Tournament vs Fitness proportionate

Ahmad Shabir Galili
29949009
asg1g18@soton.ac.uk

1. Introduction

This paper attempts to reproduce and expand on the premise explored in [7]. The paper explores the effectiveness of crossover on a simple fitness landscape over a mutation-based approach [2] for genetic algorithms. The landscape has been made such that a simple hill-climbing algorithm would struggle to find the peak fitness obtainable, illustrating the advantage of crossover in instances. The paper experiments with a landscape in which two building block assembly can navigate the landscape, but not mutation. The matter of crossover speeding up building block assembly is still a matter of research to this day [5].

The genotype this problem revolves around is a binary one of length 2n, where n is varied throughout the experiment. The fitness of the landscape is defined in equation 1, i and j are the number of ones in each half of the genotype; i is the number of ones in the first n bits and j the subsequent n bits. R is a matrix of uniformly distributed values between (0.5,1] with the sole purpose of introducing noise into the landscape. The values being as they are cause a plethora of local maxima across the landscape to prevent hill-climbers from merely ascending a single gradient. These values remain unchanged for each n value. Maximum fitness is achieved when the genotype contains ones in all of its nucleotide sites. Each of the genotypes is initialised randomly to a combination of zeroes and ones.

$$f(G) = R_{(i,j)}(2^i + 2^j) \tag{1}$$

The genotypes are sorted into 20 demes; each deme contained 20 genotypes following the multi deme island model explored in [8] to ensure diversity amongst the population. In each generation, every deme exchanged a single genotype chosen using fitness proportionate selection to another randomly selected deme. This migration occurs in both algorithms and allows for good alleles to spread throughout the population. The algorithm stops when the max fitness is observed in any one of the demes when i and j are both equal to n.

The experiment explored two approaches, one in which the algorithm utilised mutation to attempt to solve the problem, and another in which the mutation was combined with crossover. The mutation was applied to each site at a probability of $\frac{1}{2n}$ in both simulations. The crossover method implemented was the one-point crossover method, in which a random point is chosen within the genotype, and all points to the left-hand side of the point are chosen from parent 1, and the remaining from parent 2.

The methods incorporate a bit of elitism in the form of retaining the fittest genotype from the previous generation to ensure the method performs on par with a hill-climber in the worst-case scenario.

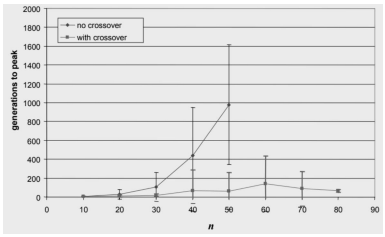
Outside of the fittest genotype being included in the next generation unaltered, and a migrant from another deme; the remaining genotypes in the next generation are chosen via fitness proportionate selection. This selection method has two forms of implementation. Both mimic a roulette wheel in their operation with each genotype being a segment. The proportion of the wheel each genotype occupies is proportionate to the fitness. The two aforementioned methods include one that spins the wheel numerous times and returns a single genotype every time, whereas the second method has numerous points around the edge it uses to get its values. The former method was implemented for this model. The proportion of the wheel dedicated to each genotype is governed by equation 2 where f(i) is the fitness of a genotype i.

$$P(i) = \frac{f(i)}{\Sigma_i f(i)} \tag{2}$$

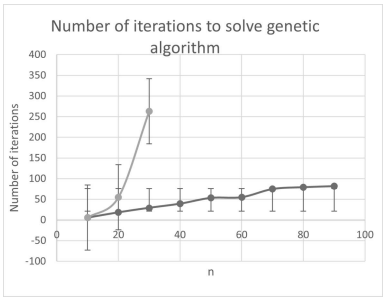
This approach is an example of the aforementioned building block hypothesis (BBH) in practice. The problem detailed requires the finding of the fittest genotype which is achieved when both i and j are equal to n. This objective can be split into two smaller objectives, maximising i and maximising j, the two building blocks. For this approach to work, the demes must maintain diversity such that some demes can maximise gene 1 and the others gene 2. These solutions will then ideally be put together by the crossover and consequently solve the problem. This is a demonstration of the two-building block assembly as there are two smaller objectives.

2. Reimplemented results

The resultant reimplemented graphs can be seen in 1 and 2. The graphs generated in the paper took an average of 30 trials, but the recreations are the result of 10 trials. A key point to note is several runs had to be discarded as they took over 2000 iterations to solve. These were not included in the average and instead a successful run to their place.

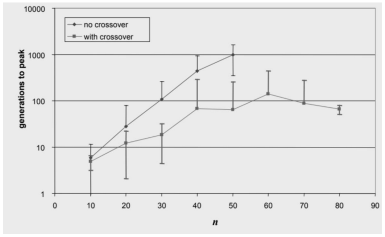


(a) Target Figure 3.

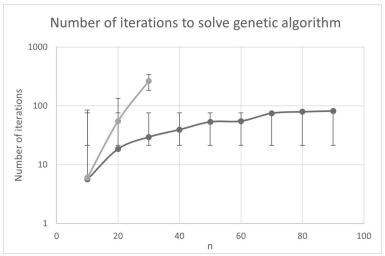


(b) Reproduced Figure 3.

Figure 1: Graph showing the number of iterations required to solve both the crossover and non-crossover case for varying n values. The error bars are one standard deviation. The reproduced results are an average of 10 runs.



(a) Target Figure 3(b).



(b) Reproduced Figure 3(b).

Figure 2: Graph showing the number of iterations required to solve both the crossover and non-crossover case for varying n values on a logarithmic scale. The error bars are one standard deviation. The reproduced results are an average of 10 runs.

2.1. Crossover

The results for the crossover case very much reflect the results in the original paper. As the number of sites in the genotype increase, so does the average number of iterations required to solve it. In the results showcased in the paper, this trend is also observed with the exception of the n = 60 case which as seen by the standard deviation for that case, could be due to a few large readings (anomalies) which do occasionally happen. The number of iterations on average is also very similar to those obtained for the crossover example.

One element which is not shown here is the number of trials that did not yield a solution in under 2000 iterations. As the length of the genotype increased, the likelihood of the max fitness being achieved decreased, with the case of n = 90 being solved on average once every 6 trials or so. This was not a problem for the n values 30 and below, which almost always achieved maximum fitness within 2000 iterations. An indicator of whether the algorithm would solve the problem was the ability for it to maintain diversity amongst the 20 demes. This is somewhat a matter of chance due to the random nature of the initialisation of the genotypes. For the genetic algorithm to work, some demes would need to find the case where i = n and others j = n so that crossover could merge these two genotypes. Instances where only i is solved or only j is solved caused the algorithm to get stuck in a cycle and not improve without relying on mutation. Relying on mutation, for problems with length n = 80 and thus 160 sites, is an unrealistic expectation as the mutation is likely to only change one site every generation and not necessarily always for the better.

2.2. Non-crossover

For the non-crossover example, there is a differentiation in results. The original paper can solve the problem until n = 50 in under 2000 iterations whereas the reimplementation was only able to solve it until n = 30. There was one instance where the n = 40 case was solved but this value was not recorded. The trend of the values that were obtained is in line with those in the paper. There is a larger disparity in the number of iterations required to solve for the n = 20 and 30 cases, but the same exponential behaviour is exhibited. This disparity was in the form of requiring more iterations to solve, for example in the original paper, 100 iterations were required for n = 30 compared to the average of 260 by this algorithm. A possible cause for this is a slight inaccuracy in reimplementation or an ill-defined parameter in the

original paper. The pseudorandom nature of initialisation could also have adversely affected the results.

This experiment highlights a situation where the crossover is vastly superior to a mutation-based approach. For every n value, the crossover based approach reached the fittest genotype faster than the mutation approach and consequently, meant it was able to solve for all n values. An interesting question that could be posed as a result of this could be the effect of removing the elitism. This will test the ability of the fitness proportionate selection to generate good children without a guaranteed best-case scenario.

3. Extension

The paper that was replicated was exploring the benefits of crossover versus a mutation-based algorithm for achieving maximum fitness. During the implementation of this project, one difficulty I encountered was replicating the results of the original paper in regards to the number of iterations it took to solve the genetic algorithm, specifically the non-crossover example. Whilst researching possible fixes for this disparity, I explored alternatives to fitness proportionate selection as found in [6]. This brought around my hypothesis, 'Tournament selection will achieve maximum fitness in fewer iterations than fitness proportionate selection'. This hypothesis will be explored in the landscape proposed in this paper.

To offer a fair comparison to the results already gathered, elitism in the form of carrying over the previous fittest genotype to the next generation unchanged is observed for this experiment too. In addition to the elitism, the migration rate was also unchanged with each deme having exactly one migrant in every generation. The deme that this migrant shall come from will be assigned randomly, but the specific row which will be sent over will be chosen using tournament selection. The mutation rate will remain the same at $\frac{1}{2n}$. Each deme will maintain all 20 genotypes every generation with the first 2 spots occupied by the maximum from the previous generation and the migrant respectively, and the remaining 18 being chosen using the tournament selection method.

3.1. Tournament selection

Tournament selection [3] is a method used to select the parents for the next generation of genotypes. It is an alternative to the fitness proportionate selection method and as seen in [10], performs better in the examples explored. This experiment was to show that the tournament selection method will perform better as it carries over more fit genes whilst maintaining diversity.

In the context of this experiment, for each child to be produced, five genotypes were chosen at random from within the deme and their fitnesses were compared. The genotype with the best fitness was used as a parent for the next generation, hence the name tournament. The number of genotypes chosen for each round is very significant, if too many were chosen then the probability the fittest genotype would appear in each sample would be high and thus, there would not be much diversity in the deme. A lack of diversity would result in the algorithm getting stuck at a local maximum which needs to be avoided, but too much diversity would result in the algorithm taking an excessive amount of time to solve, if at all.

The value of 5 was chosen with these factors in mind as there is a one in four chance the fittest genotype would be a part of the sample meaning in every generation, you would expect ten of the children to be a product of the fittest in the crossover example.

3.2. Purpose of investigation

A key part of genetic algorithms is ensuring it yields results and in a timely manner. This paper sought to offer the crossover method as an alternative to a mutation-based approach as it is better suited to solving those problems. Tournament selection is another change that can be made to these algorithms to yield results and faster. When undertaking this exercise, this was something that became apparent quickly as especially for cases where the n value became particularly large, it was extremely time-consuming to run the algorithm. This was particularly problematic as the algorithm often would not conclude within the assigned 2000 iterations for said larger n values resulting in runs needing to be restarted.

In recent times, research has focused on integrating Markov chains into tournament selection methods with a non-elitist algorithm [1]. Elitism negatively impacts the diversity of the population as it forces in the previous fittest genotype which may not necessarily be positive.

3.3. Prediction

I think this experiment will show that tournament selection will solve the genetic algorithms faster on average in comparison to the fitness proportionate selection approach. The fitness proportionate method typically does not assign as high a probability for the selection of the fittest genotype as the tournament selection does. As previously mentioned in the crossover example, in a deme of 20 genotypes, you would expect 10 children in the next generation to have one of the parents be the fittest genotype. On the other hand in the fitness proportionate example, this number would be far fewer as although it would have the highest probability of selection, it would still have somewhere in the region of 10%, but this number varies in every generation.

This will not always be a positive change, however, as it results in less diversity within demes which raises the issue of local maxima. This problem can be subverted by the various demes initialising very sporadically which could compensate for the lack of diversity within demes, but this is

leaving the matter to chance. For larger values of n, I think the consequences of this will be highlighted as they may require more runs before successful completion.

4. Extension results

In order to compare the two selection methods, 10 trials of the tournament selection method were also completed for the various n values.

4.1. Crossover

In the crossover case, for every n value, the tournament selection method outperformed fitness proportionate selection. The difference between the two also increased with increasing n as in the n = 10 case the difference is only 1, but in the n = 60 case, this had increased to the fitness proportionate selection taking on average 15.3 more runs to find the maximum fitness as shown in 3.

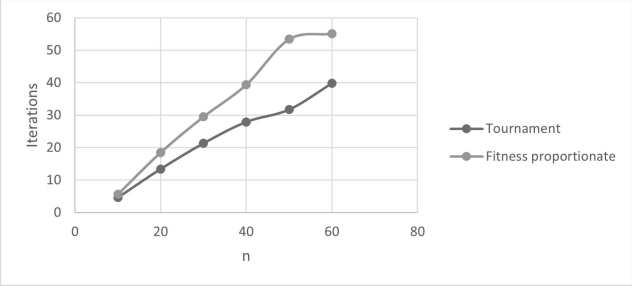


Figure 3: Graph showing the number of iterations required to solve the crossover genetic algorithm based on an average of 10 runs. The graph shows for every n value, the algorithms which use tournament selection outperform the ones using fitness proportionate selection.

There were some problems with the tournament selection method as anticipated in the prior section. For example larger n values, comparative to the fitness proportionate selection, it would on average take more runs for the algorithm to find the solution within 2000 iterations. This can be seen in Table 1, which shows for all n values that were experimented with tournament selection took more runs to complete five algorithms. This experiment does however have a flaw, as the random initialisation and R values could result in some runs finishing earlier than others but to consistently be outperformed could indicate that there is some underlying cause.

4.2. Non-crossover

The non-crossover case further supported the hypothesis as tournament selection achieved maximum fitness before fitness proportionate selection as shown in Figure 4. For n = 10, the two selection methods once again have little to differentiate the two of them, but for the remaining n values, it is clear that tournament selection significantly outperforms the alternative. This is highlighted by the n = 20 case which requires on average 150 fewer iterations to be solved by tournament selection compared to fitness proportionate selection.

Table 1: Table showing the number of trials the two methods required in order to achieve maximum fitness 5 times.

	Number of trials to solve 5 times	
n	Fitness proportionate	Tournament
50	6	8
56	12	22
58	7	18
60	11	14
62	10	14

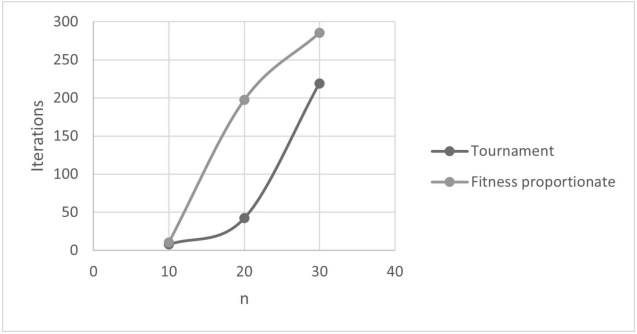


Figure 4: Graph showing the number of iterations required to solve the non-crossover genetic algorithm based on an average of 10 runs. The graph shows for every n value, the algorithms which use tournament selection outperform the ones using fitness proportionate selection.

The equivalent to Table 1 for the non-crossover case was not produced as the algorithms could not solve beyond an n value of 30 so it was redundant.

An interesting observation in this experiment was the tournament selection was still unable to produce the highest fitness genotype with 80 sites within 2000 iterations. This outcome was a slight surprise as given n = 30, was solved on average in 220 iterations, I expected allowing the algorithm to run for an hour, retrying every time it exceeded 2000 iterations, would produce a single instance of this fit genotype but it did not solve it. An explanation for this is the mutation-based approach is still far too overly dependent on the unlikely statistical event of switching all sites to 1's without flipping back.

4.3. Non-crossover hill climb

The results recorded thus far are in line with the original hypothesis, but this was done on a single equation. This is

not sufficient to be able to draw any definitive conclusions about the matter so another equation was tested, equation 3. This equation also has the same fittest genotype with all 1's as with the original equation but no noise in the landscape, making it a simple oneMax hill-climber problem [4]. Given that both tournament selection and fitness proportionate selection would be able to solve this problem with ease, this run was done with the mutation method without crossover.

$$f(G) = 3i^2 + 2j \tag{3}$$

The reason the equation was changed is as the fitness values will massively be reduced for this function in comparison to the original equation. This difference does not effect the tournament selection method, but changes the probabilities for the fitness proportionate selection method.

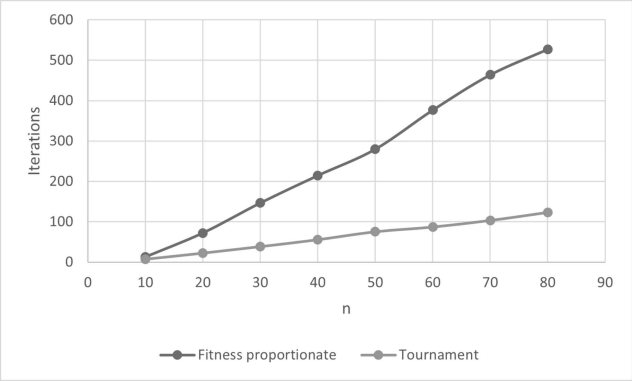


Figure 5: Graph showing the number of iterations required to solve $3i^2 + 2j$.

Figure 5 was produced by averaging 10 runs for each algorithm and just like all previous experiments support the statement that tournament selection will require fewer iterations to achieve the maximum fitness genotype. Of all results thus far, Figure 5 shows the biggest differentiation in results as the gradient of the fitness proportionate results is significantly steeper at 6.2 than the tournament selection at 1.5. The number of iterations for both selection methods is increasing linearly, but the tournament selections method allows it to produce more children of the fittest genotype which is especially helpful when the landscape has only one gradient to navigate. This distinction results in the algorithm being able to climb much faster than fitness proportionate selection.

5. Conclusion

The hypothesis this extension was testing was whether the tournament selection would solve genetic algorithms faster than fitness proportionate selection. This was tested using two different fitness landscapes, one from the paper this project was reimplementing [7] and another devised to test the ability on a oneMax landscape. The experiments support the hypothesis stated, and the results reinforce the findings of [1] which states that tournament selection solves genetic algorithms in fewer iterations. For both landscapes that results were collected for, regardless of the length of genotype, the tournament selection algorithm achieved maximum fitness fastest. In the oneMax landscape, it required on average 50 iterations fewer for every additional 20 sites in the genotype.

The difference in performance on the noisy landscape defined by Equation 1, was not as stark as the oneMax, but as n increased so did the number of iterations between the two algorithms. Although the data suggests the initial hypothesis was correct, this is by no means definitive proof. This was only tested on two landscapes which could have favoured tournament selection. In more complex landscapes where maximum fitness is not all 1's or all 0's and there are more local maxima, tournament selections' overreliance on the fittest genotype could be its downfall as the diversity amongst the demes decrease.

One aspect that was also troubling in regards to tournament selection was the fact that the data collected in Table 1 suggests it is more likely to run simulations that do not solve in a timely manner. The method deployed to obtain these results is also slightly problematic as it did not control the random initialisation which could influence the algorithms' ability to solve the problem but attempts were made to offset this by setting a random seed for each run and taking multiple readings and averaging the output.

The initial reimplementation was able to reproduce the results presented in [7] for the crossover algorithm but was not able to attain optimal solutions for the mutation algorithm in the same ballpark as the paper. This could be due to several factors, such as the initialisation of the demes and possibly a slight inaccuracy in the algorithm reimplementation but the trend shown is identical in exponential nature to the original paper. The crossover is shown to perform better in this specific fitness landscape but again this should not be overstated.

In order for a definitive conclusion on the two methods of implementation, several trials on a variety of fitness landscapes are required and even in such a scenario a conclusion being drawn definitively is unlikely. The same is true for the two selection methods as although tournament appears superior, it is for limited sample size and fitness proportionate selection being the de facto standard for so long indicates the merit of its ability to maintain diversity amongst its population. Research is still ongoing to find variations of fitness proportionate selection which could enhance the existing performance such as those in [9].

This paper has shown that the paper by Watson [7] has mostly reproducible results and given an example of the advantage of tournament selection over fitness proportionate

selection. The extension showed the effectiveness of tournament selection in contrast to fitness proportionate selection but highlighted that this increased speed of convergence comes at the cost of reduced diversity and consequently more iterations required to solve the problem. The findings support the work done by [1] and [10].

References

[1] Anton V. Eremeev. On proportions of fit individuals in population of mutation-based evolutionary algorithm with tournament selection. *Evolutionary Computation*, 26(2):269–297, 2018.

[2] Stephanie Forrest and Melanie Mitchell. Relative building-block fitness and the building-block hypothesis. In *Foundations of genetic algorithms*, volume 2, pages 109–126. Elsevier, 1993.

[3] Brad L Miller, David E Goldberg, et al. Genetic algorithms, tournament selection, and the effects of noise. *Complex systems*, 9(3):193–212, 1995.

[4] Melanie Mitchell, John Holland, and Stephanie Forrest. When will a genetic algorithm outperform hill climbing. *Advances in neural information processing systems*, 6, 1993.

[5] Dirk Sudholt. How crossover speeds up building block assembly in genetic algorithms. *Evolutionary computation*, 25(2):237–274, 2017.

[6] Dirk Thierens and David Goldberg. Convergence models of genetic algorithm selection schemes. In *International conference on parallel problem solving from nature*, pages 119–129. Springer, 1994.

[7] Richard A Watson. A simple two-module problem to exemplify building-block assembly under crossover. In *International Conference on Parallel Problem Solving from Nature*, pages 161–171. Springer, 2004.

[8] Sewall Wright. Evolution in mendelian populations. *Genetics*, 16(2):97, 1931.

[9] Fengrui Yu, Xueliang Fu, Honghui Li, and Gaifang Dong. Improved roulette wheel selection-based genetic algorithm for tsp. In *2016 International conference on network and information systems for computers (ICNISC)*, pages 151–154. IEEE, 2016.

[10] Jinghui Zhong, Xiaomin Hu, Jun Zhang, and Min Gu. Comparison of performance between different selection strategies on simple genetic algorithms. In *International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06)*, volume 2, pages 1115–1121. IEEE, 2005.