Statistical optimisation and tuning of GA factors

Andrei Petrovski
School of Computing
The Robert Gordon University
St. Andrew Street
ABERDEEN, AB25 1HG
ap@comp.rgu.ac.uk

Alexander Brownlee
School of Computing
The Robert Gordon University
St. Andrew Street
ABERDEEN, AB25 1HG
sb@comp.rgu.ac.uk

John McCall
School of Computing
The Robert Gordon University
St. Andrew Street
ABERDEEN, AB25 1HG
jm@comp.rgu.ac.uk

Abstract- This paper presents a practical methodology of improving the efficiency of Genetic Algorithms through tuning the factors significantly affecting GA performance. This methodology is based on the methods of statistical inference and has been successfully applied to both binary- and integer-encoded Genetic Algorithms that search for good chemotherapeutic schedules.

1 Introduction

Genetic Algorithms (GAs) have been extensively studied in computer science and used in real-world applications to find one or a series of good (optimal) solutions from a vast search space [6]. The speed of exploring the solution space by GAs and the quality of the solutions found are affected to a great extent, and sometimes even determined, by the design of Genetic Algorithms – in particular, by the type of encoding, selection, genetic operators, and the values of their salient parameters.

Although GAs have been applied to a wide range of problems in numerous areas of science and engineering, there is no generally accepted methodology of theoretical and experimental analysis of the influence of the operators and parameters (including the interactions thereof) involved in the design of Genetic Algorithms. The insight into which are the most relevant and influential factors in GA desing becomes even more important when Genetic Algorithms are applied to large or hard real-world problems [13]. When this happens, either the time necessary for GAs to find an acceptable solution significantly increases or the quality of final solutions decreases, which motivates the designer of Genetic Algorithms to fine tune GA factors.

However, the stochastic nature of interactions within a GA population makes it difficult to determine the effect of each individual factor adjustment on the population dynamics. When analysing the influence of these factors, most attention should be given to the ones that affect GA performance in a statistically significant way. Then, it would be possible to avoid the neccessity for a detailed analysis of different GA configurations that lead to Genetic Algorithms with very similar behaviour patterns.

Most commonly GA practitioners set the values of GA factors through a process of trial and error, or applying a case-based approach from related or similar problems in the literature. There exist, however, a number of algorithmic schemes to improve efficiency of GAs [2]:

- ad hoc GA factor tuning;
- parallelisation and meta-GA optimization;
- adaptation tuning and hybridisation.

In our previous work [12], we developed a more rigorous methodology for tuning GA factors, which was validated and thoroughly examined by Czarn *et al.* in a more recent study [4]. This methodology is based on statistical analysis of the relationship between the factors and performance of Genetic Algorithms, which involves factorial experimental design, the ANalysis Of the VARience (ANOVA), regression modelling, and response curve analysis. The motivation for developing and applying this methodology is to enrich the designer of Genetic Algorithms with a reliable tool for selecting the values of controllable factors that significantly affect algorithms' performance.

In Section 2 the optimisation problem of cancer chemotherapy is described, which will be used as a benchmark for our methodology. Section 3 gives the implementation details of our experiments. The results of statistical tuning of GA factors are presented in Section 4. Section 5 concludes this paper with the discussion on the meaning of the results obtained and the directions of future work.

2 Background

One of the important merits of Genetic Algorithms is their ability to tackle real-world problems, which are very hard or even unsolvable by traditional optimisation techniques. One such problem is the composition of an optimal schedule for anti-cancer chemotherapy treatment. This is a non-linear optimal control problem that is subject to contradictory constraints [8]. As can be seen from [1], [14] and [15], Genetic Algorithms favourably compare with other search heuristics in this problem domain.

A detailed description of the problem is given in our previous paper [11]; here we only provide the details of how anti-cancer treatments can be represented and evaluated.

2.1 Chemotherapy treatments as GA chromosomes

For multi-drug treatments the solutions to the problem of chemotherapy optimisation may be expressed as decision vectors $\mathbf{c} = (C_{ij}), i \in \overline{1,n}, j \in \overline{1,d}$ of n discrete doses for each of the d anti-cancer drugs used. Using the Genetic Algorithms' terminology, the representation space $\mathbf{I}$ (a discretized version of the search space Ω) can then be expressed as a Cartesian product:

$$\mathbf{I} = A_1^1 \times \ldots \times A_1^d \times \ldots \times A_n^1 \ldots \times A_n^d \quad (1)$$

of allele sets A_i^j. Each allele set uses a 4-bit representation scheme

$$A_i^j = \left\{ a_1 a_2 a_3 a_4 : a_k \in \{0,1\} \forall k \in \overline{1,4} \right\} \quad (2)$$

so that each drug dose C_{ij} takes an integer value in the range of 0 to 15 concentration units. In general, with n treatment intervals and up to 2^p concentration levels for d drugs, there are up to 2^{npd} individual elements. Henceforth we assume that $n = 10$ and that the number of available drugs is also restricted to ten.

The values $n = 10$ and $d = 10$ result in the representation (search) space of power $|\mathbf{I}| = 2^{400}$ individuals, referred to as chromosomes.

Thus, a chromosome $x \in \mathbf{I}$ can be expressed as

$$x = \left\{ a_1 a_2 a_3 \ldots a_{4nd} : a_k \in \{0,1\} \; \forall k \in \overline{1,4nd} \right\} \quad (3)$$

and the mapping function $m : \mathbf{I} \rightarrow \mathbf{C}$ between the individual $\mathbf{I}$ and the decision vector $\mathbf{C}$ spaces can be defined as

$$C_{ij} = \Delta C_j \sum_{k=1}^{4} 2^{4-k} a_{4d(i-1)+4(j-1)+k}, \; \forall i \in \overline{1,n}, \, j \in \overline{1,d} \quad (4)$$

where ΔC_j represents the concentration unit for drug j. This function symbolizes the decoding algorithm to derive the decision vector $\mathbf{c} = m(x)$ from a chromosome x.

If this vector violates any of the constraints imposed on cancer chemotherapy treatment, detailed in [11]; penalties are applied to the fitness function based on the following optimisation objective:

$$\underset{\mathbf{c}}{\text{maximise}} \quad J(\mathbf{c}) = \int_{t_1}^{t_n} \ln\left(\frac{\Theta}{N(\tau)} \right) d\tau \quad (5)$$

which corresponds to minimising the overall tumour burden during treatment [8].

If a decision vector does not break any of the constraints then we will call it *feasible*. In the solution space Ω all feasible vectors comprise a feasible region, which will be the target of our search. The efficiency of Genetic Algorithms applied to our problem is measured by the time necessary for GAs to find at least one feasible solution. This time depends on a number of factors and is characterised by a random variable, the descriptions of which follow.

2.2 Measure of GA efficiency

In order to analyse the performance of Genetic Algorithms we introduce a random variable that characterises the efficiency of GAs. It has been found in [12] that a good measure of efficiency is the number of generations Ψ, which are required in order to reach the feasible region in the solution space.

Due to the stochastic nature of GA search, Ψ is a random variable. Our previous study has established that Ψ has the Weibull probability distribution, and the distribution of $\log(\Psi)$ can be approximated by the Gaussian curve to a reasonable degree of accuracy [12]. This enabled us to apply ANOVA for analysing the effect of tuning the GA factors.

2.3 Factors affecting the efficiency

Genetic Algorithms possess explorative features, characterised by the population size and the probability of mutation, and exploitative features, characterised by the type of selection used, by the probability of crossover and by the crossover operator itself. However, the mere presence of these features does not guarantee the efficiency of GA search. It is necessary to strike the right balance between explorative and exploitative features of Genetic Algorithms.

These features in the present paper are specified by the following factors:

- probability of crossover - ϕ_1;
- probability of mutation - ϕ_2;
- selection method - either tournament or linear ranked roulette wheel selection [10] - ϕ_3;
- crossover method - either weighted average [2, 9] or standard two-point [10] - ϕ_4;
- mutation method - either convex space or number creep [5] - ϕ_5;
- creep mutation step [5] - ϕ_6;
- fitness normalisation slope [5] - ϕ_7.
- population size - ϕ_8.

The variation of these factors will have different effects on the efficiency of Genetic Algorithms. In the following section we demonstrate a methodology that enables us to identify the GA factors most significantly affecting the performance and to fine tune the values of such factors.

3 Statistical Inference

Having established the measure of GA efficiency and having specified the factors that most likely affect it, we can now attempt to find a mathematical formula that expresses this measure in terms of such factors:

$$\Psi = \Psi(\phi_i) \quad i \in \overline{1, l} \tag{6}$$

where l is the number of GA factors that significantly affect the performance. Then the problem of GA efficiency improvement can be confined to the task of finding the factor values ϕ_i^{opt} that optimise the performance measure (6).

This task will be accomplished in three steps. First of all, a screening experiment will be conducted reducing the number l of GA factors that need to be included into the model (6). Only significant factors, variation of which noticeably (in a statistical sense) affects the performance, will remain in the model.

After the screening experiment, a regression model of Ψ in terms of significant factors will be obtained. Using this model, standard calculus techniques can be applied in order to determine the optimal values of the significant GA factors.

However, before commencing the statistical analysis of GA performance, let us introduce a more advanced (in comparison with the one developed in [12]) representation of the solution space Ω.

3.1 Integer representation of cancer chemotherapy treatments

Traditionally, Genetic Algorithms employ fixed-length binary encoding such as that already shown to work for the cancer chemotherapy problem [11, 12]. Arguments for such an encoding are based on its allowance for a higher degree of parallelism and on the fact that each chromosome possesses the maximum number of schemata [9, 10]. Furthermore, it can be argued that binary encoding can represent any data, thus allowing the basic mutation and crossover operators to be used in a wide range of applications. Given its widespread use and established knowledge base, the authors chose binary encoding to initially represent solutions to the optimisation problem of cancer chemotherapy.

However binary encoding is not the only way to represent such solutions. One of the alternatives is to use a string of integers – in the cancer chemotherapy problem this results in each string of four bits being replaced by a decimal number of value 0-15. This has multiple advantages:

- It is more natural to represent numbers in this way, making the final implementation easier to understand and maintain [10].
- Keeping the numbers in their raw (integer) form enables us to preserve any domain-specific knowledge associated with the problem [5].
- Integer encoding – and indeed any real number encoding – opens up the possibility of new crossover and mutation operators more suited to the data. Rather than flipping random bits while undergoing mutation, values can be adjusted more predictably; similarly, more flexible mathematical operations may be used to implement crossover [9].

Additionally, while Holland's schema counting argument predicts that alternative encodings will yield poorer performance compared with their binary equivalents, empirical results have shown that the opposite is true in certain situations [2, 10]. It seems that correct choice of encoding is highly problem-specific and it is hard to predict whether one will be better than another.

In the present study an attempt will be made to show that integer encoding does result in a better performance for the cancer chemotherapy problem compared to the original binary encoding. This will be proved by first implementing and optimising an integer-based version of the original cancer chemotherapy GA, and then comparing its performance with that of the binary-encoded GAs on the same optimisation problem.

3.2 Screening experiment

In order to ascertain how many of the GA factors specified in Section 2.3 significantly affect the performance, we will use a 2^{8-2} fractional factorial design for a screening experiment. Factorial designs allow the study of multiple factors in the same experiment and the assessment of the manner in which these factors interact [3].

In the present context, the interaction between factors refers to the possibility of one factor producing different effect on the response variable when the value of another factor is changed. Examining interactions is important because a significant interaction means that the effect of one GA factor cannot be considered independently of the others.

Czarn *et al* rightly pointed out that different pseudorandom number generator seeds may contribute to significant differences in the value of the response variable Ψ. Grouping experimental runs into homogeneous blocks, each of which starts with the same number generator seed, limits the cause of variation within blocks to the factors under study, thereby reducing the noise and sharpening the comparisons [4].

This measure has been introduced in the following screening experiment; so was the workup procedure that attempts to balance an ANOVA design, which is also explained in [4].

ANOVA, as described in [3], essentially splits the total variations in the values of Ψ into variation contributed by the GA factors $\phi_i, i \in \overline{1,8}$, by their interactions, by blocking, and by error. Error is expressed in terms of residuals, which are simply random deviations of the observed values from the expected values under the assumption that there is no deterministic effect.

In order to ascertain that a GA factor ϕ_i has a statistically significant effect, we compare the variation contributed by the factor with the variation contributed by random error. The ratio of these two contributions is referred to as an F-value; it enables us to determine the probability, called the p-value, of observing such F-

value under the hypothesis that ϕ_i does not significantly affects the performance.

If the p-value is equal or less than a chosen level of significance α (usually 0.01 or 0.05), this would suggest that a particular GA factor has a significant effect upon the response variable Ψ.

The results of the enhanced fractional factorial (FF) experiment are given in Table 1.

TABLE 1. *2^{8-2} FF Experiment*

Factor	F-value	p-value
Constant		0.000
ϕ_1	-1.5	0.925
ϕ_2	**-217.6**	**0.000**
ϕ_3	20.3	0.212
ϕ_4	-7.3	0.650
ϕ_5	-15.4	0.341
ϕ_6	**-68.9**	**0.000**
ϕ_7	-28.9	0.080
ϕ_8	**-174.6**	**0.000**

The analysis of the screening experiment is presented in Figure 1 where the significance of each GA factor is shown in the form of a histogram :

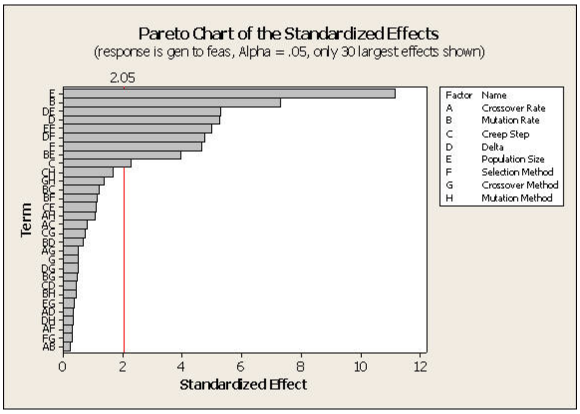

Figure 1. MINITAB analysis of the screening experiment

As can be seen from Table 1, only three GA factors, viz. the probabilities of mutation (ϕ_2), creep mutation step (ϕ_6), and population size (ϕ_8) significantly affect the performance. The effects of other factors are indistinguishable from the effect which might be caused by random errors of performance measurements; thus, the other factors will be excluded from further analysis. The significant factors ϕ_2, ϕ_6, and ϕ_8, on the other hand, will be examined more thoroughly in the next section, where a regression model will be constructed in terms of these factors.

3.3 Regression model

When studying continuous factors, it is interesting to find conditions (values of the factors) that lead to a particular response, usually minimum or maximum. The responses of an experiment when considered as a function of the possible values of the factors form a *response surface*, which is usually expressed in the form of a regression model.

The performance measure is obtained by repeating each GA run 30 times. The same set of 30 random populations was generated for each factor setting. Each population was allowed to run to a maximum of 500 generations.

The response surface experiment uses a central composite design [12], the results of which are given in Table 2.

TABLE 2. *Response Surface Experiment*

Factor	Coefficient	p-value
Constant (C)	**1536.24**	**0.000**
ϕ_2	-252.53	0.829
ϕ_6	**-311.61**	**0.002**
ϕ_8	**-23.65**	**0.000**
ϕ_2^2	427.01	0.756
ϕ_6^2	25.92	0.007
ϕ_8^2	**0.13**	**0.000**
$\phi_2\phi_6$	4.76	0.972
$\phi_2\phi_8$	-2.63	0.647
$\phi_6\phi_8$	**1.56**	**0.004**

Using the coefficients given in Table 2 we are able to build the following regression model:

$$\Psi = C - 311.61\phi_6 - 23.65\phi_8 + 0.13\phi_8^2 + 1.56\phi_6\phi_8 \quad \textbf{(7)}$$

Figure 2 depicts the relationship between the response variable and the first two significant factors (similar plots can be viewed for other significant factors and their interactions).

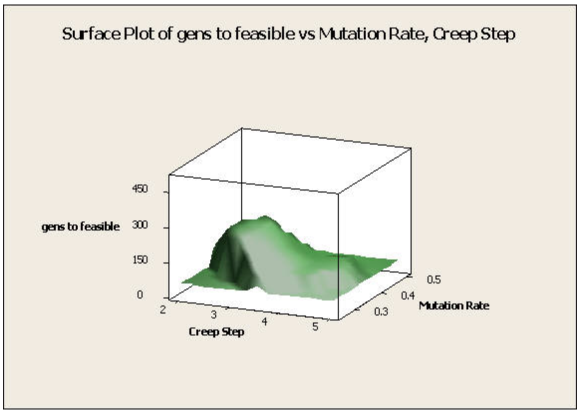

Figure 2. Ψ as a function of ϕ_2 and ϕ_6

The regression model can be used to find the optimal values of the significant GA factors. These values are obtained by differentiating (7) with respect to each factor in turn, setting each derivative equal to zero and solving the resulting system of equations. Table 3 gives the optimal value for each factor deemed to be significant by the screening experiment:

TABLE 3. *Optimal values of GA factors*

Factor	Optimal value	Rounded value
ϕ_2	**0.092**	n/a
ϕ_6	**3.54**	**4**
ϕ_8	**75.675**	**76**

4 Results

In order to verify that the factors' estimates obtained from the regression model really fine tune an integer-encoded GA, we will compare its performance with that of the best binary-encoded Genetic Algorithm found for this problem in our previous study [12].

Two comparisons will be made: with respect to the number of generations necessary for finding a feasible solution and with respect to the maximum fitness found in a specified number of generations (NB. The latter test ensures that the integer encoding still yields the same quality of solutions). These comparisons were made by running the binary- and integer-encoded GAs 200 times with the same set of 200 random starting populations.

Figure 3 and Table 4 summarise the results of these comparisons:

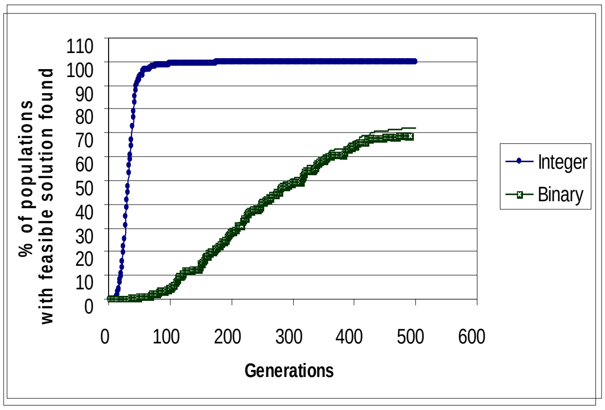

Figure 3. Binary- vs. integer-encoded GAs in finding the feasible region in Ω

Figure 3 shows a clear advantage in using the integer encoding. Not only do a higher percentage of populations reach a feasible solution in the same time, but all populations find a feasible solution within 500 generations – something the binary encoded GA was unable to achieve. The steepness of the slope corresponding to the integer-encoded GA can have an important practical application – in situations when Genetic Algorithms have unsuccessful runs, a search can be safely restarted after much fewer generations in comparison with a binary-coded GA [12].

Table 4 below statistically proves that the difference in the mean values for the binary- and integer-coded GAs cannot be explained by random variations.

TABLE 4. *t-test of GA efficiency*

Difference in means	t-test	p-value
204.92	28.096	0.000

Secondly, we compared the best solutions found by the two GAs within the imposed limit of 500 generations. The results are presented in Table 5 in the form of a t-test comparing the best solutions found by the competing GAs.

TABLE 5. *t-test of the quality of GA solutions*

Difference in means	t-test	p-value
1.328	10.574	0.000

Again it is clear that the integer encoding yields a significant improvement over the binary equivalent; thus, we are not sacrificing the quality of final results for speed of operation (in fact, we are improving on it).

5 Discussions

Factor tuning is a traditional way of testing and comparing different values of GA factors before the real run of the algorithm. The main difficulties with factor tuning are [7]:

- mistakes in factor settings can result in sub-optimal algorithm's performance or even in unsuccessful runs;
- due to factor interaction, the tuning of these factors by trial and error is impractical for real problems;
- optimal settings for one problem are not necessarily the best for a slightly different problem in the same domain.

The developed methodology addresses these problems in situations when factor tuning is worth the effort, when the possible ranges of factors can be identified in advance, and when the response surface obtained during statistical experiments is consistent. We believe that there are many practical applications that satisfy these requirements – cancer chemotherapy optimisation is one of them.

Genetic Algorithms have already been proven to be useful in the optimisation of cancer chemotherapy treatment regimes [11, 12, 14, 15]. In this paper it has been shown that a substantial improvement of GA efficiency for the problem of cancer chemotherapy optimisation can be obtained by changing the encoding scheme from binary to integer. The improvement has been achieved both in terms of speed and the quality of final solutions.

Several interesting observations have been made over the course of this analysis; primarily these concern the difference between the factors important in fine tuning an integer-encoded GA and those of enhancing a binary-encoded GA for the cancer chemotherapy problem.

It would seem that crossover rate does not have a significant effect on an integer-encoded GA in this application. Czarn *et al* also made this observation for some of the benchmark functions in their work [4]. One reason for this may be that in addition to its recombination role, crossover also usually helps mutation to explore the search space by separating bits which are grouped together to form a single integer value. When alleles are represented by integers, the points at which crossover can occur are fixed, preventing such collaboration between mutation and crossover and reducing the role played by crossover on the whole. This may also explain why the integer encoding yields higher performance – crossover is recombining to produce good chromosomes without being able to destroy individual values within them by the described form of mutation. Incidentally, during initial construction of the integer-encoded GA it was discovered that the lack of any crossover yields very poor performance, leading to the conclusion that some crossover is beneficial.

Mutation, on the other hand, does have a significant effect on the performance. Both the magnitude and frequency of mutations play significant roles in the GA's execution speed, provided that their values strike the right balance between destructiveness (high mutation rate and large creep step) and lack of solution space coverage (low mutation rate and small creep step).

What is also interesting is that the choice of operators used for both mutation and crossover does not seem to be important for the problem under investigation, which only partly aggrees with the fundings in [13]. More would need to be done to determine the reason for the apparent similarity in performance of quite distinct operators, especially in the case of mutation.

The final point we would like to make is that even though finding a treatment schedule is not a hard real-time task, the introduction of a more complex fitness function to include more drugs, treatment intervals, dosage levels or constraints is likely to increase the evaluation cost considerably. Thus, it is important to improve on processing efficiency wherever possible and the use of the developed methodology of factor fine tuning together with integer encoding for potential GA solutions seems to be an effective approach to achieve this.

Acknowledgment

The authors would like to express their gratitude to the Carnegie Trust for the Universities of Scotland for supporting Alexander Brownlee with a studentship during the summer 2004.

Bibliography

[1] Agur, Z., Hassin, R., and Levy, S. (**In press**) *Optimizing chemotherapy scheduling using local search heuristics.* Journal of Operations Research .

[2] Baeck, T., Fogel, D., Michalewicz, Z. (Eds.) (1997) *Handbook of Evolutionary Computation*. Oxford University Press.

[3] Berthold, M. and Hand, D. J. (Eds.) (2003) *Intelligent Data Analysis: An Introduction.* 2nd Edition, Springer, ISBN 3-540-43060-1.

[4] Czarn, A., *et al.* (2004) *Statistical Exploratory Analysis of Genetic Algorithms.* IEEE Transactions on Evolutionary Computation, **8**(4), pp. 405-421.

[5] Davis, L. *et al* (1991) *Handbook of Genetic Algorithms.* Van Nostrand Reiinhold.

[6] Dhar, V. and Stein, R. (1997) *Intelligent Decision Support Methods: The Science of Knowledge Work.* Prentice Hall, New Jersey 07458, ISBN 0-13-519935-2, Chapter 5.

[7] Eiben, A and Smith, J. (2003) *Introduction to Evolutionary Computing*. Springer, ISBN 3-540-40184-9.

[8] Martin, R. and Teo, K. (1997) *Optimal Control of Drug Administration in Cancer Chemotherapy.* World Scientific, Singapore New Jersey London Hong Kong.

[9] Michalewicz, Z. (1994) *Genetic Algorithms + Data Structures = Evolution Programs.* 2nd Edition, Springer-Verlag.

[10] Mitchell, M. (1998) *An Introduction to Genetic Algorithms.* MIT Press.

[11] Petrovski, A. and McCall, J. (2001) *Multi-Objective Optimisation of Cancer Chemotherapy Using Evolutionary Algorithms.* Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization: Zurich, Switzerland. Springer: Lecture Notes in Computer Science **1993**, pp. 531-545.

[12] Petrovski, A., Wilson, A. and McCall J. (2000) *Statistical identification and optimisation of significant GA factors.* Proceedings of the 5th Joint Conference on Information Sciences (JCIS'2000): Atlantic City, USA. Volume 1, ISBN 0-9643456-9-2, pp. 1027-1030.

[13] Rojas, I., et al. (2002) *Statistical Analysis of the main parameters involved in the design of a genetic algorithm.* IEEE Transactions of Systems, Man and Cybernetics Part C: Applications and Reviews, **32**(1), pp. 31-7.

[14] Villasana, M., Ochoa, G. (2004) Heuristic Design of Cancer Chemotherapies. IEEE Transactions on Evolutionary Computation, **8**(6), pp. 513-521.

[15] Tan, K., et al. (2002) *Automating the drug scheduling of cancer chemotherapy via evolutionary computation*. Artificial Intelligence in Medicine, **25**, pp. 169-185.