Parallel Niche Pareto AlineaGA – an Evolutionary Multiobjective approach on Multiple Sequence Alignment

Fernando José Mateus da Silva[1*], Juan Manuel Sánchez Pérez[2], Juan Antonio Gómez Pulido[2], Miguel A. Vega Rodríguez[2]

[1]School of Technology and Management, Computer Science and Communication Research Centre, Polytechnic Institute of Leiria, Leiria, Portugal, fernando.silva@ipleiria.pt

[2]Dept. Tecnologías Computadores y Comunicaciones, Escuela Politécnica, Universidad de Extremadura, Cáceres, Spain, {sanperez,jangomez,mavega}@unex.es

Summary

Multiple sequence alignment is one of the most recurrent assignments in Bioinformatics. This method allows organizing a set of molecular sequences in order to expose their similarities and their differences. Although exact methods exist for solving this problem, their use is limited by the computing demands which are necessary for exploring such a large and complex search space. Genetic Algorithms are adaptive search methods which perform well in large and complex spaces. Parallel Genetic Algorithms, not only increase the speed up of the search, but also improve its efficiency, presenting results that are better than those provided by the sum of several sequential Genetic Algorithms. Although these methods are often used to optimize a single objective, they can also be used in multidimensional domains, finding all possible tradeoffs among multiple conflicting objectives. Parallel AlineaGA is an Evolutionary Algorithm which uses a Parallel Genetic Algorithm for performing multiple sequence alignment. We now present the Parallel Niche Pareto AlineaGA, a multiobjective version of Parallel AlineaGA. We compare the performance of both versions using eight BAliBASE datasets. We also measure up the quality of the obtained solutions with the ones achieved by T-Coffee and ClustalW2, allowing us to observe that our algorithm reaches for better solutions in the majority of the datasets.

1 Introduction

One of the most common tasks in Bioinformatics is the alignment of molecular sequences. Several biological modelling methods depend on a precise sequence alignment. These methods include phylogenetic reconstruction, profiles and structure prediction; and are applied to different areas such as functional genomics, evolutionary studies, structure modelling, mutagenesis experiments and drug design [30]. Multiple sequence alignment can help compare the structure-relationship among sequences by building connections among their elements [15] and exposing sequences' similarities and differences [22]. Despite the fact that this is a recurring task, there are no methods for computing biologically perfect alignments [20], not only due to the mathematic approach underlying the majority of the existing methods, but also because of the size and complexity of the search space involved.

[*]To whom correspondence should be addressed.

Presently, there are two main methods for performing multiple sequence alignment: the progressive method and the iterative method. However, multiple sequence alignment can also be computed by exact algorithms, which attempt to compute an optimal or suboptimal alignment within well defined boundaries. Even so, this procedure is greatly limited by the intensive computer resources that are necessary for aligning a large number of sequences [18]. Although the progressive approach is the one adopted by the majority of today's multiple sequence alignment tools [20], it has some limitations. In this method the alignment is built by progressively aligning pairs of sequences according to their similarity. If the similarity computation is incorrect, it will have consequences on the final result, which will be aggravated gradually as more sequences are added to the alignment [21]. One of the most visible examples based on this method is ClustalW 2.0 [16], which is probably the most commonly used multiple sequence alignment program [7]. Another example is T-Coffee [21], one of the best algorithms available for this purpose [17]. In opposition, iterative methods produce an alignment and then refine it during a number of iterations until no further improvement can be made [8]. They seek to optimize a scoring function that reflects biological events such that optimizing the score leads to a correct alignment [17]. Evolutionary Algorithms (EAs), such as Genetic Algorithms (GAs) and Parallel GAs (PGAs) are examples of iterative methods.

Despite GAs and PGAs are often used to solve single objective problems, they can also be used in multiobjective domains, where they search for the Pareto front [23] - a set of solutions which presents the best possible optimization of the objectives in test. Niching methods, can be used for distributing the solutions along the Pareto front, increasing the diversity of the population and allowing parallel convergence into different good solutions [23]. Parallel AlineaGA [27] is a parallel EA for solving the multiple sequence alignment problem that makes use of the MPI.NET [12] implementation of the Message Passing Interface (MPI) protocol [13] for communication. The algorithm uses a PGA with a simple local search strategy embedded on some of its mutation operators for optimizing two objective functions.

In this work, we test a multidimensional strategy that makes use of a niching mechanism named Equivalence Class Sharing (ECS) [14] for optimizing both the sum-of-pairs and the identity of the alignments. We test this approach using eight BAliBASE [29] datasets, and we compare the quality of the found solutions with the ones found by its single objective version. We also compare these results with the ones produced by ClustalW2 [16] and T-Coffee [21].

This paper is organized as follows. In the next Section we introduce concepts underlying our research. In Section 3, we present a brief explanation regarding the multiobjective PNPAlineaGA methods. Section 4 presents the experiments performed in order to validate and observe our results. Finally, the ending Section presents final considerations on our work.

2 Background

This Section introduces background and terminology information on the most relevant concepts related with our technical solution for solving the multiple sequence alignment task.

2.1 Alignment

An alignment disposes the sequences in such a way that displays where the sequences are similar, and where they differ. Spaces can be introduced in the sequences for finding regions of similarity among them. These spaces are often referred to as gaps, represented by the ”-” sign. The introduction of gaps into the sequences, allows the alignment to be extended into regions where one sequence may have lost or gained sequence characters not found in the other. An optimal alignment is the one that exhibits the most correspondences and the fewest differences, but which may or may not be biologically meaningful [22]. Figure 1 shows an example of a multiple sequence alignment.

```
KV--NDEV-GEAL
KV--NEEV-GEAL
KVGA-AGEGAEAL
KVGGHGEYGAEAL
```

Figure 1: Example of a Multiple Sequence Alignment.

2.2 Island Parallel Genetic Algorithms

EAs are search methods that take their inspiration from natural selection and survival of the fittest in the biological world. They differ from more traditional optimization techniques in that they engage a search from a population of solutions, not from a single point. GAs are a category of EAs which focus on optimizing general combinatorial problems [11]. They are adaptive and robust search processes whose search methods are inspired in natural events such as genetic inheritance and Darwinian strife for survival [19]. In GAs, a fixed size population of solutions is evolved throughout a series of iterations by means of genetic operators such as crossover and mutation [6]. Crossover merges features in the population by swapping corresponding segments of two randomly chosen individuals (parents), and forming two new similar ones (offspring). Mutation usually applies a random change within a solution. Both operators are used according to a defined probability rate [2]. In order to carry this evolution, every solution has an associated fitness score which reflects its performance and that can be used for determining which solutions can be used for producing new individuals.

Although parallelization is often related to speed up the optimization processes, in GAs it can also result in an effectiveness improvement as consequence of its structured population and parallel execution [1]. Multiple population GAs, make use of various independent subpopulations that exchange solutions from time to time. This trade of solutions is called migration and it is restricted by several parameters, such as the migration rate, that is the number of individuals that are exchanged within the subpopulations; the migration interval, that states when the migration occurs; and the topology, which establishes the connection among subpopulations [3]. This class of algorithms, often referred as Island PGAs (IPGA) [20, 22], are easy to implement, since they run several independent GAs simultaneously, with an extension of subroutines that handles migration. Also, multi-processor computers are becoming widely available, and even single processor machines can simulate their behavior by using technologies as MPI [13].

2.3 Multiobjecive Optimization and Equivalence Class Sharing

Although a great majority of optimization problems are solved as a single objective problem, where the algorithm tries to optimize a single objective function, in reality there are usually more than one conflicting objective to optimize. Multiobjective optimization has been adopted by several different areas of investigation, representing one actual and important area in science and engineering [28]. Unlike single objective optimization, which results in an optimal single solution, multiobjective problems have a set of solutions which represent the best trade-off among the multiple conflicting objectives [14]. These are Pareto optimal solutions and lie on the Pareto front. There are two properties to have in mind when optimizing multiple objectives: the convergence to the Pareto front, ensuring the generation of near-optimal solutions; and uniform diversity, indicating a good distribution of the solutions along the Pareto front [28].

Fitness sharing [9] is a mechanism for maintaining population diversity. It distributes the population over different peaks in the search space by reducing the fitness of highly similar solutions, preventing the convergence to a single point in the Pareto front and allowing parallel convergence into multiple good solutions [23]. ECS [14] is a particular method of sharing which can be applied to the selection stage of the GA, avoiding genetic drift. This technique assumes that candidate solutions, mutually dominated or non-dominated, are equally fit, and computes the niche count of the candidates, selecting the one with the smallest individual number on its neighborhood, maintaining diversity along the front. The niche count, m_i, is computed by adding a sharing function, $Sh(d_{i,j})$, over all members of the population, according to (1).

$$m_i = \sum_{j=1}^{n} Sh(d_{i,j}) \tag{1}$$

Here, $d_{i,j}$ is the distance between solutions *i* and *j*, which can be based on either phenotype or genotype similarity. The sharing function is given by (2).

$$Sh(d_{i,j}) = \begin{cases} 1 - \frac{d_{i,j}}{\sigma_{share}}, & \text{if } d \leq \sigma_{share} \\ \\ 0, & \text{if } d > \sigma_{share} \end{cases} \tag{2}$$

σ_{share} represents the niche radius. Solutions within this radius are in the same neighborhood, and therefore increase the niche count.

3 Methods

Parallel Niched Pareto AlineaGA (PNPAlineaGA) uses an IPGA which relies on a star topology for connecting the different processes. It has been written in C♯, using the .NET Framework 4 and MPI.NET [12] for interconnecting the different processes.

3.1 Representation, Evaluation, Crossover and Mutation

PNPAlineaGA uses real multiple sequence alignments to represent the solutions. Every solution of each subpopulation is randomly generated by putting each sequence in an array line and randomly inserting gaps on its sequences so as their size becomes the same. Then, they are combined and mutated inside each subpopulation during a number of generations, trading solutions periodically. All solutions are evaluated for determining their fitness in each generation.

Two different objectives are evaluated: the sum-of-pairs (3), which adds the scores of all the pairwise comparisons on each column of the alignment [21]; and the identity, i.e. the number of fully identical columns in the alignment.

$$Sum-of-Pairs = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} ScoringMatrix(l_i, l_j) \tag{3}$$

The pairwise alignment cost of every amino acid is determined by the PAM 350 [5] scoring matrix. As well, a gap penalty of -10 [25] is applied when an amino acid is aligned with a gap.

To combine the solutions three crossover operators are used. Two of them derive from Goldberg's standard one point crossover [10], one defines the cut point at a random line and the other at a random column of the alignment. The third one is derived from RecombineMatchedCol [4], and originates a fully identical column from one of the parents, in the other parent for generating the offspring. These operators are randomly selected within each generation. All of the crossover operators are better described in [25].

Mutations can occur, introducing new characteristics in the population and thus, increasing diversity. In order to not modify the sequences and consequently invalidating the alignment, changes can only be made to the gaps and not to the amino acids. Therefore, inserting or shifting gaps are the most effective ways of introducing new patterns in the population. PNPAlineGA uses six mutation operators. Each mutation operator is randomly selected and it is applied to an individual according to the defined mutation probability. If the mutated solution is worst than the original one, a new mutation can be applied to the mutated individual. This can be repeated until the fitness improves or during a certain number of attempts previously defined by the user. In our tests, we perform a maximum of 2 tries because it represents a good tradeoff between speed and robustness, without transforming completely the individuals in a single generation. These operators are divided into stochastic and greedy ones which insert, shift and merge gaps in the sequences of the alignments. The greedy ones, named "Smart" operators [25], embed a simple local search mechanism and only produce mutations whenever an improvement will be made to the original solution. These operators make use of a direction probability which determines the insertion/shift of the gaps at the beginning or at the end of the alignment. This probability is initially set to the center of the alignment and it is updated according to the results, increasing the probability of inserting/moving the gaps to the left or right in the alignment. Also, each operator tries to improve the alignment to the maximum of 3 tries. The "Smart" operators and their effects are deeply explained in [24].

3.2 Selection

The Niched Pareto GA selection scheme uses Pareto Domination Tournaments (PDT) and ECS [14] for sustaining multiple Pareto optimal solutions. Unlike to the normal binary tournaments, PDTs allow the control of the domination pressure by means of a sampling scheme. Here, two random candidate solutions are picked from the population for selection. Also, a random set of population individuals is chosen to form a comparison set which is used to evaluate the candidate solutions. The candidate who dominates all solutions in the comparison set wins the tournament. Domination pressure can be controlled by adjusting the size of the comparison set. In general, a comparison set of 10% of the population size leads to a tight and complete distribution over the front [14]. Whenever both candidates are dominated or non-dominated by the sample set, ECS is used. This method selects the candidate which presents the smallest number of solutions in its neighborhood. We use the Euclidean distance for computing $d_{i,j}$. The niche radius σ_{share} is calculated according to (4), which assumes that the solution set has a previously known finite number of peaks q [23]. However, as in this domain there is no way of knowing this number beforehand, we opted for considering 4 peaks because it was the value which performed better for the majority of the datasets in our previous experiments [26].

$$\sigma_{share} = \frac{r}{\sqrt[n]{q}} \tag{4}$$

In (5), r is defined by computing the lower and upper bounds of each dimension n on every generation.

$$r = \frac{1}{2}\sqrt{\sum_{k=1}^{n}(x_{k,max} - x_{k,min})^2} \tag{5}$$

3.3 Migration

PNPAlineaGA uses a star topology for connecting the different processes. In this model, every subpopulation is assigned to a different process where evolves independently. On every migration interval, each slave process sends its best solutions to the master process and the master process sends its best individuals to each slave process. This topology has the advantage of allowing the increase or decrease of the number of processes without changing the algorithm's implementation. It is also more fault tolerant, because if one of the slave processes stops, it will not stop the program from running.

Three solutions are traded among the different islands: the one which presents the best compromise between the two objectives; the best solution for the sum-of-pairs objective; and the best solution for the identity objective. These migrants replace each subpopulation's worst solutions, holding their size unaltered. In order to prevent bottleneck issues caused by slower processes, this is performed asynchronously. The communication topology relies on MPI.NET [12].

4 Testing and Results

To obtain the best possible solutions that optimizes both sum-of-pairs and identity of the alignments we have tested PNPAlineaGA with eight BAliBASE [29] datasets. BAliBASE is a database of manually refined multiple sequence alignments. It was developed in order to establish a high quality structured and classified benchmark database for evaluating multiple sequence alignment programs [29]. Four of the datasets (1aho, 1fmb, 1plc and 1dox) present more than 35% of identity among its sequences and the remaining ones (1fjlA, 1hpi, 1pfc and 1ycc) present 20% to 40% of identity. The parameters choice for our tests are based on our previous work [27]; therefore, subpopulation size is 100, the comparison set for the PDTs is

Table 1: Comparing PNPAlineaGA with Parallel AlineaGA, ClustalW2 and T-Coffee results. SOP, sum-of-pairs; ID, identity.

Dataset	BAliBASE		ClustalW2		T-Coffee		Islands	PNPAlineaGA Average Best		Parallel AlineaGA Average Best	
	SOP	ID	SOP	ID	SOP	ID		SOP	ID	SOP	ID
1aho	2015	12	1644	8	2010	11	4	2116.03	11.97	2145.67	10.50
							8	2195.80	12.13	2207.77	11.03
1fmb	1706	25	1780	24	1805	24	4	1835.70	26.30	1859.80	25.40
							8	1855.20	26.30	1876.23	25.86
1plc	2403	18	2387	14	2529	18	4	2432.27	18.30	2537.53	17.57
							8	2515.33	18.73	2590.57	17.83
1dox	1234	22	1020	19	1133	21	4	1176.60	22.20	974.40	9.53
							8	1257.63	23.40	1162.03	16.30
1fjlA	1740	6	1770	6	1801	6	4	1531.37	4.80	1558.17	4.50
							8	1712.67	5.87	1710.03	5.43
1hpi	1280	10	1087	10	1148	11	4	1164.70	12.40	1190.37	11.33
							8	1158.17	12.73	1199.43	11.63
1pfc	2216	13	2231	9	2425	12	4	2389.07	14.63	2501.80	12.70
							8	2461.57	14.00	2533.33	12.86
1ycc	963	11	798	5	520	11	4	995.97	8.27	1099.33	6.50
							8	1064.73	8.50	1152.20	7.43

10, crossover and mutation rates are 0.8 and 0.4 respectively, migration occurs every 100 generations, and the algorithm execution is limited to 1000 generations.

4.1 Solution Quality

Table 1 presents the results achieved by PNPAlineaGA and compares them with the ones achieved by single objective Parallel AlineaGA [27], ClustalW version 2.0.12 [16] (available at: www.ebi.ac.uk/Tools/clustalw2) and T-Coffee version 8.93 [21] (www.ebi.ac.uk/Tools/tcoffee), all using the PAM 350 scoring matrix and default gap penalty values. ”BAliBASE” column presents the sum-of-pairs and identity scores for the reference alignments, calculated using the PAM 350 and a -10 gap penalty. The ”ClustalW2” and ”T-Coffee” columns presents both sum-of-pairs and identity of the best found solution. ”Average Best” for both ”PNPAlineaGA” and ”Parallel AlineaGA” presents the average best scores for the sum-of-pairs and identity over 30 independent runs of each program. All these tests were performed for 4 and 8 island configurations.

Globally, the 8 island configuration achieves better scores than the 4 island versions on both versions of Parallel AlineaGA. However, despite we are not focusing on communication costs, the 8 island model has the drawback of a higher execution time. Also, Parallel AlineaGA finds solutions with higher sum-of-pairs scores, which is expected because it does not try to optimize conflicting objectives, such as PNPAlineaGA. Still, there is one exception in the 8 island configuration of dataset 1dox, where the optimization of both objectives allowed PNPAlineaGA to perform better on both objectives. Considering the identity objective, PNPAlineaGA behaves better than Parallel AlineaGA in all datasets.

Comparing to the BAliBASE sum-of-pairs scores, PNPAlineaGA is able to find better values for the majority of the datasets, with the exceptions of 1fjlA and 1hpi. Considering the identity objective, PNPAlineaGA is able to find solutions with higher scores than BAliBASE on 6 of the 8 datasets. PNPAlineaGA behaves better than ClustalW2 and T-Coffee on all of the datasets except 1fjlA, where it is not able to find better scores for both objectives, and 1ycc, where it surpasses only in the sum-of-pairs score.

4.2 Population's Evolution

As explained in Section 3.3, in PNPAlineaGA there are three solutions which are considered for migration. Figures 2 to 17, present the fitness evolution of these three solutions on each subpopulation, for each dataset, on the 4 and 8 island models of PNPAlineaGA in 4 particular moments: generations 100, 400, 700 and 1000. The results presented here were obtained by averaging each solution's sum-of-pairs and identity scores from the 30 runs of the program.

Comparing the evolution of the subpopulations for dataset 1aho on the different island configurations, represented in Figures 2 and 3, it is noticeable that the configuration which uses 8 islands presents a better overall result, which is visible in the four generations in study.

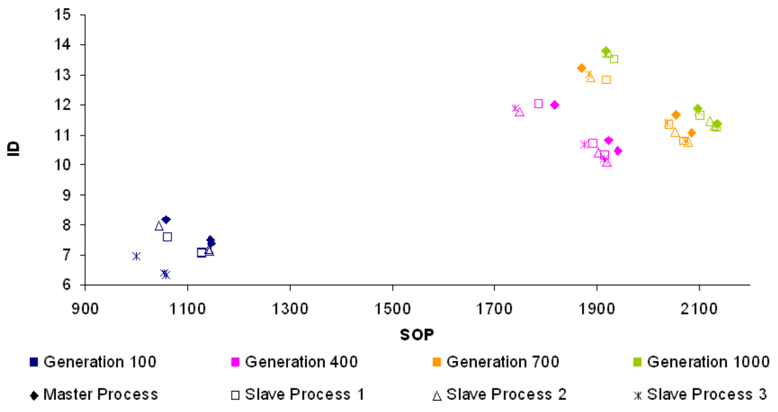

Figure 2: Population's Evolution for 1aho using PNPAlineaGA 4 island model.

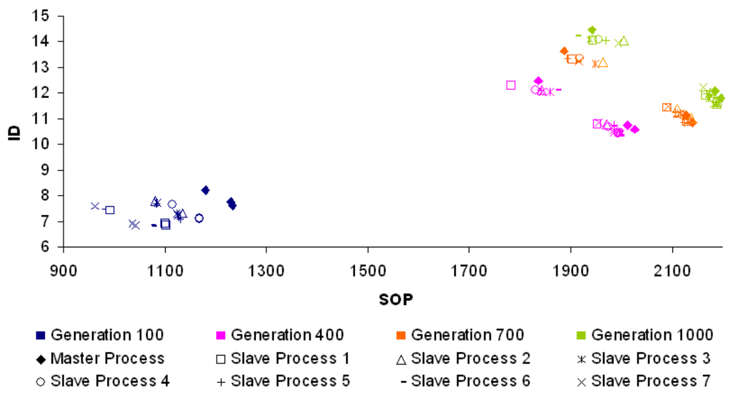

Figure 3: Population's Evolution for 1aho using PNPAlineaGA 8 island model.

Figures 4 and 5 depict the evolution of the best solutions of the different subpopulations for dataset 1fmb, on the 4 and 8 island models, respectively. The advantage of using a larger model is clearer in the early stage of execution of the algorithm. Nevertheless, despite the solutions found by the 4 islands model appear to be similar, the majority of the 8 island model solutions present better throughout evolution, and the final solution set is also better.

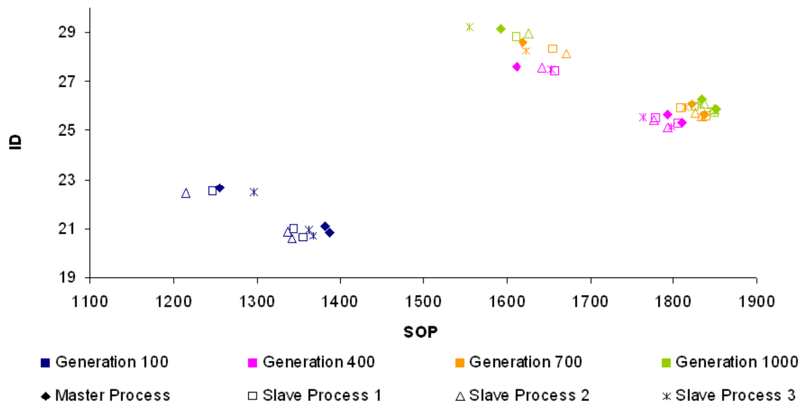

Figure 4: Population's Evolution for 1fmb using PNPAlineaGA 4 island model

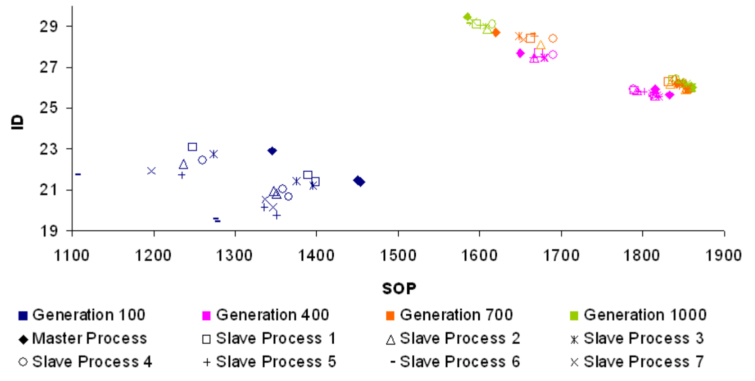

Figure 5: Population's Evolution for 1fmb using PNPAlineaGA 8 island model

Comparing the best solutions evolution on 1plc dataset, in Figures 6 and 7, we can observe that, despite that in an early state of execution, there is not a visible advantage of using a larger parallel model, that use is rewarded in the subsequent generations. The bigger capacity of exploration and exploitation of a larger number of subpopulations, allows the 8 island model to reach better quality solutions, with higher scores on both objectives in test.

Figures 8 and 9 represent the evolution of the best solutions in dataset 1dox. In generation 100, the model with 4 islands presents higher identity and sum-of-pairs scores. However, in the subsequent generations, the 8 island model reaches higher values on both objectives. That can be observed from generation 400. The final solution set also presents a better tradeoff between the identity and the sum-of-pairs, what is translated in better scores and higher quality solutions.

By comparing Figures 10 and 11, it is possible to examine the performance of the 4 and the 8 island parallel evolutionary algorithm on dataset 1fjlA. For this dataset, the 8 island model performed better since generation 100. While in generation 400 the best sum-of-pairs score of the 4 island model is around 1300, the 8 island model presents a score near 1500, which is approximately the maximum sum-of-pairs score of generation 700 in the 4 island model. This better performance reflects on the final solution set, which, for the 8 island model, presents much better scores on both objectives, and therefore, achieves better quality solutions.

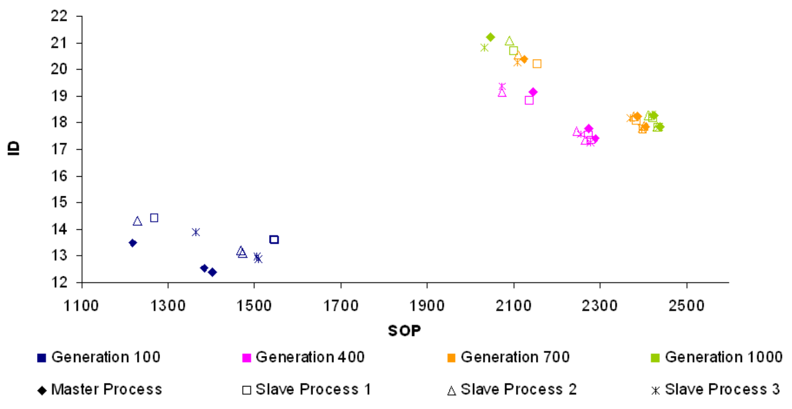

Figure 6: Population's Evolution for 1plc using PNPAlineaGA 4 island model

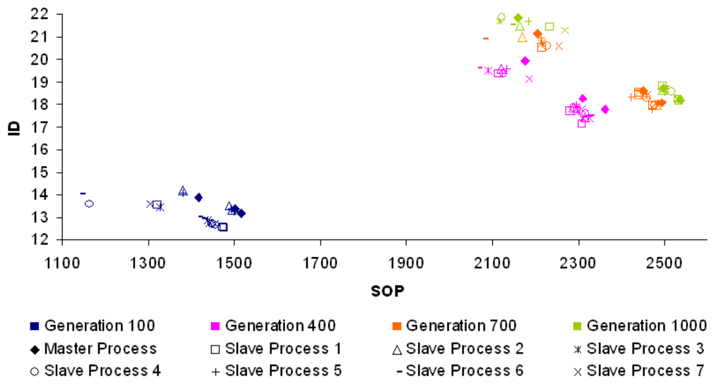

Figure 7: Population's Evolution for 1plc using PNPAlineaGA 8 island model

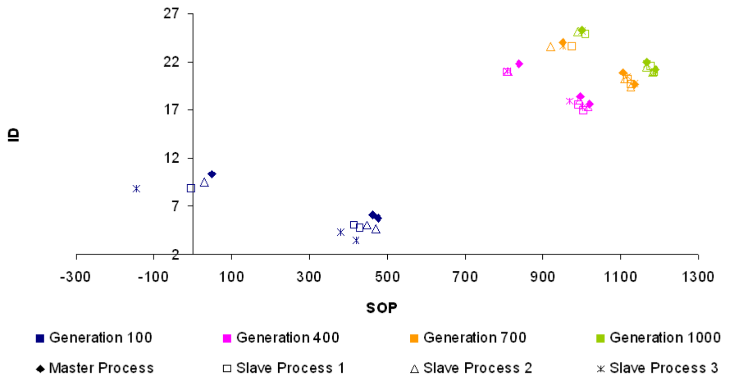

Figure 8: Population's Evolution for 1dox using PNPAlineaGA 4 island model

In dataset 1hpi, the evolution of the best solutions in the 4 island model, shown on Figure 12, results in a final solution set quite similar to the one presented by the 8 island model, depicted on Figure 13. In fact, a closer observation allows determining that in this case, the 4 island model presents a slightly better final sum-of-pairs score. This however, does not happen when considering the identity score, which is higher in the 8 island model.

In dataset 1pfc, the evolution of the best solutions performed better in the 8 island model of PNPAlineaGA, shown in Figure 15, resulting on a better quality final solution set than the one achieved by the 4 island model, in Figure 14. If in generation 100 the superiority of the larger 8 island model is only visible through the sum-of-pairs score, in the remaining generations that

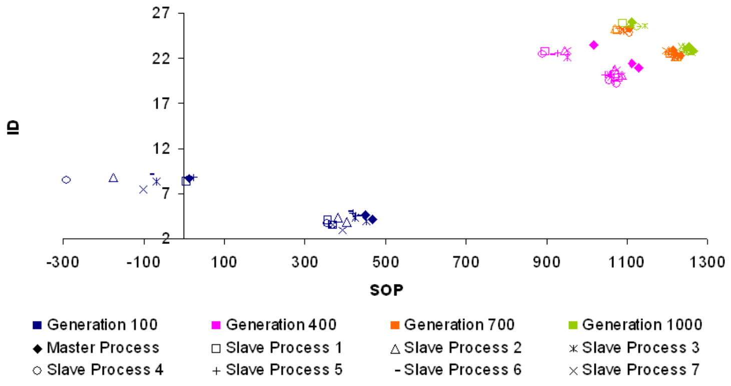

Figure 9: Population's Evolution for 1dox using PNPAlineaGA 8 island model

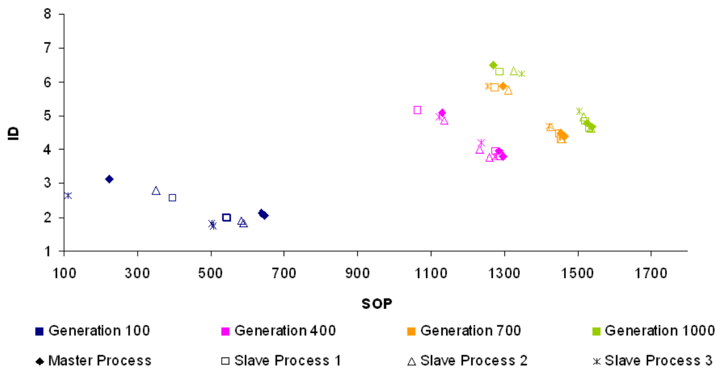

Figure 10: Population's Evolution for 1fjlA using PNPAlineaGA 4 island model

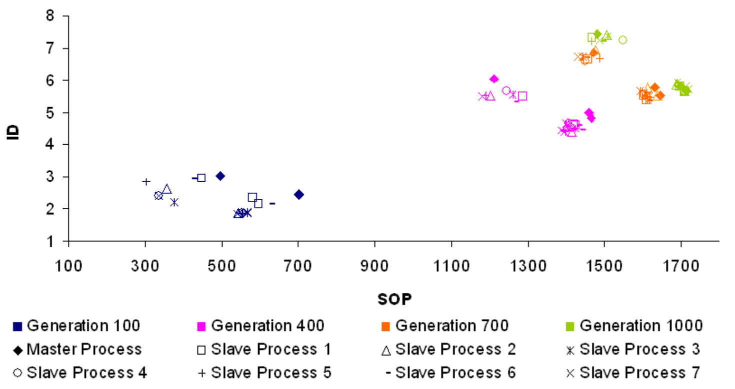

Figure 11: Population's Evolution for 1fjlA using PNPAlineaGA 8 island model

changes. Here it is possible to observe that the 8 island model presents better quality results in all objectives in test. The use of a larger number of subpopulations allows a better exploration of the Pareto front, which is wider in the 8 island model.

In dataset 1ycc, both models present an initial similar evolution such as Figures 16 and 17 depict. However in generation 700 the advantage of using the larger model becomes visible. From this point, the 8 island model presents higher scores on both objectives. The quality of the solutions of the 8 island model in generation 700 is similar to those present in the final solution set found by the 4 island model.

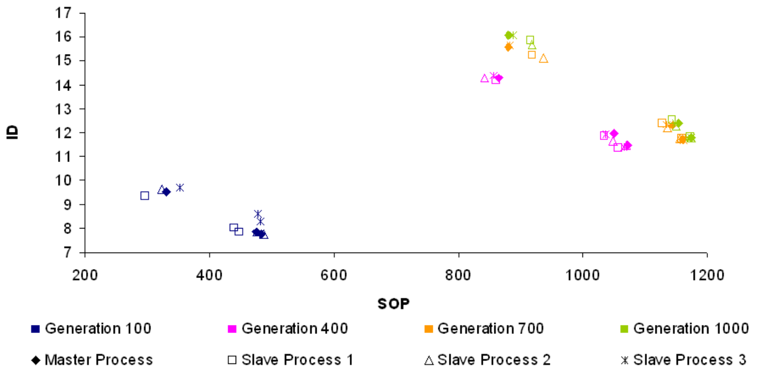

Figure 12: Population's Evolution for 1hpi using PNPAlineaGA 4 island model

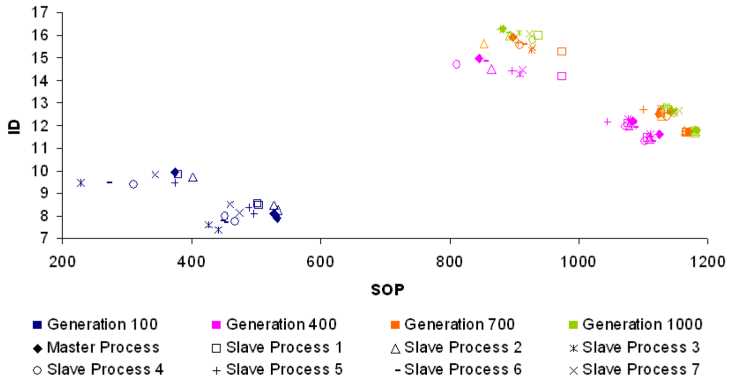

Figure 13: Population's Evolution for 1hpi using PNPAlineaGA 8 island model

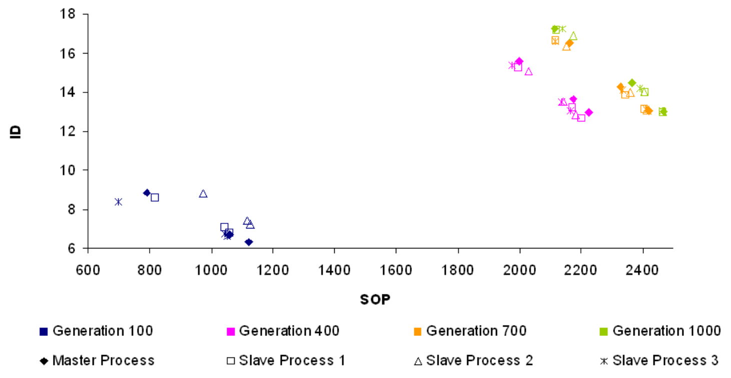

Figure 14: Population's Evolution for 1pfc using PNPAlineaGA 4 island model

5 Conclusions

PNPAlineaGA is able to achieve quality solutions which can be observed by comparing its results with those presented by the test datasets. In the most cases, it can overcome two of the most popular sequence alignment tools: ClustalW [16] and T-Coffee [21]. By optimizing two objectives it is possible to achieve alignments which are similar to those present in the reference datasets. However, this may be of little use when aligning sequences which have distant evolutionary relations, and that consequently, are expected to present low identity scores. In these cases, optimizing only the sum-of-pairs may be the best option. The big advantage of the multiobjective approach becomes evident in cases where the sequences to align present a

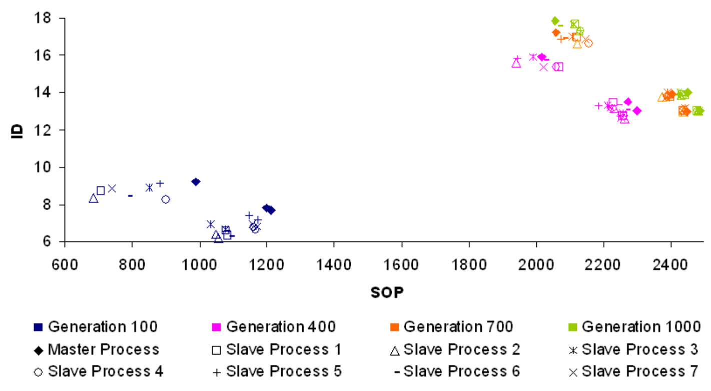

Figure 15: Population's Evolution for 1pfc using PNPAlineaGA 8 island model

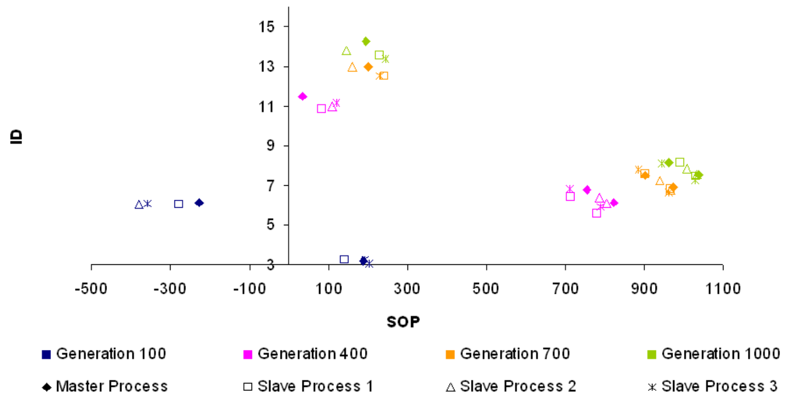

Figure 16: Population's Evolution for 1ycc using PNPAlineaGA 4 island model

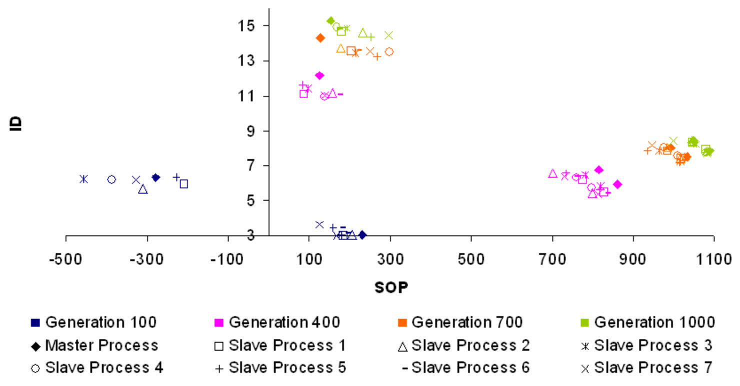

Figure 17: Population's Evolution for 1ycc using PNPAlineaGA 8 island model

close evolutionary relationship. Also, given that the optimal alignment is obtained mathematically and not biologically, the existence of a final set of solutions can be an advantage to help choosing the one closest to the biological reality. In these tests, it has been possible to observe that the use of a larger number of populations, allows a better exploration and exploitation of the search space. This fact contributes to find better solutions in fewer generations. However, although this study does not consider the execution costs, it has been possible to observe that increasing the number of islands leads to greatly increased run times. This disadvantage can be overcome by installing PNPAlineaGA in a computer cluster, where its speed and performance can be optimized.

Acknowledgements

This work has been partially supported by the Polytechnic Institute of Leiria (Portugal) and the MSTAR project Reference: TIN2008-06491-C04-04/TIN (MICINN Spain).

References

[1] E. Alba and J. M. Troya. A survey of parallel distributed genetic algorithms. *Complexity*, 4(4):31–52, 1999.

[2] L. A. Anbarasu, P. Narayanasamy, and V. Sundararajan. Multiple molecular sequence alignment by island parallel genetic algorithm. *Current Science*, 78(7):858–863, 2000.

[3] E. Cantú-Paz. A survey of parallel genetic algorithms. *Calculateurs Paralleles, Reseaux et Systems Repartis*, 10(2):141–171, 1998.

[4] Kumar Chellapilla and Gary B. Fogel. Multiple sequence alignment using evolutionary programming. In Peter J. Angeline, Zbyszek Michalewicz, Marc Schoenauer, Xin Yao, and Ali Zalzala, editors, *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, volume 1, pages 445–452, Washington DC, USA, 1999. IEEE Press.

[5] M. O. Dayhoff, R.M. Schwartz, and B.C. Orcutt. A model of evolutionary change in proteins. In *Atlas of Protein Sequence and Structure*, volume 5, pages 345–352. National Biomedical Research Foundation, 1978.

[6] Kenneth De Jong. Learning with genetic algorithms: An overview. *Mach Learning*, 3(2-3):121–138, 1988.

[7] R. C. Edgar and S. Batzoglou. Multiple sequence alignment. *Current Opinion in Structural Biology*, 16(3):368–373, 2006.

[8] G. Fogel and D. Corne. *Evolutionary computation in bioinformatics*. Morgan Kaufmann Pub, 2003.

[9] D. E. Goldberg and Jon Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms and their Application*, pages 41–49, Cambridge, Massachusetts, United States, 1987. L. Erlbaum Associates Inc.

[10] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, Reading, MA, 1989.

[11] Perry Gray, William Hart, Laura Painton, Cindy Phillips, Mike Trahan, and John Wagner. A survey of global optimization methods, 10/2007 1997.

[12] Douglas Gregor and Andrew Lumsdaine. Design and implementation of a high-performance mpi for c♯ and the common language infrastructure. In *Proceedings of the 13th ACM SIGPLAN Symposium on Principles and practice of parallel programming*, pages 133–142, Salt Lake City, USA, 2008. ACM. 1345228 133-142.

[13] W. Gropp, E. Lusk, and A. Skjellum. Using mpi: portable parallel programming with the message passing interface. 1999.

[14] J. Horn, N. Nafpliotis, and D. E. Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, 1994.

[15] JT Horng, LC Wu, CM Lin, and BH Yang. A genetic algorithm for multiple sequence alignment. *Soft Computing*, 9(6):407–420, 2005.

[16] M. A. Larkin, G. Blackshields, N. P. Brown, R. Chenna, P. A. McGettigan, H. McWilliam, F. Valentin, I. M. Wallace, A. Wilm, and R. Lopez. Clustal w and clustal x version 2.0. *Bioinformatics*, 23(21):2947, 2007.

[17] T. Lassmann and E. L. L. Sonnhammer. Quality assessment of multiple alignment programs. *FEBS Letters*, 529(1):126–130, 2002.

[18] Odile Lecompte, Julie D. Thompson, Frédéric Plewniak, Jean-Claude Thierry, and Olivier Poch. Multiple alignment of complete sequences (macs) in the post-genomic era. *Gene*, (270):17–30, 2001.

[19] Zbigniew Michalewicz. *Genetic algorithms + data structures = evolution programs - Third, Revised and Extended Edition*. Springer, 3 edition, 1996.

[20] Cédric Notredame. Recent evolutions of multiple sequence alignment algorithms. *PLoS Comput. Biol*, 3(8):e123, 2007.

[21] Cédric Notredame, Desmond G. Higgins, and J Heringa. T-coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 302(1):205–17, 2000.

[22] S.K. Pal, S. Bandyopadhyay, and S.S. Ray. Evolutionary computation in bioinformatics: A review. *IEEE Transactions on Systems Man and Cybernetics Part C-Appl and Rev*, 36(5):601–615, 2006.

[23] O. M. Shir and T. Back. Niche radius adaptation in the cma-es niching algorithm. *Lecture Notes in Computer Science*, 4193:142, 2006.

[24] Fernando José Mateus Silva, Juan Manuel Sánchez Pérez, Juan Antonio Gómez Pulido, and Miguel Ángel Vega Rodríguez. Optimizing multiple sequence alignment by improving mutation operators of a genetic algorithm. In *Ninth International Conference on Intelligent Systems Design and Applications*, pages 1257–1262, 2009.

[25] Fernando José Mateus Silva, Juan Manuel Sánchez Pérez, Juan Antonio Gómez Pulido, and Miguel Ángel Vega Rodríguez. An evolutionary approach for performing multiple sequence alignment. In IEEE, editor, *2010 IEEE World Congress on Computational Intelligence*, pages 992–998, Barcelona, Spain, 2010.

[26] Fernando José Mateus Silva, Juan Manuel Sánchez Pérez, Juan Antonio Gómez Pulido, and Miguel Ángel Vega Rodríguez. A niched pareto genetic algorithm for multiple sequence alignment optimization. In Joaquim Filipe, Ana L. N. Fred, and Bernardette Sharp, editors, *International Conference on Agents and Artificial Intelligence*, volume 1 - Artificial Intelligence, pages 323–329, Valencia, Spain, 2010. INSTICC Press.

[27] Fernando José Mateus Silva, Juan Manuel Sánchez Pérez, Juan Antonio Gómez Pulido, and Miguel Ángel Vega Rodríguez. Parallel alineaga: An island parallel evolutionary algorithm for multiple sequence alignment. In IEEE, editor, *2010 International Conference on Soft Computing and Pattern Recognition*, pages 279–284, Cergy Pontoise, Paris, France, 2010.

[28] El-Ghazali Talbi. *Metaheuristics: From Design to Implementation*. Wiley Publishing, 2009.

[29] J. D. Thompson, F. Plewniak, and O. Poch. A comprehensive comparison of multiple sequence alignment programs. *Nucleic Acids Research*, 27(13):2682, 1999.

[30] J. D. Thompson and O. Poch. Multiple sequence alignment as a workbench for molecular systems biology. *Current Bioinformatics*, 1(1):95–104, 2006.