Hybrid Metaheuristic for Combinatorial Optimization based on Immune Network for Optimization and VNS

Rodney O. M. Diana
PPGMMC/CEFET-MG
Av. Amazonas 7675
Belo Horizonte, MG, Brazil 30510-000
rodneyoliveira@dppg.cefetmg.br

Sérgio R. de Souza
PPGMMC/CEFET-MG
Av. Amazonas 7675
Belo Horizonte, MG, Brazil 30510-000
sergio@dppg.cefetmg.br

Elizabeth F. Wanner*
PPGMMC/CEFET-MG
Av. Amazonas 7675
Belo Horizonte, MG, Brazil 30510-000
efwanner@gmail.com

Moacir F. França Filho
CEFET-MG
Av. Amazonas 7675
Belo Horizonte, MG, Brazil 30510-000
franca@des.cefetmg.br

ABSTRACT

Metaheuristics for optimization based on the immune network theory are often highlighted by being able to maintain the diversity of candidate solutions present in the population, allowing a greater coverage of the search space. This work, however, shows that algorithms derived from the aiNET family for the solution of combinatorial problems may not present an adequate strategy for search space exploration, leading to premature convergence in local minimums. In order to solve this issue, a hybrid metaheuristic called VNS-aiNET is proposed, integrating aspects of the COPT-aiNET algorithm with characteristics of the trajectory metaheuristic Variable Neighborhood Search (VNS), as well as a new fitness function, which makes it possible to escape from local minima and enables it to a greater exploration of the search space. The proposed metaheuristic is evaluated using a scheduling problem widely studied in the literature. The performed experiments show that the proposed hybrid metaheuristic presents a convergence superior to two approaches of the aiNET family and to the reference algorithms of the literature. In contrast, the solutions present in the resulting immunological memory have less diversity when compared to the aiNET family approaches.

CCS CONCEPTS

•Theory of computation → Scheduling algorithms; Evolutionary algorithms; Theory of randomized search heuristics;

KEYWORDS

Artificial Immune Systems, Immune Network, Evolutionary Algorithms, Scheduling

ACM Reference format:
Rodney O. M. Diana, Sérgio R. de Souza, Elizabeth F. Wanner, and Moacir F. França Filho. 2017. Hybrid Metaheuristic for Combinatorial Optimization based on Immune Network for Optimization and VNS. In *Proceedings of GECCO '17, Berlin, Germany, July 15-19, 2017,* 8 pages.
DOI: http://dx.doi.org/10.1145/3071178.3071269

1 INTRODUCTION

Artificial Immune Systems (AIS) are intelligent methodologies conceived from theories and observations made on the behavior of the immunobiological system of vertebrates [9]. The earliest AIS models were proposed for applications in machine learning and data mining [7]. From the model proposed by [15], for a dynamic job sequencing problem, in which the synergy between AIS and optimization is evidenced, numerous approaches have emerged.

Most optimization-oriented AISs were based on the clonal selection principle, among which we can highlight the CLONALG [10], B-Cell algorithm [18] and the immune network for optimization opt-aiNET [8]. This latter makes use of the immune network theory proposed by [17], with some variations, such as opt-aiNetFS [11] and cob-aiNET [5]. Algorithms based on the immune network theory are recognized by the simultaneous coverage characteristic of different regions of the search space. This behavior refers to a large global optimization capability, an essential feature in metaheuristics.

An important aspect to emphasize is that the algorithms of the aiNET family were originally introduced for the exploration of continuous search spaces. In these, random or pseudo-random changes promoted by somatic hypermutation in solution structures do not tend to lead to major transformations. In contrast, combinatorial optimization problems tend to be very sensitive to random changes, and such a change can trigger transformations in the entire structure of the solution, which rarely leads to improvement without refinements.

*Now at School of Engineering and Applied Science - Aston University - Birmingham UK

GECCO '17, Berlin, Germany

978-1-4503-4920-8/17/07…$15.00
DOI: http://dx.doi.org/10.1145/3071178.3071269

In view of this peculiarity of combinatorial problems, several specific approaches to these problems have been proposed. Historically, trajectory methods have been used with great success in the literature. In general, these methods start from an initial solution and run through the search space in an iterative way, transforming the solution at each iteration, counting on both strategies that lead to local optimal solutions, as well as allowing the search to escape from these local optimal solutions. Considering the global optimization capabilities of AIS and the local exploitation characteristics of the trajectory methods, [13] proposed a variation of opt-aiNET for combinatorial optimization, called COPT-aiNET. Already [14] proposed a hybrid approach, based on the CLONALG and Variable Neighborhood Search (VNS) [21], for solving a job sequencing problem in parallel machines. This last approach presented high performance compared to the algorithms proposed in the literature for the problem. It is widely known in the literature that CLONALG exhibits inferior performance over aiNET approaches (see [8], [9]).

This paper presents a general purpose metaheuristic, based on characteristics of the immune network, of higher performance for combinatorial problems. The proposed metaheuristics is based on the strategies of exploitation of the search space employed by the aiNET family and characteristics of the trajectory methods found in [14]. This new metaheuristic, called VNS-aiNET, makes changes in the internal and external structure of the COPT-aiNET, modifying the behavior of the method, which results in a better search space coverage, but reduces the multimodal power of the same. In order to evaluate the performance of the proposed methodology, it was applied for solving the problem of makespan minimization in unrelated parallel machines with sequence-dependent setup times. The performance and the diversification of the model solutions are compared to the algorithms COPT-aiNet and COB-aiNet[C] [4], both of the aiNet family for combinatorial optimization. Comparisons with the CLONALG approach proposed by [14] and with reference approaches in the literature for the set of benchmark instances used are performed.

This paper presents the following organization: Section 2 discusses some issues related to the COPT-aiNET. In addition, the general structure of the VNS-aiNET algorithm is introduced. Section 3 presents the problem to be dealt for evaluating the proposed approach, as well as the details of the operators that make up the proposed algorithm. Section 4 shows the achieved results and a statistical analysis of them, comparing both the convergence and the diversity of the solutions. Finally, Section 5 presents the final considerations and perspectives for future work.

2 VNS IMUNNE NETWORK FOR COMBINATORIAL OPTIMIZATION

The clonal selection principle, proposed by [3], is a process existing in the adaptive immune system, which is triggered when the organism identifies that it has been invaded by a pathogen. The surface of a pathogen is composed of molecular signatures called antigens. When a defense cell comes into contact with some antigen, the immune response is activated and clonal expansion of cells occurs, producing higher concentrations in cells that have higher affinity with this antigen. These cells secrete antibodies, if they are compatible with the antigens, identify them, and the immune system performs the elimination of them.

The principle of identification of antigens from the clonal expansion is called positive selection. In the immune network theory proposed by [17], it is stated that defense cells, in addition to the identification of antigens, can identify the cells themselves as an invading pathogen and, consequently, suppress these cells. This process is called negative selection. A direct consequence is the autoregulation of the system, since once the immune response has been able to eliminate the pathogens, most of the cells used in this response must be discarded by the system, leaving only the cells that have become part of the immunological memory.

The opt-aiNET algorithm makes use of these concepts to regulate the size of the population, ensuring that the solutions are adequately spread through the search space, thus enabling a greater coverage of the same. For this, the opt-aiNET has a phase of suppression of the cells, performed after maturation of the same by the clonal expansion. At this phase, the solution with the highest affinity to the antigen (fitness) is selected within a region of the search space, and all the cells neighboring a distance (cell affinity/cell) σ_s are eliminated from the population.

The algorithms based on opt-aiNET ([4], [5], [8], [11], [13]) use a simple fitness function, which considers only the values of the objective function to be minimized or maximized. This approach entails limitations to the performance of the algorithm. In case of restricting population size, solutions in local minima can prevent cells that walk to promising regions of the search space may be used in the next generation, since the method gives preference to solutions already stagnated in local minimum, due the best value of the objective function. [13] emphasizes that this feature tends to guarantee multimodality to the method. An alternative to avoid this problem could be to leave the population without a defined size, being autoregulated by the operator, as initially proposed by [8]. However, this approach tends to generate difficulties in very extensive search spaces, in which solutions already stagnant in local minima would being continued to participate in the process of clonal expansion, generating a high computational cost, and, therefore, making the method unfeasible.

To solve these problems, it is proposed here that each cell $c_i \in P_u$, where P_u is the active population, be increased by a maturation factor mat^i_{factor}, which is decremented at each iteration in which the algorithm identifies that the solution does not show improvement in the objective function. This factor should contribute negatively to the fitness function, thus helping to avoid the problems identified above. Cells that reach maximum maturation factor are eliminated from P_u, since they do not continue to contribute to the immune response. The maturation factor also contributes directly to the number of clones generated from c_i in the process of clonal expansion, in addition to defining the degree of mutations performed in the process of somatic hypermutation. This factor also contributes negatively about the multimodality characteristic of the algorithm, making it necessary to perform adjustments, if one wishes to guarantee it. The maturation factor is related to the aging mechanism [6], but, unlike this, it is integrated to the fitness function, thus allowing a faster removal by the suppression operator of cells trapped at local minimums, in addition to influence on

clonal expansion and in the mutation rate. All these consequences and specific details will be described in the following Section 3.

Another change performed in the COPT-aiNet structure is during the clonal expansion process. In COPT-aiNet is proposed that each cell of the population be subjected to consecutive clonal expansions and contractions until the immune response stabilizes. Each expansion is defined as the generation of N_{c_i} clones, each clone being subjected to a number of mutations, inversely proportional to its fitness value. Then, the contraction is performed by selecting the cell, for each cell / clone subset, with the best value of the objective function. Finally, the fitness value is recalculated and the process is repeated until the immune response stabilizes.

This approach is a kind of random local search, being a process similar to that performed by the Simulated Annealing metaheuristic [19], but, unlike it, no worsening solution is accepted. As discussed earlier, combinatorial optimization problems are very sensitive to random or pseudo-random changes, which hardly result in immediate solution improvement without some refinement. In view of this, characteristics of VNS metaheuristic were introduced in the clonal expansion phase. In this method, a solution trapped at a local minimum is disturbed by random movements in one or more neighborhood structures. A local search is then performed, systematically increasing the size of the neighborhood structures exploited. As, in the process of clonal expansion, the mutation itself already causes disturbances in the clones, it is enough, afterwards, to carry out the local search process in the defined neighborhood structures. Thus, at the end of the local search, the solutions are already in local minima and there is no need for several expansions and contractions, therefore, only an expansion is performed. It is worth noting that clones of the same cell can reach completely distinct regions in the search space. In this way, after the expansion, the re-selection process (contraction) is not performed, as in the original algorithm. The set formed by the union of the sets P_c (of population of cloned cells) and P_u (of cells of origin) is forwarded to the process of suppression in all the generations. Note that the cells $c_i \in P_u$ are not submitted to the mutation and local search. The local search performed in COPT-aiNET, after m generations without improvement, is also discarded.

Taking into account the discussed restrictions regarding the algorithm COPT-aiNET and the changes suggested above, an extension of the immune network for combinatorial optimization, called VNS-aiNet, is proposed in the Algorithm 1.

Algorithm 1 Variable Neighborhood Search Immune Network (VNS-aiNET)

```
1:  procedure VNS-AINET
2:      u ← 0;
3:      M_u ← {};
4:      P_u ← Generate nAB cells randomly;
5:      while Stop criterion not satisfied do
6:          Evaluate the fitness of cells c_i ∈ P_u;
7:          Define number of clones nC_i for each c_i ∈ P_u;
8:          P_c ← clone cells c_i ∈ P_u;
9:          Apply mutation to each clone cl_k ∈ P_c;
10:         Apply local search to each clone cl_k ∈ P_c;
11:         P_u ← P_u ∪ P_c;
12:         Update factor mat^i_factor of cells c_i ∈ P_u;
13:         R_u ← Remove cells from P_u where mat^i_factor = 1;
14:         Update the fitness of cells c_i ∈ P_u;
15:         P_{u+1} ← suppress by fitness P_u;
16:         Add nAB − |P_{u+1}| new cells to P_{u+1};
17:         M_{u+1} ← suppress by objective function M_u ∪ R_u;
18:         u ← u + 1;
19:     end while
20:     M_u ← suppress by objective function M_u ∪ P_u;
21: end procedure
```

3 SCHEDULING AND OPERATORS

In order to evaluate the VNS-aiNET algorithm, a job sequencing problem in unrelated parallel machines with sequence-dependent setup-times and goal of minimizing makespan was chosen. This problem lies in one of the sub-areas of combinatorial optimization that presents a higher concentration of works, besides presenting a series of approaches in the literature. The problem is defined as: (i) the assignment of a set $N = \{1, \ldots, n\}$ with n independent jobs to a set $M = \{1, \ldots, m\}$ of m continuously available unrelated parallel machines; (ii) the sequence processing of jobs on each machine; and (iii) the completion time of the processing of each job, in order to minimize the maximum completion time of all jobs, named makespan (C_{max}). Each job j involves a single operation to be performed on a single machine i, requiring a processing period that can not be interrupted, once it has been started, and whose duration p_{ij} depends on the machine i chosen to process it (situation that characterizes the machines as unrelated). The transition between two jobs j and k, adjacent in the job sequence assigned to the machine i, imposes a setup-time s_{ijk}, which depends on the processing sequence of the jobs, as well as the machine i where the two jobs are processed.

3.1 Solution representation and memory initialization

A cell (solution to the scheduling problem) is represented as a vector of pointers with m positions. The index of each position states the identifier of a machine. Each position of the machine vector points to a linked list, each position represents a job assigned to the machine and the order of the list shows the processing sequence on the machine.

The initial population, called P_0, is constructed through the random generation of nAB solutions. The generator must use a uniform probability distribution. An initial immunological memory M_0 empty is also defined.

3.2 Antigen cell affinity

In optimization problems, there is no antigen to be recognized, but instead an objective function to be optimized. Thus, the cell-antigen affinity is usually interpreted as a fitness function exclusively linked to the objective function of the problem itself. As discussed in Section 2, this approach can lead to some difficulties, that may result in premature convergence of the algorithm. Considering these restrictions, the following fitness function is proposed in this work:

$$f_{Ag}(c_i) = (1 - \alpha) f_{like}(c_i) + \alpha f_{mat}(c_i) \quad (1)$$

where $f_{like}(c_i) \in [0,1]$ is a fitness-like function, shown in Equation (2). This first term considers the quality of the solution in relation to the other cells of the population. The term $f_{mat}(c_i) \in [0,1]$ is a function inversely proportional to the number of generations in which the solution c_i did not show any improvement, being able to be trapped in a region that is not very promising of the search space or having found a local minimum. This term reduces the affinity, discouraging the exploration of the solution by the other operators, as we will see in the next sections; α is an algorithm parameter that regulates how priority $f_{mat}(c_i)$ should be about $f_{like}(c_i)$. The fitness-like function $f_{like}(c_i)$ is given by:

$$f_{like}(c_i) = 1 - \frac{f(c_i) - \min_{j \in P_u}(f(c_j))}{\max_{j \in P_u}(f(c_j)) - \min_{j \in P_u}(f(c_j))} \quad (2)$$

where $f(c_i)$ is the value of the objective function to be minimized, given, for the specific problem studied, by the makespan of solution c_i; $\min_{j \in P_u}(f(c_j))$ and $\max_{j \in P_u}(f(c_j))$ are, respectively, the minimum and maximum values of the makespan present in the population P_u. The function $f_{mat}(c_i)$ is given by:

$$f_{mat}(c_i) = 1 - mat^i_{factor} \quad (3)$$

For each cell $c_i \in P_u$, a maturation factor mat^i_{factor} is created. This factor starts with a value equal to 0 and is incremented by a mat_+ index to each generation without improvement, this index being a parameter of the algorithm. If the maturation factor reaches the maximum value ($mat^i_{factor} = 1.0$), the cell is removed from the population. It should be noted that both the factor increment as the cell removal procedure of the population are discussed in Section 3.6.

3.3 Cloning

As discussed in Section 2, the cloning process is the first step of clonal expansion, where cells with greater affinity for the antigen are stimulated to multiply. One of the most important issues discussed in this paper is to avoid computational effort in solutions that are in not promising regions of the search space or that are already in the local minima of their region. Thus, it is proposed that each cell c_i generates a number nCi of distinct clones, directly proportional to the fitness function, and given by:

$$nC_i = \max(1, f_{Ag}(c_i) * nC_{max}) \quad (4)$$

where nC_{max} is a parameter of the algorithm, defined as the maximum number of clones by cell.

To the extent that a solution fails to show improvement, its affinity with the antigen, represented by $fAg(c_i)$, will be reduced, thus discouraging the cloning and exploitation of this solution, even if it has a good evaluation of the objective function.

Each clone generated is an identical copy of the origin cell, including the value of the maturation factor mat^i_{factor}.

3.4 Hypermutation

Hypermutation is the process in which cell diversification and maturation takes place. To guarantee both features, COPT-aiNET uses the mutation rate inversely proportional to the value of the fitness function. Therefore, cells with low performance are submitted to high levels of mutation (diversification), whereas solutions with a high value of the fitness function are submitted to small mutations to guarantee the maturation of the cells.

Due to the characteristics of the combinatorial problems already discussed in Section 2, it is proposed to modify the hypermutation process by adding features of the VNS metaheuristic. In this metaheuristic, a perturbation is made in the solution and then a local search is carried out, systematically increasing the space used by the search. The disturbance must start as a small noise and, to the extent that a solution can not escape from a local minimum, this noise must be high, in order to lead the solution to another region of the search space. In this sense, the perturbation can be understood as the proposed somatic hypermutation operator for the COPT-aiNET, with the only difference being that the mutation level is directly proportional to the maturation level of the cell, as can be seen in equation:

$$n_{mut} = round(\max(1, e^{-f_{mat}(c_i)})) \quad (5)$$

$$c_i = T_{mut}(c_i, \beta * n_{mut}) \quad (6)$$

where β is a parameter of the algorithm, representing the maximum number of possible mutations; $T_{mut}(c_i, n_{mut})$ is a function performing n_{mut} mutations in the c_i cell; and:

$$round(n) = int(n) + \begin{cases} 1, & \text{random}() \leq n - int(n); \\ 0, & \text{otherwise.} \end{cases}$$

This approach allows a solution trapped in a region or in a local minimum be submitted to a larger change in its structure, allowing the local search process can mature it and send it to a minimum of better quality. The function $T_{mut}(c_i, n_{mut})$ must be specific to the problem. Considering the problem under study, for each unit of T one of the transformations described below is randomly selected and applied to the solution :

- **Insertion**: given the list of machines M, obtain machine i, such that $C_i = C_{max}$, remove from i a randomly chosen job j, insert j at a random position k of machine w, chosen randomly at M, w not necessarily being other than i;
- **Change**: given the list of machines M, obtain machine i, such that $C_i = C_{max}$, randomly select at i a job j, randomly select a machine w and a job k of w. Then, change k with j. Machine i is not necessarily different from w, but $j \neq k$ case $i = w$.

3.5 Local Search

In the local search phase one must systematically explore different neighborhood structures. For this, it is first necessary to define which operations define each structure. In this work, the same 4 structures used in [14] were used, which are described as:

- **Internal Exchange**: two jobs, $j \neq k$, assigned to the same machine i, where $i \mid C_i = C_{max}$, have their positions changed;
- **Internal Insertion**: a job j, assigned to machine i, where $i \mid C_i = C_{max}$, is inserted into a position p of machine i;
- **External Insertion**: a job j, assigned to machine i, where $i \mid C_i = C_{max}$, is shifted to a machine w, $i \neq w$;
- **External Exchange**: The job j, assigned to machine i, where $i \mid C_i = C_{max}$, and a job k assigned to machine w, where $i \neq w$, are exchanges between each other.

For each of the above structures, an exploitation strategy must be defined. The same strategies used in [14] were used. In the

case of the first two structures, a complete descent strategy of the best-improvement type is performed; for the latter two, a reduction strategy, already used in [14], is carried out. It is important to emphasize that these actions are always executed only on the machine that defines the makespan. This is a characteristic of the problem under study, which reduces the search space to be exploited, since only changes in this machine can change the value of the objective function.

As stated earlier, the VNS defines that neighborhood structures should be explored in a systematic way. For this, [21] proposes to use, together with the VNS, the local search heuristic VND (Variable Neighbourhood Descent) [21]. In the present work, the same structure of this heuristic proposed by [14] is used.

3.6 Update of maturation factor and removal of fully matured cells

Both the cloning and mutation actions described in the previous sections as well as the suppression action described in the next section depend directly on the maturation factor mat^i_{factor}. This factor indicates the time period in which a given solution c_i does not show improvement in the objective function.

After the process of mutation and maturation of the cloned cells, it is expected that they have escaped from local minima in which the genitor cells are there. Thus, all cloned solutions have their objective function evaluated and, if they show improvement over their genitor cell, their maturation factor mat^i_{factor} is canceled; otherwise, it is incremented by mat_+ $in[0, 1]$, this being a parameter of the algorithm. All genitor cells have the factor mat^i_{factor} decremented by mat_+, since they do not change from previous generation $u - 1$.

When a cell reaches the condition $mat^i_{factor} \geq 1$, it is considered to have reached the maximum maturation level. Thus, it must be removed from the active population P_u, since it has reached its maximum contribution to the immune response. If it is not desired to maintain the multimodality characteristic of AIS, it is enough to verify if this cell is the best known solution; if not, it is then discarded. If it is desired to maintain multimodality, a memory set M_u must be created such that $M_u \neq P_u$. Hence, solutions that reach the maturation threshold in generation u are drawn from P_u and inserted into R_u; then the set $S_u = M_u \cup R_u$ is subjected to the suppression process, forming M_{u+1}. This process does not consider the fitness function proposed here, but rather the value of the objective function as it is in the opt-aiNET algorithm. At the end of all generations, one must apply to the set $S_u = M_u \cup P_u$ the same rule of suppression.

3.7 Suppression

After the clonal expansion process, the immune response is stable. At this point, an interaction occurs between the cells of population P_u, in order to regulate the amount of cells in the population. By the immune network theory, cells with high degree of affinity between each other identify as pathogens, marking each other for the removal by the immune system. The similarity between the structure of two cells defines the affinity between them. It is observed, therefore, that solutions in the same subspace of search compete among themselves for the permanence in the population.

Algorithm 2 Population suppression model

```
1: procedure SUPPRESSION(P_u, σ_s, nAB)
2:     i ← 0;
3:     P_{u+1} ← {};
4:     Insert Best c_i ∈ P_u in P_{u+1};
5:     Sort P_u by function f_Ag(c_i);
6:     while |P_{u+1}| < nAB and i < |P_u| do
7:         c_i ← P_u[i];
8:         if Af_cell(c_i, P_{u+1}) < (1 − σ_s) then
9:             P_{u+1} ← P_{u+1} ∪ c_i;
10:        end if
11:        i ← i + 1;
12:    end while
13:    return P_{u+1};
14: end procedure
```

In all approaches of the opt-aiNET family, when two cells are in the same region of the search space, the cell with the highest value of the objective function is selected to remain in the population. As discussed in Section 2, this approach in populations of restricted size can lead to the transfer to the next generation of stagnant solutions at local minima, wasting computational resources and making it impossible a larger coverage of the search space. In order to overcome this issue, in this paper is proposed that, in the competition between cells belonging to the same subspace, limited by the radius σ_s, the cell with the highest value of the fitness function is selected, in the form as it is presented in equation 1, in which solutions trapped in local minima have their value penalized. The method thus attempts to ensure a continuous immune response, prioritizing solutions of good quality, but not trapped. This approach directly impairs the multimodality capability of the algorithm, since the best solution found by the method can be lost. Thus, to get around this, the best solution must always be inserted in the $u + 1$ generation population.

The pseudo-code presented in Algorithm 2 defines the suppression operator. The affinity function $Af_{cell}(c_i, P_{u+1})$ among cell c_i and the cells chosen to remain in next generation is given by:

$$Af_{cell}(c_i, P_{u+1}) \quad = \quad 1 - \min_{\forall s_j \in P_{u+1}} d(c_i, s_j) \tag{7}$$

where $d(c_i, s_j) \in [0, 1]$ is the distance function between cells c_i and s_j, which must be defined according to the structure of the problem under study. In this work we use the distance function proposed in [14] and given by:

$$d(c_1, c_2) \quad = \quad \frac{1}{N} \sum_{k=1}^{N} \varphi(k, c_1, c_2) \tag{8}$$

where:

$$\varphi(k, c_1, c_2) = \begin{cases} 0, & \text{If the job } k \text{ is immediately preceded in } c_1 \text{ and} \\ & c_2 \text{ by the same job } j \text{ and both are assigned} \\ & \text{to the same machine } i; \\ 1, & \text{otherwise.} \end{cases}$$

Since the suppression can generate a population of size less than nAB, $nAB - |P_{u+1}|$ new cells are generated randomly at the end of the suppression phase, which are inserted into the active population P_{u+1} and contribute to the diversification of the solutions in the population.

4 RESULTS

The experiments performed in this paper are conducted using a set of 1350 instances, presented in [1] and available in [22]. This set of instances are divided into 3 subsets. In the first subset, called *Proc_Dominant*, the processing times are uniformly distributed in the interval [125, 175], while the setup times follow the distribution [50, 100], i.e. the processing times are always greater than the setup times. In the second subset, the distributions are inverted, with the setup times being greater than the processing times. This subset is named *Prep_Dominant*. In the third subset (*Balanced*) there is a balance between processing times and setup times, both distributed in the interval [50, 100]. Each subset has 450 instances, configured with $n = \{20, 40, 60, 80, 100, 120\}$ and $m = \{4, 6, 8, 10, 12\}$, forming 30 groups of different dimensions ($n \times m$), with 15 instances in each group.

In order to verify the performance of the VNS-aiNET, it is proposed to compare this new algorithm to the COPT-aiNET [23] and its variation COB-aiNET[C] [4]. Both were implemented following all the available guidelines, respectively, in [13] and [4]. As the goal is to compare the overall performance of metaheuristics, the local search operator proposed by [14] is used in both algorithms. In addition, the VNS-aiNET algorithm is compared with the Clonal-SL [14], ACO [2] and HABC [20] metaheuristics, which are presented as references for the set of instances evaluated here. The results of these algorithms were made available by the authors.

All the implemented algorithms in the present work were developed in Java language, with JDK 1.8. The experiments were run in an environment with Intel i5-2450M processor with 2.5 GHz, 4GB of RAM and Linux environment Ubuntu 16.04. In [12] it is shown that the number of distinct solutions in unrelated parallel machines problems is given by $(n + m - 1)!/(m - 1)!$. It can be observed that the dimension of the search space is directly proportional to n and inversely proportional to m. Thus, the stopping criterion was established in terms of the execution time in seconds and is defined by the expression:

$$Time_{instance} = \frac{n}{m} \quad (9)$$

Each of the algorithms studied in this work was applied 5 times to each instance. For each result obtained, the relative percentage deviation (RPD) between the solution's makespan found by a given algorithm ($method_{sol}$) and the lower bound ($lower_{instance}$) defined by [1] is calculated for each instance of the benchmark set. The RDP calculation is done in the form:

$$RPD_{instance} = \frac{method_{sol} - lower_{instance}}{lower_{instance}} \quad (10)$$

Table 3 shows the means of the relative percentage deviations for each subset of instances of the same dimensions, that is, instances with the same values of m and n.

In order to verify if there is statistical difference between the general performance of the algorithms in relation to the convergence, a one-way ANOVA is performed, this being 6 levels, which are the evaluated algorithms. An experimental design with blocks was used, so that the groups m vs. n are blocked, since they can influence the overall average. The value $RDP_{average}$ of the group m vs. n presented in Table 3 is considered as the value of the block. The residuals assumptions of normality, independence and homoscedasticity are verified. A confidence level of 95% ($\alpha = 0.05$) is used.

Table 1: Parameters for VNS-aiNET algorithm

Parameter	
Cell numbers in population (nAB)	10
Weight of f_{mat} in fitness function (α)	0.468
Max. number of clones per cells (nC_{max})	6
Maturity factor (mat_+)	0.035
Mutation parameter (β)	2
Suppression threshold (σ_s)	0.072

Table 2: Parameters for COPT-aiNET and COB-aiNET

Parameter	aiNET	
	COPT	COB
Initial number of cells (nAB)	15	25
Max. number of cells ($maxAB$)	50	40
Max. number of clones per cells (nC_{max})	6	9
Number of new antibodies (n_{cels})	6	–
Initial concentration (C_0)	–	0.62
Number of iterations between consecutive local searches (ls_{freq})	7	3
Mutation parameter (β)	1	–
Initial mutation parameter (β_i)	–	0.49
Final mutation parameter (β_f)	–	0.08
Suppression threshold (σ_s)	0.149	0.10

4.1 Parameters

The definition of the parameters of a metaheuristic is one of the factors of greater influence on the quality of the results. In order to avoid that the parameters adversely affect on the results obtained by the algorithms implemented in this work, both the VNS-aiNET algorithm as the COPT-aiNET and COB-aiNET algorithms have their parameters defined by the iRace framework [16]. Each algorithm used 10,000 simulations, and each simulation consumed the maximum time defined by Equation (9). The parameters obtained by the iRace framework for each algorithm are presented in the tables 1 and 2.

4.2 Results for convergence

When analyzing Table 3, it is clear that the VNS-aiNET algorithm presents an average superior performance in relation to the COPT-aiNET and COB-aiNET algorithms. Even for low-dimensional instance groups ($n = 20$), VNS-aiNET shows better performance. It can also be observed that, as the dimensionality of the problem is increased, the performance of the VNS-aiNET algorithm is also improved against the performance of the COPT-aiNET and COB-aiNET algorithms. Although the COB-aiNET presents a slight improvement over COPT-aiNET, both present a quite inferior performance, in all the instance groups, to the reference approaches in the literature. Comparing the VNS-aiNET to the reference methods, we can see that it presents an average performance superior to these in all groups of instances. The HABC metaheuristic presents performance equal to the VNS-aiNET only in sets with low dimensionality ($n = 20$).

In spite of the performance presented in Table 3, when looking at Figure 1 it is not possible to state that the apparent better performance achieved by VNS-aiNET is statistically significant. If we analyze the ANOVA, present in Table 4, we can see that at least

Table 3: Average RDP per machine instance group (m) in relation to jobs (n)

m	n	aiNET			ACO	HABC	Clonal
		VNS	COPT	COB			
4	20	**6.2**	7.4	7.1	6.5	**6.2**	6.3
	40	**3.5**	6.2	5.9	4.4	3.8	4.2
	60	**2.6**	5.9	5.6	3.4	3.0	3.2
	80	**2.2**	5.8	5.6	3.0	2.8	2.8
	100	**2.0**	5.7	5.5	2.6	2.5	2.6
	120	**1.9**	5.6	5.5	2.3	2.4	2.4
6	20	**22.1**	22.5	22.2	22.2	**22.1**	22.2
	40	**7.3**	10.1	9.9	8.9	7.6	7.9
	60	**3.6**	7.5	7.1	5.4	4.2	4.2
	80	**4.9**	8.3	7.9	6.1	5.5	5.5
	100	**3.3**	7.3	6.9	4.6	3.9	3.8
	120	**2.7**	7.1	6.8	3.4	3.2	3.1
8	20	**25.2**	26.1	25.8	25.9	**25.2**	25.4
	40	**5.7**	9.4	9.0	7.2	5.9	6.2
	60	**8.5**	11.5	11.2	9.8	8.9	9.0
	80	**3.7**	8.2	7.9	5.5	4.3	4.3
	100	**5.2**	8.9	8.6	7.1	5.9	5.6
	120	**3.1**	8.0	7.7	4.4	3.7	3.6
10	20	**11.0**	12.9	12.6	12.6	**11.0**	11.2
	40	**6.5**	10.5	10.0	9.9	6.8	6.8
	60	**5.0**	9.5	9.1	7.0	5.5	5.5
	80	**4.3**	9.0	8.7	5.8	4.9	4.8
	100	**3.9**	8.8	8.5	5.9	4.5	4.4
	120	**3.6**	8.6	8.3	5.3	4.4	4.1
12	20	**28.9**	31.0	30.6	30.2	**28.9**	29.2
	40	**23.1**	25.2	24.8	24.6	**23.1**	23.2
	60	**5.5**	10.2	9.8	7.8	6.0	5.9
	80	**8.2**	12.0	11.8	10.0	8.8	8.4
	100	**9.5**	12.6	12.4	10.8	10.3	9.7
	120	**4.0**	9.3	8.9	5.5	4.7	4.4
Total		**7.6**	11.0	10.7	8.9	8.0	8.0

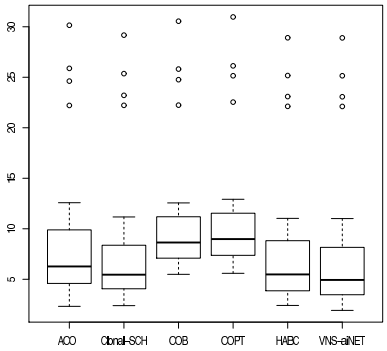

Figure 1: Boxplot for total average RDP of each algorithm.

one of the algorithms presents a mean convergence statistically different from the others ($p - value < 0.05$). When observing Figure 2, which presents a Tukey HSD test, with 95% confidence ($a = 0.05$), it is possible to conclude that the performance achieved by the VNS-aiNET algorithm is statistically significant compared to the achieved by the COPT-aiNET and COB-aiNET algorithms. Comparing the VNS-aiNET with the approaches proposed in the literature, it is noticed that this algorithm clearly presents a superior performance against the ACO. When compared to Clonal-SL and to HABC, the VNS-aiNET presents a statistically superior performance to both, although the margin of error is very close to the limit.

Table 4: ANOVA with $\alpha = 0.05$ for total average RDP

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Algorithm	5	335	66.98	170.3	<2e-16
MxN	29	8368	288.56	733.6	<2e-16
Residuals	145	57	0.39		

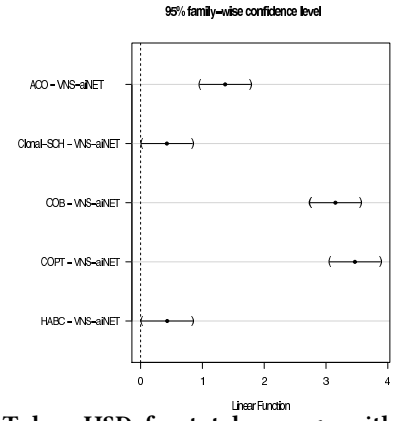

Figure 2: Tukey HSD for total average with confidence level 95%.

4.3 Results: Diversity of Solutions

Multimodality is an important feature of algorithms based on the immune network theory. As discussed in Section 2, the proposed fitness function for VNS-aiNET results in changes in the behavior of the algorithm, especially in the suppression process, thus having potential impacts on components that aid to the multimodality of VNS-aiNET. In Section 3.6 an alternative is proposed for the case where it is desired that the multimodality of the method be preserved. To verify the efficiency of this alternative, a boxplot graph is shown in Figure 3, with the mean distance between the 10 best solutions present in memory M_u, resulting from the VNS-aiNET, COPT-aiNET and COB-aiNET methods. It can be clearly seen that the proposed changes to the VNS-aiNET directly led to the diversity of the solutions present in M_u, with an average of 80% similarity between the solutions. On the other hand, COPT-aiNET and COB-aiNET present average similarity between solutions of less than 10%.

Figure 4 presents the average quality of the 10 best solutions in memory, in relation to the best memory solution. It can be noticed that the VNS-aiNET is able to keep in memory solutions of higher quality, in relation to the best solution found by the method itself. This is particularly interesting because, despite of losing in diversity of solutions, the VNS-aiNET maintains a more homogeneous population in relation to the objective function value, averaging a deviation of 0.2% of the best solution found by the method. The COPT-aiNET and COB-aiNET algorithms present mean deviation of 1.2% and 0.8%, respectively.

5 CONCLUSIONS

In this work the immune network for combinatorial optimization COPT-aiNET is evaluated, identifying that its global optimization power is not fully exploited by the way in which the solutions are

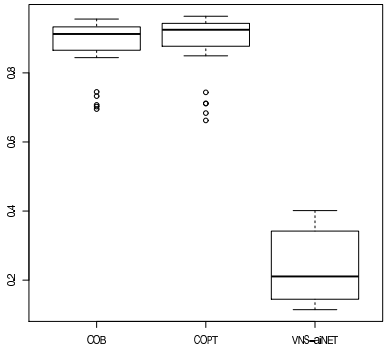

Figure 3: Boxplot showing the diversity among the 10 solutions with the lowest makespan present in memory.

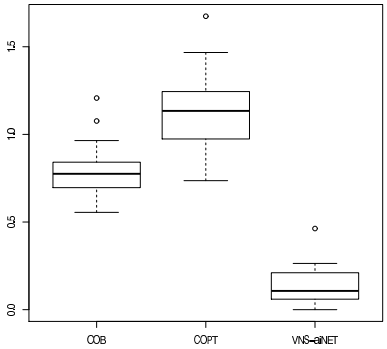

Figure 4: Boxplot for RDP of the average makespan of the 10 best solutions regarding the best solution.

explored and selected for the next generation. To correct this limitation, a new hybrid metaheuristic called VNS-aiNET is proposed, which integrates the VNS trajectory method with the COPT-aiNET algorithm. Alterations in the cell/antigen affinity function are also proposed, allowing new regions of the search space to be prioritized and exploited, when the solutions present in the immunological memory are trapped in local minima of their search subspace. The proposed hybrid metaheuristic is evaluated in a job scheduling problem, which have been a relevant problem in last few years in combinatorial optimization literature. The results indicate that the proposed metaheuristics presented significant performance against the COPT-aiNET and COB-aiNET[C] approaches, both based on the immune network theory for combinatorial problems. The VNS-aiNET algorithm also presented slightly better results than the ACO, HABC and Clonal-SL approaches, algorithms known in the literature as references for the job scheduling problem under study. The experiments carried out indicated that the changes made in VNS-aiNET, for its better performance, add negatively impact in the multimodality of the method, showing lower diversity in the resulting memory solutions when compared to COPT-aiNET and COB-aiNET[C]. However, the solutions in the memory present a very reduced percentage deviation in relation to the best solution of the memory, when compared to the COPT-aiNET and COB-aiNET [C] algorithms.

Despite of the good performance presented, it is necessary that the VNS-aiNET be applied to other combinatorial problems and compared to other metaheuristics, in order to verify its true efficiency and its generalization capacity.

ACKNOWLEDGMENTS

The authors are grateful to CEFET-MG, CNPq and FAPEMIG for the support needed to carry out this work, and to Dr. Ghaith Rabadi for making available the benchmark instances used.

REFERENCES

[1] A. Al-Salem. 2004. Scheduling to minimize makespan on unequal parallel machines with sequence dependent setup times. *Engineering Journal of the University of Qatar* 17, 1 (2004), 177–187.
[2] J.-P. Arnaout, R. Musa, and G. Rabadi. 2014. A two-stage Ant Colony optimization algorithm to minimize the makespan on unrelated parallel machines—part II: enhancements and experimentations. *Journal of Intelligent Manufacturing* 25, 1 (2014), 43–53.
[3] F. M. Burnet. 1957. A modification of Jerne's theory of antibody production using the concept of clonal selection. *Australian Journal of Science* 20 (1957), 67–69.
[4] G. P. Coelho, F. O. de França, and F. J. Von Zuben. 2011. A Concentration-based Artificial Immune Network for combinatorial optimization. In *Proc. of the 2011 IEEE Congress of Evolutionary Computation (CEC)*. 1242–1249.
[5] G. P. Coelho and F. J. Von Zuben. 2010. A concentration-based artificial immune network for continuous optimization. In *Proc. of the 2010 IEEE Congress on Evolutionary Computation (CEC)*. 108–115.
[6] V. Cutello, G. Nicosia, M. Pavone, and J. Timmis. 2007. An Immune Algorithm for Protein Structure Prediction on Lattice Models. *IEEE Transactions on Evolutionary Computation* 11, 1 (Feb 2007), 101–117.
[7] D. Dasgupta. 1998. *Artificial Immune Systems and Their Applications*. Springer.
[8] L. N. de Castro and J. Timmis. 2002. An Artificial Immune Network for Multimodal Function Optimization. In *Proc. of the 2002 Congress on Evolutionary Computation (CEC'02)*, Vol. 1. 699–704.
[9] L. N. de Castro and J. Timmis. 2002. *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer Verlag.
[10] L. N. de Castro and F. J. Von Zuben. 2000. The Clonal Selection Algorithm with Engineering Applications. In *Workshop Proc. of the 2000 GECCO Workshop on Artificial Immune Systems and Their Applications*. Las Vegas, USA, 36–37.
[11] F. O. de França, G. P. Coelho, and F. J. Von Zuben. 2009. On the diversity mechanisms of opt-aiNet: A comparative study with fitness sharing. In *Proc. of the 2009 IEEE Congress on Evolutionary Computation (CEC)*, Vol. 1. 3523–3530.
[12] M. F. de França Filho. 2007. *GRASP e Busca Tabu aplicados a problemas de programaccão de tarefas em máquinas paralelas (in portuguese)*. Ph.D. Dissertation. State University of Campinas, Campinas - SP.
[13] J. S. de Sousa, L. de C. T. Gomes, G. B. Bezerra, L. N. de Castro, and F. J. Von Zuben. 2004. An Immune-Evolutionary Algorithm for Multiple Rearrangements of Gene Expression Data. *Genetic Programming and Evolvable Machines* 5, 2 (2004), 157–179.
[14] R. O. M. Diana, M. F. de França Filho, S. R. de Souza, and J. F. de Almeida Vitor. 2015. An Immune-inspired Algorithm for an Unrelated Parallel Machines' Scheduling Problem with Sequence and Machine Dependent Setup-times for Makespan Minimisation. *Neurocomputing* 163, C (2015), 94–105.
[15] E. Hart, P. Ross, and J. Nelson. 1998. Producing Robust Schedules Via An Artificial Immune System. In *Proc. of the 1998 IEEE International Conference on Evolutionary Computation (ICEC'98)*, Vol. 1. 464–469.
[16] M. L. Ibáñez, J. D. Lacoste, L. P. Cáceres, M. Birattari, and T. Stützle. 2016. The iRACE package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives* 3 (2016), 43–58.
[17] N. K. Jerne. 1974. Towards a network theory of the immune system. *Annales d'immunologie* 125, 1-2 (1974), 373–389.
[18] J. Kelsey and J. Timmis. 2003. Immune Inspired Somatic Contiguous Hypermutation for Function Optimisation. In *Proc. of the 2003 International Conference on Genetic and Evolutionary Computation: PartI (GECCO'03)*. 207–218.
[19] S. Kirkpatrick, D. C. Gellat, and M. P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* 220 (1983), 671–680.
[20] S.-W. Lin and K.-C. Ying. 2014. ABC-based manufacturing scheduling for unrelated parallel machines with machine-dependent and job sequence-dependent setup times. *Computers and Operations Research* 51 (2014), 172–181.
[21] N. Mladenovic and P. Hansen. 1997. Variable Neighborhood Search. *Computers and Operations Research* 24 (1997), 1097–1100.
[22] SchedulingResearch. 2016. Accessed on August 5. (2016). http://SchedulingResearch.com
[23] S. S. F. Souza, R. Romero, and J. F. Franco. 2015. Artificial immune networks Copt-aiNet and Opt-aiNet applied to the reconfiguration problem of radial electrical distribution systems. *Electric Power Systems Research* 119 (2015), 304–312.