Verifying the Impact of the Tournament Selection on Genetic Algorithms

Yuri Lavinas[†], Claus Aranha[†], Marcelo Ladeira[††]

University of Tsukuba, Japan[†], University of Brasilia, Brazil[††]

The tournament selection holds a popular condition among Genetic Algorithm Selection schemes. In this work we want to observe the impact into the quality of the Genetic Algorithm applied to the BBOB benchmark functions given the variety of one of its parameter, the tournament size. From the literature, we found out that there is a predilection to values for the tournament size, such as 2 or 3. To analyze this impact, we examined the results of different values for the selection operator through simulations the 24 BBOB noise free benchmarks functions with 10, 20 and 40 dimensions and comparing their behavior. The results indicate that for the BBOB benchmark functions the impact of chaging the values of the tournament size have little impact into the quality of the results.

1 Introduction

Genetic Algorithm (GA) is a well-known metaheuristic that is used to seek solutions given a search space. The GA commonly relies on operators such as the crossover operator, the mutation operator and the selection operator. Each of these operators have their own and specific role in the GA.

The selection operator selects solutions to compose the next generation given their measured fitness function quality value, with higher solutions having higher probability to compose the next generation. The crossover operator generates new solutions by recombining the genetic materials of pairs of individuals to generate new offspring. The mutation operator is applied to the population to sustain diversity, by changing the value of a gene. Both crossover and mutation operators are applied to explore new points in the search space [7] .

One of the most used selection operator is the tournament selection operator. It selects individuals based on their fitness value and on a given parameter, named the tournament size. Controlling the tournament size parameter is one way of adjusting the balance between the exploration and exploration [7] , which governs the search process in GA. Although, it is a very important balance, it cannot be set precisely [10] . The selection pressure of tournament selection is expected to increase as the tournament size becomes larger. In other words, the higher the selection pressure, the more the better individuals influences the next generation [20] .

As a rule of thumb, small values, such as 2 or 3, are widely used, with small selection pressure and may be good choices, but,very often, the search process converges too slow with smaller tournament size (and too fast with bigger tournament size). [10] .

We would like to reconsider the idea that exists a relationship between exploration versus exploitation and the tournament size parameter by verifying, experimentally, the real impact of the tournament size on the Genetic Algorithm.

Consequently, our goal in this paper, is to observe the performance of the tournament selection in a real-valued Genetic Algorithm. For that we applied the Genetic Algorithm to 24 noise-free BBOB benchmark functions with 10, 20 and 40 dimensions [14] . We explore the number of individuals to be selected by the Tournament operator, with values from 2 to 25.. Then we analyze the

Verifying the Impact of the Tournament Selection on Genetic Algorithms

[†] Yuri Lavinas(yclavinas@gmail.com)

[††] Claus Aranha(caranha@cs.tsukuba.ac.jp)

[††] Marcelo Ladeira(mladeira@unb.br)

Department of xxx bbb University (†)

Graduated School of yyy bbb University (††)

results to verify any relationship among the tournament size value, the benchmark functions, and the performance. We also explore briefly the idea that the value of the tournament size value had different effects at different points in the algorithm.

From the literature, it is possible to understand that there is a preference to small values for the tournament size, as 2 or 3 [11, 13, 1, 15, 26, 16, 9, 5, 18, 6, 22, 25, 23] . We could trace back to 1991, and we found out that even from that time is has been a popular choice. However, we must highlight that these values are chosen arbitrarily and has small scientific justification. For instance, in 2009, Nicolau used a more pragmatic approach and set an empirical equation to his binary GA.

These things are described here and there. we do this and that here and there too. results and other stuff are here and there.

2 Tournament Selection

The selection scheme is intended to improve the average quality of the GA by giving individuals of higher quality a higher probability of composing the next generation [7] . It is also an important factor to the convergence quality of the GA and the goal of any selection scheme is to favor the proliferation of good individuals in the population of the GA [15] .

Goldberg and Deb [12] stated that there exists several selection schemes commonly used in GA, such as the rank selection, the roulette-wheel Selection and the tournament selection. They also declare that the complexity of the tournament selection is O(n) and say that tournaments are often held with tournament size s = 2, although larger tournament sizes may be analyzed.

2.1 The Tournament Selection Procedure

Tournament selection is translation and scaling invariant [19] , meaning that the behavior of the selection method is not affect by and translation or scaling algorithm [3] .

Tournament selection is conducted as follows: choose some number, k, of individuals randomly from the population and copy the best individual from this group into an intermediate population, and repeat N times, algorithm 1.

Algorithm 1: Tournament Procedure

input : individuals, k, tournsize

- individuals: A list of individuals to select from.
- k: The number of individuals to select.
- tournsize: The number of individuals participating in each tournament.

output: list of selected individuals

```
1 selected = list()
2 for i = 1 to k do
3 |   candidates = choose_Random(individuals,
  |     tournsize)
4 |   selected[i] = max(candidates, by: fitness
  |     value)
5 end
6 return selected
```

Often tournaments are held only between two individuals (binary tournament) but a generalization is possible to an arbitrary group size k called tournament size [7, 12] . A tournament selection, as any selection mechanism in GA, simply favors the selection of better individuals of a population to influence the next generation.

2.2 Selection Schemes and BBOB functions

There are some examples of applications that applied these selection schemes with the BBOB functions. Next we show a few of this examples.

Holtschulte and Moses, in [17] used rank selection in their GA implementation and applied it to the BBOB functions. They declared that rank selection should be replaced by an elitism selection scheme to improve the performance of their GA implementation by spreading of high quality solutions through the population.

One of these elitist selection scheme is the fitness-based roulette-wheel. However, it has been shown to lead to premature convergence [4] . Nicolau, in [22] uses the tournament selection in his GA implementation also aiming to solve the BBOB functions. He declares that "for harder problem domains, such as the ones on the BBOB-Benchmark suite, the tournament selection scheme is more appropriate.".

Therefore, in this we work we chose to analyze

the tournament selection behavior in the BBOB functions. The main reasons for that are because the tournament selection is a widely used selection operator [11, 13, 1, 15, 26, 16, 9, 5, 18, 6, 22, 25, 2, 23] and because, as showed previously, it is a more adequate selection scheme to by the BBOB benchmark functions. Likewise, it is simple to code, efficient in both parallel and non parallel architectures, and its pressure is able to be adjusted for different domains [20] .

2.3 Tournament Pressure

Muhlenbein et al., in [21] , introduced the concept of selection intensity, which is to measure the pressure of selection schemes. The selection intensity is defined as the expected average fitness of the population after selection.

With higher tournament sizes, the selection pressure rises resulting in higher convergence rates, because the convergence rate of a GA is largely determined by the selection pressure. Accordingly, if the selection pressure is too low, the convergence rate will be slower if it is too high, there is an increased chance of the GA prematurely converging [20] .

We understand that it is important to explore the values for tournament size, since this value has direct impact on the quality of the final result of the Genetic Algorithm. This impact is caused because this selector holds a important role on the Genetic Algorithm, since it determines how the balance of exploration and exploitation is going to be corresponded in the Genetic Algorithm, by changing the selection intensity. Also, it is possible that this tournament size value could change given a domain of the Genetic Algorithm that used Tournament Selection, because it is expected that every domain holds a specific characteristic that would need a different proportion of this balance.

2.4 Tournament Size in the Literature

In 1991, Goldberg [11] , analyzed the tournament selection with size 2 in a real-coded GA. They recognized, empirically, that after some time only individuals with relatively high function values will be represented in the population. Also it compares the importance of mutation operators and the tournament selection operator and states that "the addition of some small amount of creeping mutation or other genetic operators should not materially affect these results".

Agrawal [1] , in 1995, implemented real-coded GA with tournament selection with size 2. Their goal was to compare the results of the simulated binary crossover (SBX) with the binary single-point crossover and with the blend crossover. They concluded that the SBX has search power similar to the single-point crossover and the SBX performs better in difficult test functions than the blend crossover.

In 1999, Tsutsui et al. [26] proposed a simplex crossover (SPX), a multi-parent recombination operator for real-coded genetic algorithms. For their experiments, they used a real-coded GA with tournament selection with size 2 and showed that "SPX works well on functions having multi modality and/or epistasis with a medium number of parents: 3-parent on a low dimensional function or 4 parents on high dimensional functions.".

Later, in 2000, Deb [9] , implemented a binary GA and also a real-coded GA, both with tournament size 2. He aimed to devise a penalty function approach that does require any penalty parameter. Because all of their search spaces are defined in the real space, the real-coded GA were more suited in finding feasible solutions, but in all cases, that this approach " have a niche over classical methods to handle constraints".

In 2001, Beyer et al. [5] , analyzed the results of using the simulated binary crossover (SBX), the fuzzy recombination operator and the blend crossover (BLX), all with no mutation and tournament size 2 in a real-coded GA. The results indicated that those crossovers exhibit similar performance and linear convergence order.

Bhunia et al. [6] , in 2009, created a tournament GA application to solve an economic production lot-size (EPL) model.

Sawyerr et al. [23] , in 2015, proposed the RCGAu, hybrid real-coded genetic algorithm with "uniform random direction" search mechanism. They used tournament selection with size 3 to explore the suite of noiseless black-box optimization testbed and the results showed great performance

in some functions as f_7 and f_{21}.

Goldberg et AL [13] , in 1993, implemented a binary GA using the tournament selection as a convenient way to control selection pressure. They wanted to study the theory aspects of the GA focusing in the mixing building block. The actual effect of the tournament value was not investigated empirically in the work.

In 1999, Harik et al. [16] studied a simple GA and experimented with the tournament size. Their work shows that a given solution quality can be obtained faster by the tournament size equal to 4 or 8, instead of tournament size equal to 2 or 40.

Also, Harik et al. [15] implement a binary coded GA using tournament selection. They analyze the impact of the tournament size on their GA with a 100-bit one-max function. They inspected this implementation with tournament sizes of 2, 4, and 8. They observed that the proportion of the building blocks changed with the tournament size, being higher for smaller tournament sizes.

In 2007, Kaelo et al. [18] compared the usage of the tournament size in a real-coded GA. They discuss the impact of using tournament with tournament size of 2 and tournament size of 3. In their work, the results of comparing these two tournament sizes indicated that the tournament with size 3 is more greedy, reduces the number of functions evaluations greatly and also improves CPU usage, therefore, it impacts negatively in the success rate of their GA.

Also in 2009, Nicolau [22] implemented a binary GA to solve the noise-free BBOB 2009 testbed. Trough a quick experimentation with a subset of benchmark functions which yielded the following equation:

$$k = P/500, \text{ where P is the population size.} \quad (1)$$

From the works showed here, it is possible to understand that there is a preference to small values for the tournament size. We traced back to 1991, and we found out that it has been a popular choice ever since. However, we must highlight that these values are chosen arbitrarily and that Nicolau used a more pragmatic approach and set an empirical equation to his binary GA.

3 Experiment Design

3.1 The Genetic Algorithm

In this work, we implemented a simple real-valued Genetic Algorithm to explore the search space of the 24 noise free N dimension BBOB benchmark functions [14] .

Genome Representation and Evolutionary Tools.

Each individual is represented as real valued array, where each element is one input to a N dimension noise free benchmark BBOB function. Therefore, each individual of the GA has size of N real value elements. Each individual's values is set to the be into the interval [-4, 4], since the target functions are bounded by these values. It uses the Uniform crossover, elitism and Gaussian mutation as evolutionary operators. The relevant parameters were set as in the Table 1 and Table 2.

Number of Evaluations	40000
Population Size	500
Crossover chance	0.9
Mutation Chance	0.1
Elitism size	1

Table 1 GA Relevant Parameters

Mean	0
Standard Deviation	1
Ind. Prob. Attribute to be Mutated	0.1

Table 2 Gaussian Mutation parameters

Fitness Function.

The fitness function considered are the N dimensions noise free benchmark BBOB function [14] .

Implementation Details.

Chuang et al. [8] , proposed a Genetic Algorithm specific to run on the noise free BBOB benchmark functions. Some good practices presented in this work were incorporated to the GA. These practice are the concept of "restart strategy" and the concept of "alleviate stagnation".

The restart strategy concept is defined by Chuang et al. as: "For each restart, the initial population is uniformly and randomly sampled within

the search space. Whenever the restart condition is met, the algorithm will be reinitialized without using any information about the last test run. This process is iterated until the stopping criteria are met, i.e., maximum number of function evaluations has been reached, or the function value is less than the target precision."

The restart condition used in the GA is value of the standard deviation of the current population. If it is smaller that 10^{-12}, then the restart condition is triggered. Therefore, when it is contemplated the algorithm population is reinitialized randomly, as it is in the first initialization.

In the GA, we consider information about the last run, by utilizing the Elitism strategy on the best individual prior to the restart. The maximum number of function evaluations chosen is $(4 * 10^5)$ and the target precision chosen is (10^{-8}), values compatible with the ones in the previous work.

Genetic Algorithms and Benchmarks Functions.

In 2009, Nicolau [22] proposed the first Genetic Algorithm on the BBOB-2009 noiseless testbed. It was a simple binary Genetic Algorithm and its results showed good results on separable functions, but poor performance is achieved on the other functions.

In 2012, Chuang et al. [8] presented the "DBRCGA". Their results showed that the DBRCGA performs with difficulty in getting a solution with the desired accuracy for high conditioning and multimodal functions within the specified maximum number of evaluations, the DBRCGA presents good performance in separable function and functions with low or moderate conditioning.

In 2013, Holtschulte et al. [17] evaluated two Genetic Algorithm on the BBOB-2013 noiseless testbed. Their results highlight the importance of carefully chosen genetic operators.

Sawyerr et al found simliar results to the one achieved by Chuang et al. on their works in 2013 [24] and in 2015 [23] . Both the PRCGA and the RCGAu were tested on the BBOB-2013 noiseless testbed. They stated that both of them had difficulties to achieve solutions with the desired accuracy for high conditioning and multimodal functions. They also state that real value GA do not efficiently solve highly conditioned problems and studies have currently been carried out to find out why [23] .

3.2 Experiments

We made two experiments aiming to verify the impact of different values for the tournament size. In the first experiment, we analyzed the relation between the tournament size and the performance on the BBOB benchmark functions. In the second experiment, we analyze the impact of fluctuating the values of the tournament size *during* the execution.

We used the Friedman Test to determine whether any of the combinations shows a significant difference in the values given the tournament size values, represented as the quality of the result given functions and dimensions. In each of these tests, we set $\alpha = 0.05$. For this analysis, the dependent variable being the value obtained, the blocking variables are the functions and the treatment variable is the tournament size value.

Fixed tournament size

To analyze the impact of the tournament size into the GA in therms of quality of result, the following experiments were performed. First we analyse different values for the tournament size applied to the BBOB benchmark functions with 10, 20 and 40 dimensions. The tournament sizes were select, arbitrarily, from 2 to 25. For each combination of BBOB benchmark functions, dimensions, and tournament size we performed 40 repetitions.

Fluctuating tournament size

To analyze the impact of fluctuating the tournament size *during* the execution of the GA in therms of quality of result, the following experiments were performed. We start with the tournament size equal to 2, as many of the work cited on section 2.4. Then when half of the evaluations are completed, the tournament size is changed to another value, chosen from 2 to 25. These values were applied to the BBOB benchmark functions with 40 dimensions For each combination of BBOB benchmark functions, dimensions, and fluctuating tournament size we performed 40 repetitions.

4 Results

To get a finer intuition about the results, we show some visual examples, separated in two groups of Figures. The first group, shows the mean value achieved by the GA given a function. The second one, shows the convergence plot with the mean of the values found at each generation, the function target value with a given tournament size.

All Figures represent the mean of 40 repetitions given a certain dimension.

To better understand the results, we performed the Friedman Test analysis over all optimum values achieved from the GA.

4.1 First Group

The Figures 1, 2, 3, 4, 5, 6, 7, 8 and 9 exemplify that for the F5, F12, and F21 function (with 10, 20 and 40 dimensions) that changing the tournament size, does not lead to significant better final values found by the GA. The gray shaded area represents the 95% confidence level interval for predictions from a linear model for each scenario showed. For all Figures, the mean of 40 repetitions is shown as bullets and the bars represent the standard deviation, with 40 dimensions.

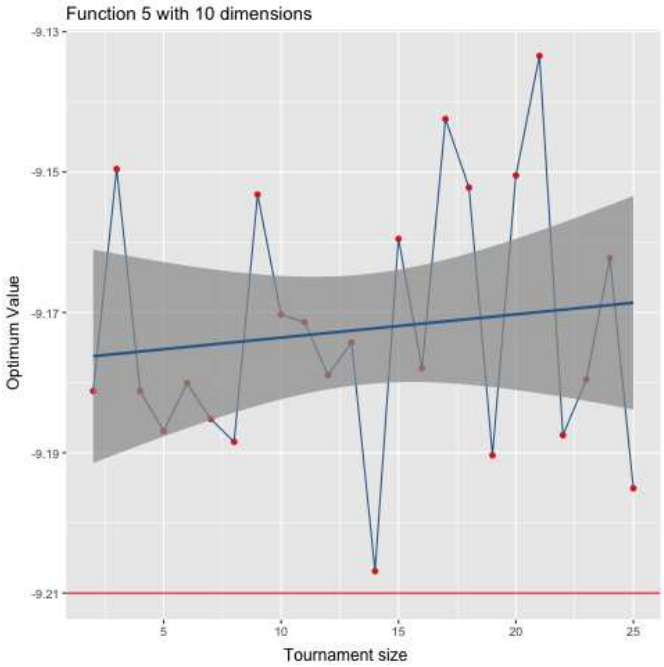

Fig. 1 Optimum values for the F5 function by the GA.

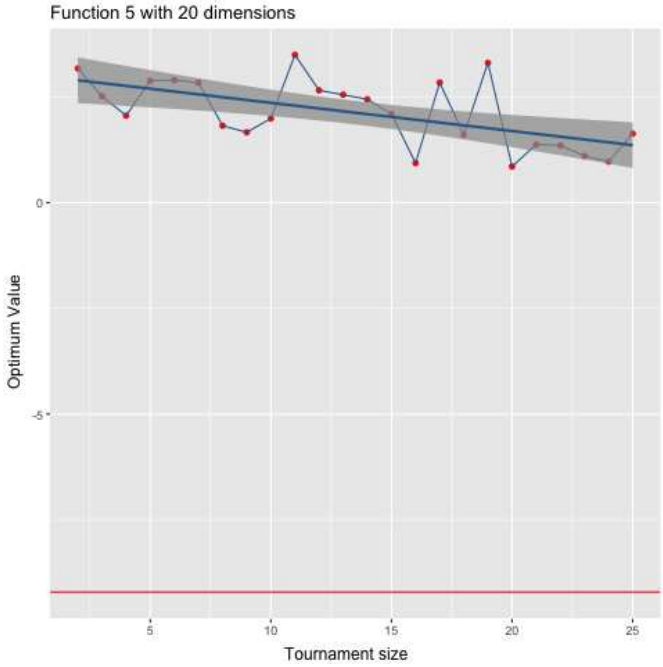

Fig. 2 Optimum values for the F5 function by the GA.

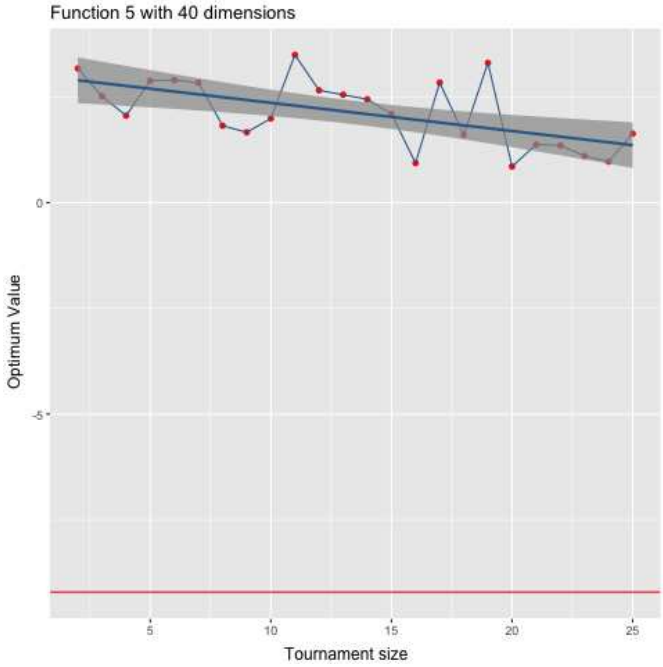

Fig. 3 Optimum values for the F5 function by the GA.

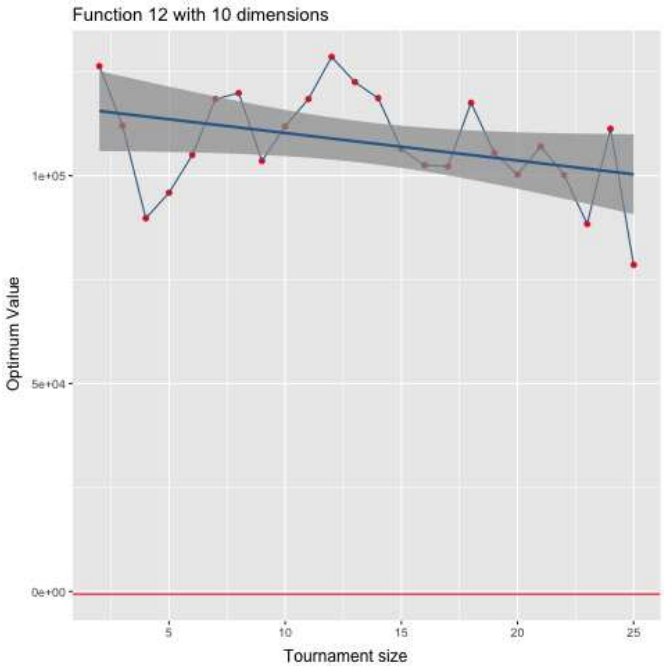

Fig. 4 Optimum values for the F12 function by the GA.

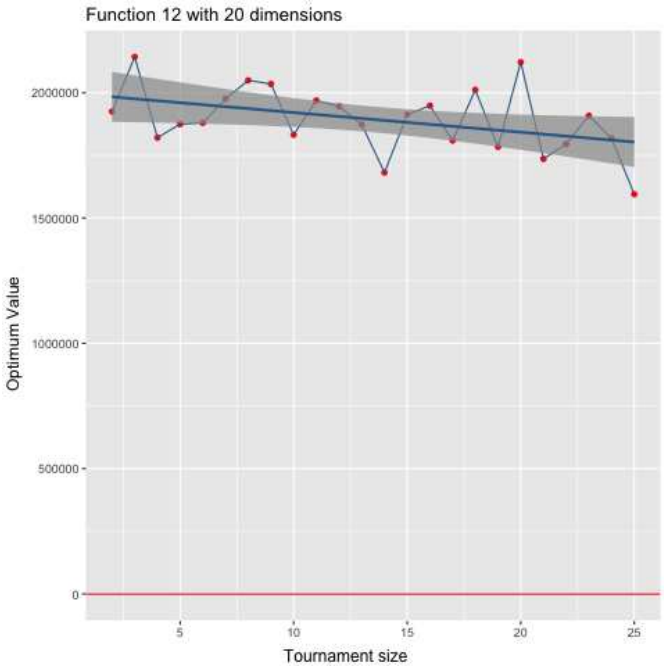

Fig. 5 Optimum values for the F12 function by the GA.

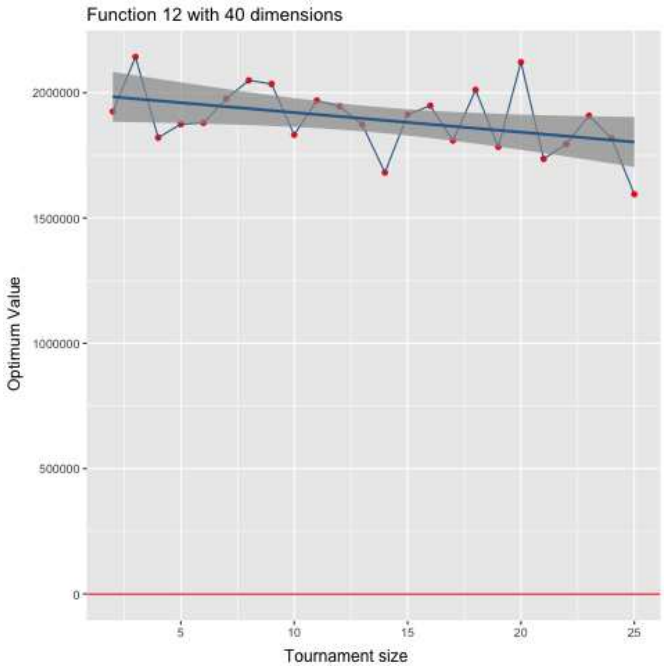

Fig. 6 Optimum values for the F12 function by the GA.

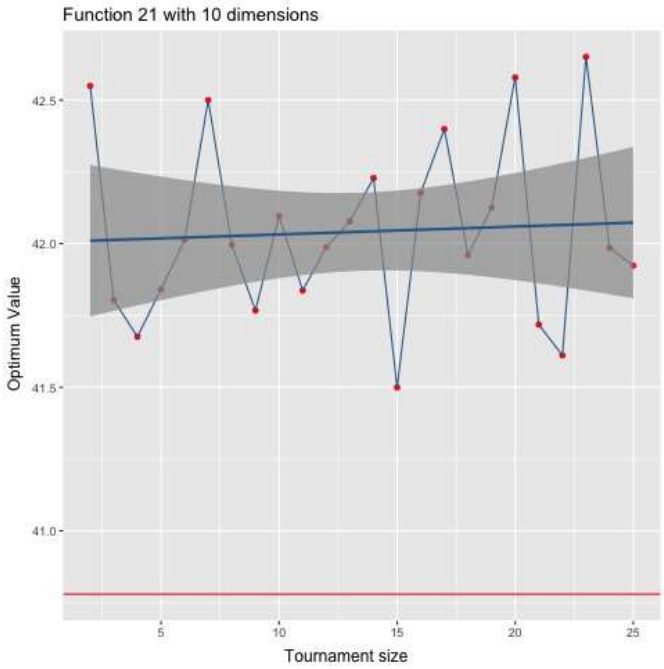

Fig. 7 Optimum values for the F21 function by the GA.

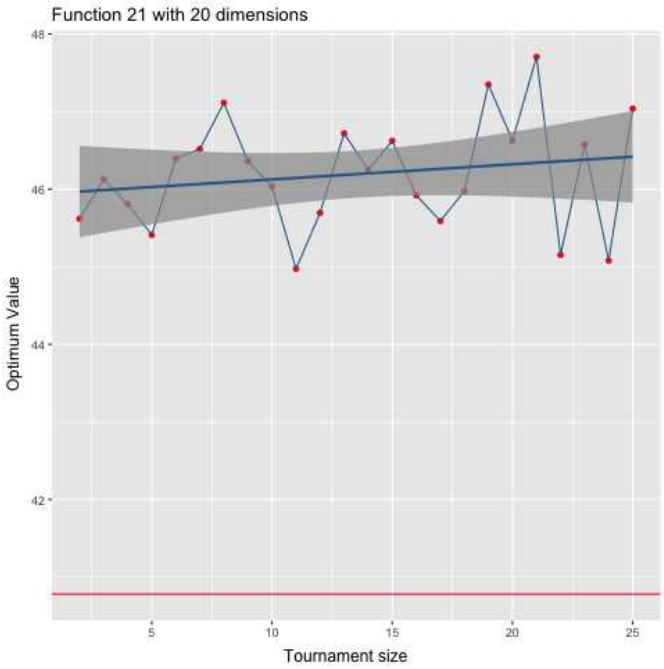

Fig. 8 Optimum values for the F21 function by the GA.

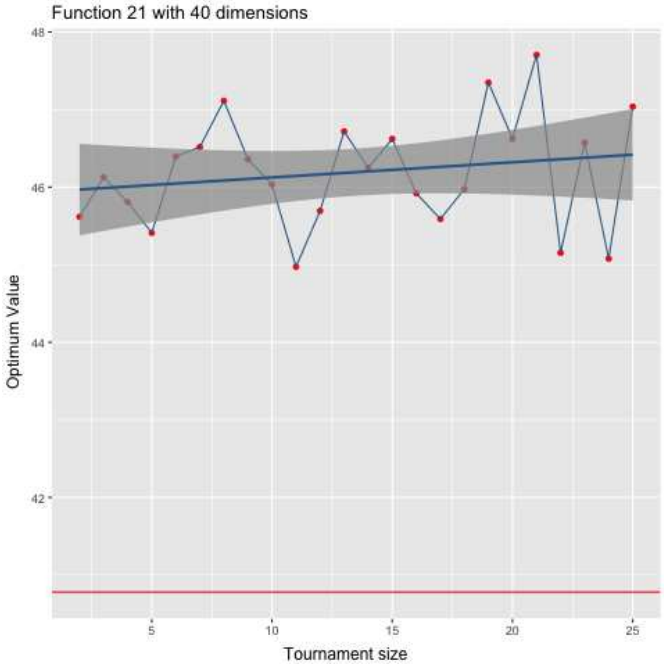

Fig. 9 Optimum values for the F21 function by the GA.

4.2 Second Group

The Figure 10 exemplifies that for the F1 function (with 10 dimensions), the GA is converging towards the optimum target value, represented by the red, vertical line.

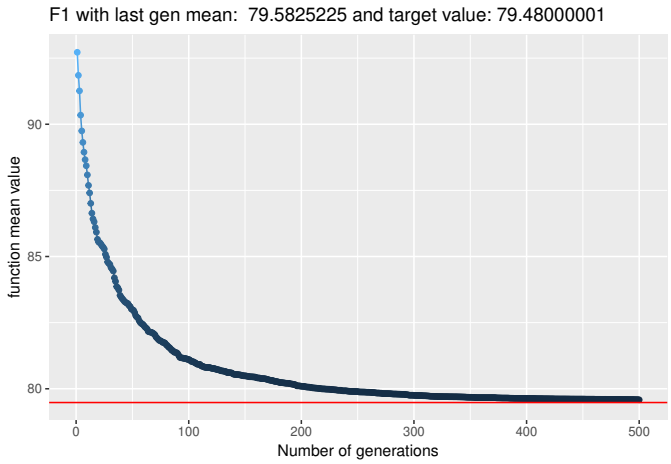

Fig. 10 Population convergence for function F1. The mean of 40 repetitions is shown, with 40 dimensions.

4.3 Friedman Test Analysis

The Friedman Test analysis showed that, as expected, there was no significant difference in values among tournament size values. The test was performed with the results obtained with 10, 20 and 40 dimensions, given the 24 BBOB benchmark functions. Detailed data can be visualized on the following Table 4.3.

Dimension size	Chi-squared	P-value
10	31.497	0.1111
20	27.903	0.2195
40	23.288	0.444

Table 3 Friedman Test results, with the dependent variable being the value obtained, the blocking variables are the functions and the treatment variable is the tournament size value.

5 Conclusion

In this paper we proposed an experimental analysis of the impact of the tournament size on the BBOB benchmark functions. First, we verified that there is no mathematical or experimental study that backgrounds any possible choice of value for the tournament size. Then, we searched into the literature, and realized that, in most cases, the most common values for tournament size are 2 or 3, though these values were used as a popular principle.

Based on that, we projected our two experiments. In the first one, we analyzed a group of tournament size values, ranging from 2 to 25 and applied a simple GA into the BBOB benchmark functions, given 10, 20 and 40 dimensions. The results showed that for any value of the tournament size did no impact was observed in improving the quality of the results for any functions in any dimension.

Further, we explored the fluctuation of the tournament size value trough the execution of the GA, by starting with the value 2 and then varied it, ranging from 2 to 25 and applied on the GA-BBOB, with the BBOB benchmark functions, given only 40 dimensions. This time also, no significant result was discovered.

We understand that there is much more that could be done in this field of study, therefore we propose that more investigations need to be done, in order to truly understand the role of the tournament size value related to the quality of final results. For that, we suggest that another kind of fluctuation, a more elaborated one, should be explored. Also, we would like to understand if our results were mainly related with the GA combination of parameters (crossover operator, mutation operator, selection operator, etc).

Hence most of the works already used tournament sizes equal to 2 or 3, and based on the results obtained allied with the complexity of the tournament selection being $O(n)$, we continue to advise that values of 2 or 3 should continue to be used as first choice.

;29MJ88%

1) Agrawal, R.B., Deb, K., Agrawal, R.: Simulated binary crossover for continuous search space. Complex systems 9(2), 115–148 (1995)

2) Aranha, C., Lavinas, Y.C., Ladeira, M., Enescu, B.: Is it possible to generate good earthquake risk models using genetic algorithms? In: Proceedings of the International Conference on Evolutionary Computation Theory and Applications. pp. 49–58 (2014)

3) Bäck, T., Fogel, D.B., Michalewicz, Z.: Evolutionary computation 1: Basic algorithms and operators, vol. 1. CRC press (2000)

4) Baker, J.E.: Reducing bias and inefficiency in the selection algorithm. In: Proceedings of the second international conference on genetic algorithms. pp. 14–21 (1987)

5) Beyer, H.G., Deb, K.: On self-adaptive features in real-parameter evolutionary algorithms. IEEE Transactions on evolutionary computation 5(3), 250–270 (2001)

6) Bhunia, A., Kundu, S., Sannigrahi, T., Goyal, S.: An application of tournament genetic algorithm in a marketing oriented economic production lot-size model for deteriorating items. International Journal of Production Economics 119(1), 112–121 (2009)

7) Blickle, T., Thiele, L.: A mathematical analysis of tournament selection. In: ICGA. pp. 9–16 (1995)

8) Chuang, Y.C., Chen, C.T.: Black-box optimization benchmarking for noiseless function testbed using a direction-based rcga. In: Proceedings of the 14th annual conference companion on Genetic and evolutionary computation. pp. 167–174. ACM (2012)

9) Deb, K.: An efficient constraint handling method for genetic algorithms. Computer methods in applied mechanics and engineering 186(2), 311–338 (2000)

10) Filipović, V.: Fine-grained tournament selection operator in genetic algorithms. Computing and Informatics 22(2), 143–161 (2012)

11) Goldberg, D.E.: Real-coded genetic algorithms, virtual alphabets, and blocking. Complex systems 5(2), 139–167 (1991)

12) Goldberg, D.E., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. Foundations of genetic algorithms 1, 69–93 (1991)

13) Goldberg, D.E., Deb, K., Thierens, D.: Toward a better understanding of mixing in genetic algorithms. Journal of the Society of Instrument and Control Engineers 32(1), 10–16 (1993)

14) Hansen, N., Auger, A., Finck, S., Ros, R.: Real-parameter black-box optimization benchmarking 2010: Experimental setup. Ph.D. thesis, INRIA (2010)

15) Harik, G., Cantú-Paz, E., Goldberg, D.E., Miller, B.L.: The gambler's ruin problem, genetic algorithms, and the sizing of populations. Evolutionary Computation 7(3), 231–253 (1999)

16) Harik, G.R., Lobo, F.G., Goldberg, D.E.: The compact genetic algorithm. IEEE transactions on evolutionary computation 3(4), 287–297 (1999)

17) Holtschulte, N.J., Moses, M.: Benchmarking cellular genetic algorithms on the bbob noiseless testbed. In: Proceedings of the 15th annual conference companion on Genetic and evolutionary computation. pp. 1201–1208. ACM (2013)

18) Kaelo, P., Ali, M.: Integrated crossover rules in real coded genetic algorithms. European Journal of Operational Research 176(1), 60–76 (2007)

19) Maza, M.d.l., Tidor, B.: An analysis of selection procedures with particular attention paid to proportional and boltzmann selection. In: Proceedings of the 5th International Conference on Genetic Algorithms. pp. 124–131. Morgan Kaufmann Publishers Inc. (1993)

20) Miller, B.L., Goldberg, D.E., et al.: Genetic algorithms, tournament selection, and the effects of noise. Complex systems 9(3), 193–212 (1995)

21) Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive models for the breeder genetic algorithm i. continuous parameter optimization. Evolutionary computation 1(1), 25–49 (1993)

22) Nicolau, M.: Application of a simple binary genetic algorithm to a noiseless testbed benchmark. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers. pp. 2473–2478. ACM (2009)

23) Sawyerr, B.A., Adewumi, A.O., Ali, M.M.: Benchmarking rcgau on the noiseless bbob testbed. The Scientific World Journal 2015 (2015)

24) Sawyerr, B.A., Adewumi, A.O., Ali, M.M.: Benchmarking projection-based real coded genetic algorithm on bbob-2013 noiseless function testbed. In: Proceedings of the 15th annual conference companion on Genetic and evolutionary computation. pp. 1193–1200. ACM (2013)

25) Sawyerr, B.A., Ali, M.M., Adewumi, A.O.: A comparative study of some real-coded genetic algorithms for unconstrained global optimization. Optimization Methods and Software 26(6), 945–970 (2011)

26) Tsutsui, S., Yamamura, M., Higuchi, T.: Multi-parent recombination with simplex crossover in real coded genetic algorithms. In: Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1. pp. 657–664. Morgan Kaufmann Publishers Inc. (1999)