling problem.** Computers & Industrial Engineering, 51, p. 791-808. 2006.

STEINHÖFEL, K., ALBRECHT, A., WONG, CK **An experimental analysis of local minimal to improve neighborhood search.** Computers & Operations Research, 30, p. 2157-2173. 2003.

TAILLARD, E. D. **Parallel taboo search techniques for the job shop scheduling problem.** ORSA Journal on Computing, 6 (2), p. 108-117. 1994.

VAN LAARHOVEN, PJM, AARTS, EHL, LENSTRA, JK **Job shop scheduling by simulated annealing.** Operations Research, 40 (1), p. 113-125. 1992.

ZHANG, CY, LI, PG, GUAN, ZL, RAO, YQ **A tabu search algorithm with a new neighborhood structure for the job shop scheduling problem.** Computers & Operations Research, 34, p. 3229-3242. 2007.