Evolutionary Algorithms Designing Nanoparticle Cancer Treatments with Multiple Particle Types

Michail-Antisthenis Tsompanas[1], Larry Bull[1], Andrew Adamatzky[1] and Igor Balaz[2]
[1] University of the West of England
[2] University of Novi Sad

There is a rich history of evolutionary algorithms tackling optimization problems where the most appropriate size of solutions, namely the genome length, is unclear *a priori*. Here, we investigated the applicability of this methodology on the problem of designing a nanoparticle (NP) based drug delivery system targeting cancer tumours. Utilizing a treatment comprised of multiple types of NPs is expected to be more effective due to the higher complexity of the treatment. This paper begins by using the well-known NK model to explore the effects of fitness landscape ruggedness on the evolution of genome length and, hence, solution complexity. The size of novel sequences and variations of the methodology with and without sequence deletion are also considered. Results show that whilst landscape ruggedness can alter the dynamics of the process, it does not hinder the evolution of genome length. On the contrary, the expansion of genome lengths can be encouraged by the topology of such landscapes. These findings are then explored within the aforementioned real-world problem. Variable sized treatments with multiple NP types are studied via an agent-based open source physics-based cell simulator. We demonstrate that the simultaneous evolution of multiple types of NPs leads to more than 50% reduction in tumour size. In contrast, evolution of a single NP type leads to only 7% reduction in tumour size. We also demonstrate that the initial stages of evolution are characterized by a fast increase in solution complexity (addition of new NP types), while later phases are characterized by a slower optimization of the best NP composition. Finally, the smaller the number of NP types added per mutation step, the shorter the length of the typical solution found.

***Index Terms*—cancer simulator, nanoparticles, NK Model, PhysiCell, variable-length genome**

I. Introduction

EVOLUTIONARY algorithms (EAs) can be applied to problems of unknown complexity through the use of variable-length genomes. Here the term genome refers to the set of variables for the given optimization problem. A seminal example of the variable-length genome methodology is the work by Fogel et al. [1] on finite state machine design through the use of a mutation-based scheme that can increase or decrease the number of nodes. A subset of variable-length problems, known as metameric representation problems [2], is tackled by utilizing a segmented variable-length genome. This means that the solutions are defined as sets of similar components. Examples of these problems include the layout of wind farms, wireless sensor networks, and composite laminate stacking problems [3]. For example, the optimization of the placement of wind turbines on a predefined site with a specific wind profile can enhance the overall efficiency of the wind farm by limiting the turbine interactions [4], [5]. Since there is no given number of turbines in the problem, a variable-length representation can be utilized. Similarly, in coverage problems, such as the planning of cellular systems [6] or of wireless sensor networks [7], [8], the placement of an unknown number of nodes to achieve the coverage of a sector is required. Despite the fact that the amount of nodes is not predefined, the optimization process has to take into account minimizing costs whilst maximizing the reliability and coverage of the instalment. Variable-length algorithms proved to be more efficient than the fixed-length ones in some cases, even when the optimal amount of components of the solution was known [2]. Other examples of variable-length representations include aspects within the field of electrical circuit design, such as designing passive filters [9], transistor amplifiers [10] and computing circuits [11], and the field of neuroevolution [12]–[15], where artificial neural network weights and topology are often optimized concurrently (after [16]).

This paper first expands on previous variable-length studies by investigating how the ruggedness of a fitness landscape influences genome length during evolution, with an abstract tunable model and a purely mutation-based approach. Here the term fitness refers to the value of the objective function in the given optimization problem. The insights gained are then applied within the bio-engineering domain. In particular, we simulate a nanoparticle (NP) based cancer treatment, where multiple different types of NPs can be included. Given the high computational cost of running the cancer simulator, the conclusions drawn from the abstract model are used as guidelines to economize computational resources.

There are various procedures in nature that can produce novel DNA sequences and hence vary genome lengths, such as horizontal gene transfers, recombination events, whole genome duplications, retrotransposons, and others. A novel sequence can have no immediate function and thus can be subject to genetic drift or can be under positive or negative selective pressure, due to beneficial or detrimental effects of mutation [17], gene dosage effects [18], the subsequent specialisation of a duplicated function [19], etc. In this paper, following [1], a simple procedure to vary genome length is implemented, where the length of an existing genome is increased with a chosen number of random genes. By doing so, the fitness of the altered individual will be instantly modified as random

Manuscript received xxxxxxx; revised xxxxx. Corresponding author: M.A. Tsompanas (email: Antisthenis.Tsompanas@uwe.ac.uk).

contributions are included. To study these phenomena, a well-known abstract NK model [20] is utilized. The NK model allows dynamic alterations in the size of fitness landscapes of specified ruggedness. Our results suggest that landscape ruggedness, the length of the new sequence added, and the presence of gene deletion can all affect the evolution of genome length.

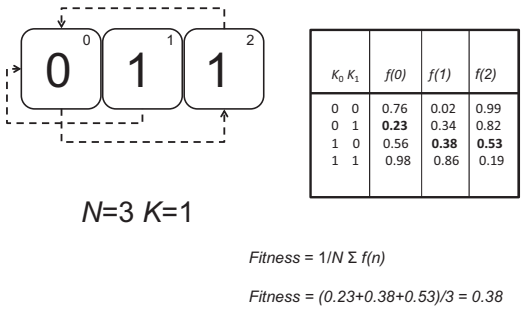
We used these findings to inform investigations of optimizing the design of a simulated, NP-based, targeted drug delivery system for cancer treatment. This simulation is based on PhysiCell [21], an open source physics-based cell simulator, which extends BioFVM [22], a large-scale transport solver. It should be noted that, given the high complexity of the simulator and its long execution times, the evolutionary optimization should ideally be kept under a computational budget. PhysiCell source code was altered to simulate the injection of multiple types of NPs with different behaviours within the same treatment. Since the appropriate NP properties for a given type of tumour are unknown, optimization can be applied as a search through a space of variable size. Therefore, to optimize both the number of types of NPs and their different features, we use a variable-length evolutionary algorithm.

The optimization problem of designing robust anti-cancer treatments can be expressed as:

$$\begin{aligned} \underset{\mathbf{X}}{\text{minimize}}\ f(\mathbf{X}) \quad & \text{where } \mathbf{X} = [x_{(1,1)}, x_{(2,1)}, ..., \\ & \quad x_{(4,j)}, x_{(5,j)}],\ j = [1, ..., 10] \\ \text{subject to} \quad & 0 \leq x_{(1,j)} \leq 1,\ j = [1, ..., 10] \\ & 0 \leq x_{(2,j)} \leq 1,\ j = [1, ..., 10] \\ & 0 \leq x_{(3,j)} \leq 10,\ j = [1, ..., 10] \\ & 0 \leq x_{(4,j)} \leq 10,\ j = [1, ..., 10] \\ & 0 \leq x_{(5,j)} \leq 10,\ j = [1, ..., 10] \end{aligned}$$

where f is the objective function or fitness function that is defined as the size of the tumour after the application of a treatment (i.e., the number of simulated cancer cells at the end of a PhysiCell run) and needs to be minimized. $\mathbf{X}$ represents the design variable vector (i.e., the genome) which is composed of sets of five variables for each type of NPs in the solution. Note that variable j in the above equation is the amount of different types of NPs used simultaneously in a treatment and can take values from 1 to 10, so the genome has variable length. Moreover, the bounds are presented for the following variables: the attached worker migration bias ($x_{(1,j)}$), unattached worker migration bias ($x_{(2,j)}$), worker relative adhesion ($x_{(3,j)}$), worker relative repulsion ($x_{(4,j)}$), worker motility persistence time (min) ($x_{(5,j)}$). For further details of the simulation refer to Section IV and for the parameters of the implemented standard steady-state EA refer to Section V.

Our findings indicate that more complex treatments (i.e., with more than one NP type) are more effective than those with a single NP type. Significantly, the evolved treatment complexity does not reach the maximum available complexity. Therefore, bloat phenomena, although present to a certain extent, are not prevailing in this application. Following findings in the NK model, the average treatment complexity varies slightly with the increase in solution length size. The smaller the number of NP types added per mutation step, the shorter the length of the typical solution found. Since there is no significant difference in the final fitnesses of the scenarios explored, for practical reasons (i.e., ease of manufacturing, the problem of interactions between drugs and lower toxicity), the solutions of smaller complexity can be considered as preferable *in vivo*.



K_0 K_1	f(0)	f(1)	f(2)
0 0	0.76	0.02	0.99
0 1	**0.23**	0.34	0.82
1 0	0.56	**0.38**	**0.53**
1 1	0.98	0.86	0.19

Fig. 1. An example NK model with parameters $N = 3$ and $K = 1$, where arrows indicate each arbitrarily chosen dependent gene (left). The table with $2^{(K+1)}$ rows containing the fitness contributions of each gene is shown (right), along with the fitness calculation for the example genome.

The paper is arranged as follows: the next two sections present the NK model and the behaviour of a simple NK mutation-based approach to variable-length optimization. The following two sections introduce the bio-engineering problem and the results of the optimization methodology applied to it, respectively. In the final section conclusions are drawn.

II. The NK Model

The NK model was introduced to investigate the characteristics of rugged fitness landscapes [20]. The basic model contains two parameters: N, the length of the binary genome; and K, the number of genes in a genome —the position of which are typically chosen at random— that influence the fitness contribution of each gene. Increasing K increases the ruggedness of the landscape, leading to an increase in the number of local optima and a decrease in their typical height [23]. Since the model assumes a high complexity of the gene interdependence effect, the only appropriate tactic is to assign random values to their contribution to fitness. As depicted in the example (Fig. 1), a table is defined with $2^{(K+1)}$ fitness values randomly chosen between 0 and 1, where there is one fitness for each combination of traits and the fitness contribution of each gene is found in the table. The total fitness of the genome is a sum of all separate fitness values of each gene, normalized by N.

To examine the evolutionary dynamics and properties of the NK model, we use a mutation-based hill-climbing algorithm (in accordance with [23]). There, the single point in the fitness space represents a converged species, i.e., the size of the population is one and it evolves by randomly changing one randomly chosen gene per generation. If the fitness of the mutated "population" increases, it keeps the new genetic configuration. If the fitness is equal, the selection is made at

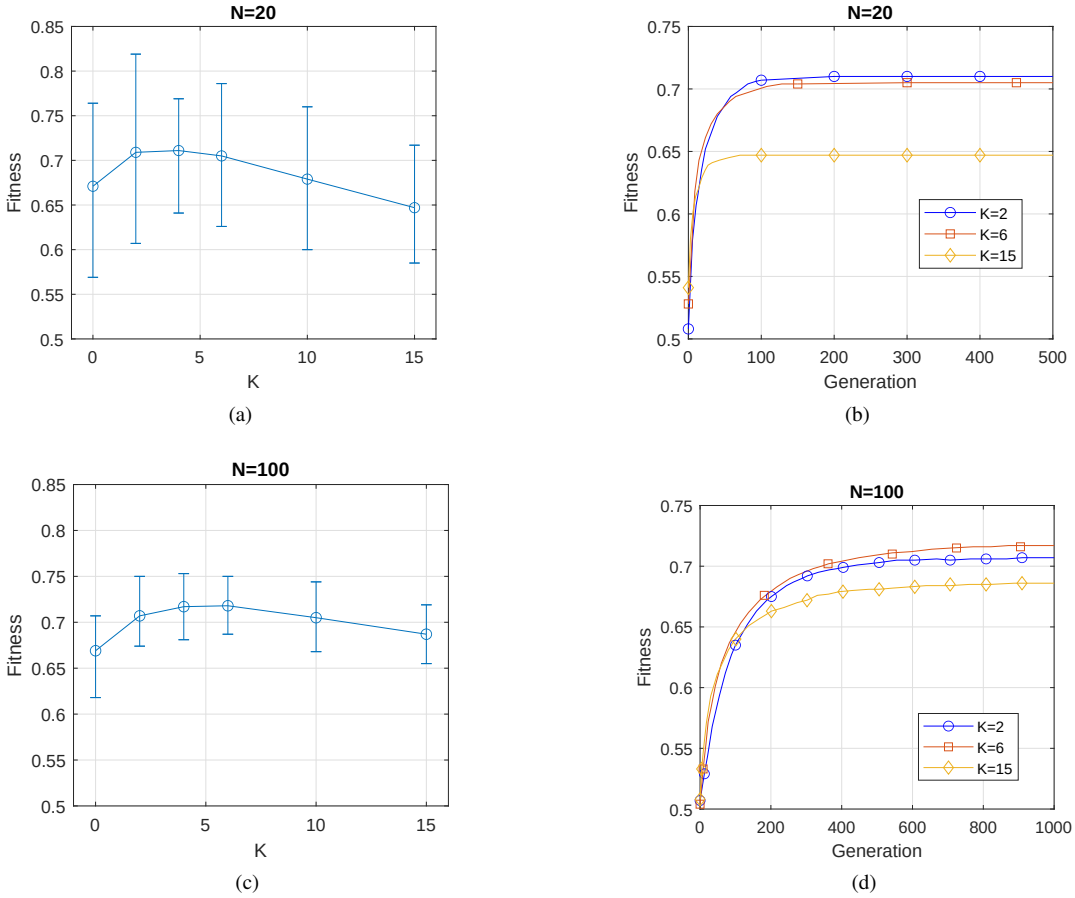


Fig. 2. (a, c) Average fitness values at 20000 generation steps for different K and N values (maximum and minimum values are indicated by vertical bars). (b, d) Example average fitness values through the initial steps of evolution.

random. All results presented in this paper, acquired with the NK model, are averaged over 10 runs (random start points) on each of 10 randomly produced NK functions for every one combination of the given parameters (N and K), i.e., 100 runs, for 20,000 generations. Here six values are chosen from $0 \leq K \leq 15$, for N=20 and N=100.

Some results of running the standard NK model with different parameters to demonstrate the characteristic behaviour of evolution depending upon the ruggedness of the fitness landscape are presented in Fig. 2. When $K = 0$ all gene fitness contributions are independent and randomly chosen within a range of [0,1]. As a result, based on order statistics, the average fitness is equal to 0.66. For small values of K (up to $K = 8$), regardless of N, the ruggedness of the landscape increases, as does the height of the better optima found. For values of K higher than 8, the increasing entanglement of gene dependencies that define their fitness contributions causes an increasing number of low fitness local optima. Note that for high K values relative to N, the central limit theorem indicates that optima will average around 0.5. The decrease in the finally found fitness when K tends towards N is an incident known as the "complexity catastrophe". The fittest individuals found for tests with $K > 6$ and $N = 20$ (as depicted in Fig. 2) are statistically significantly lower from those with $N = 100$, when tested under T-test ($p < 0.05$).

III. Genome Growth in the NK Model

In order to investigate the behaviour of variable-length genomes throughout evolution under the abstract NK model, the mutation operator is extended. In addition to varying an arbitrarily picked gene, the operator can now also add a random number (G) of new arbitrarily generated genes to the rightmost part of the available genome ($N' = N + G$) (for more details refer to the Supplementary Material). Since it is generally assumed that this new functionality assigns new dependencies on the fitness contributions, the first connection of a randomly determined gene of the pre-existing ones is assigned to the newly added ones (for $K \geq 1$). In addition, the newly added genes obtain K dependencies throughout the entire genome for their fitness contribution. Thus, the extension to the genome has a two-way influence on the fitness of an individual.

As an initial trial, we investigate the case of $G = 1$, shown in Fig. 3. In the same fashion as before, the parameter N is set to two values, $N = 20$ and $N = 100$. In the first case, with the initial parameter $N = 20$ and for values of $K > 10$, the resulting fitness (Fig. 3(a)) is enhanced compared with the static genome length (Fig. 2), with the difference statistically significant (T-test, $p < 0.05$). Here, the genome length increases by approximately three genes (or up to $N' = 23$ as depicted in Fig. 3(b)). As a result, the occurrence of the

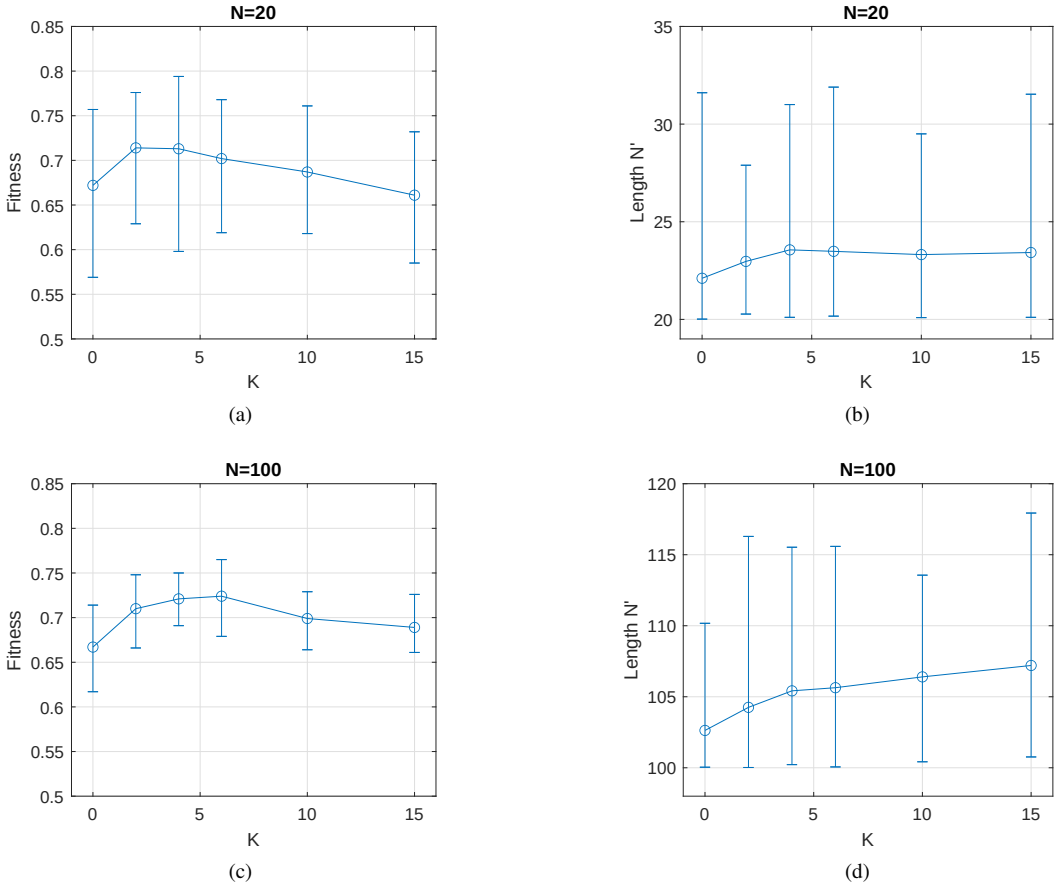


Fig. 3. (a, c) Average fitness values at 20000 generation steps for different K and N values (maximum and minimum values are indicated by vertical bars). (b, d) Average genome length values (maximum and minimum values are indicated by vertical bars) for increasing functionality by one gene ($G = 1$).

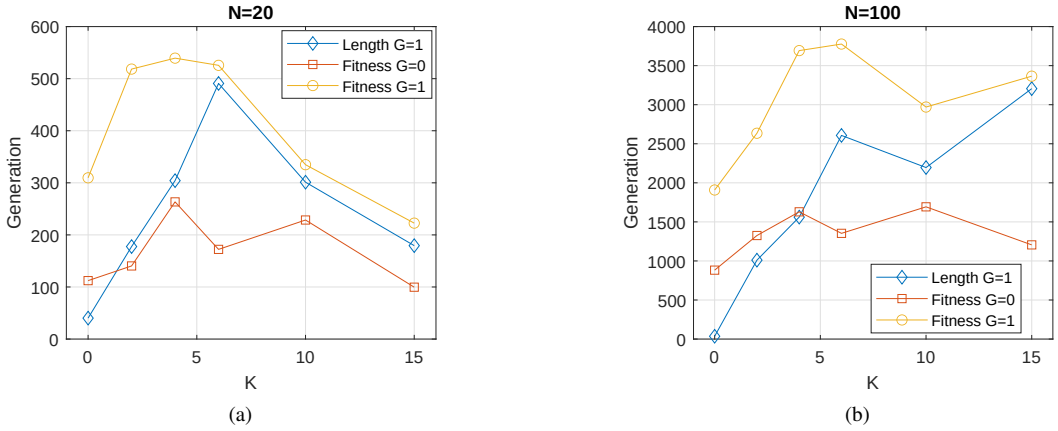


Fig. 4. Average time when the fittest individual is discovered and when the corresponding genome length is reached for combinations of N and K, and the case of $G = 1$

complexity catastrophe is no longer evident, at least for the K values explored. The final achieved genome lengths depicted in Fig. 3(b) outline a high discrepancy between maximum and minimum values; however, for lower values of K (i.e., $K < 4$), the final genome lengths are marginally smaller. Whereas, in all cases with $N = 100$, the final fitness (Fig. 3(c)) is not statistically significantly different (T-test, $p \geq 0.05$) from the standard case without genome growth. Nonetheless, the amount of added genes (Fig. 3(d)) is substantial, even double the amount that is observed with $N = 20$ (Fig. 3(b)) and $K > 4$.

As the ruggedness of the fitness landscape increases, the number of local optima increases and their typical height decreases. Hence, the increase of ruggedness provides a larger window for arbitrarily generated and added genes to make a positive fitness contribution. On the contrary, on landscapes

with lower ruggedness, high fitness optima can be expected to be located during the early stages of the search, reducing the window of opportunity for added genes to make a positive fitness contribution. This is depicted in Fig. 4, which shows the generations (y-axis) needed for the search methodology to converge to a peak for different K parameters, with $G = 0$ (no genome growth) and $G = 1$ (genome growth with one gene per step). The generation at which the genome growth stops is also indicated. It can be observed that for every K studied, the evolutionary search continues for a longer period when growth is included. This is expected due to the increase in the dimensions of the fitness landscape. It is also clear that genome growth ceases earlier for low K. Figure 5 illustrates the generation at which the additions of the first three genes to the genome are initially successful. It can be seen that the first two genes are accepted at roughly the same rate for all K and N values, whereas the acceptance of the third gene varies widely for N=20.

The effects of increasing the number of randomly added genes per growth event are also explored. Figure 6 shows results for G=20, where with $N = 20$ and $K > 4$, the final fitness is significantly enhanced compared to $G = 1$ (T-test, $p < 0.05$). This is attributed to the fact that the complexity catastrophe is more robustly avoided, as the final genome lengths are notably greater than with $G = 1$. In the case of $N = 100$, the final fitness achieved is not significantly different from the results with $G = 1$, but the final lengths are also greater.

The generations when the final fitnesses are reached and the corresponding genome lengths are illustrated in Fig. 7. It can be seen that for all K, the evolution continues for a longer period in comparison to no or limited genome growth, particularly with $N = 20$ (as in Fig. 4). The generation numbers when new genes are acquired by the genome are shown in Fig. 8. The addition of the first sets of genes occurs in comparable time steps as for the case of $G = 1$. While the second sets are also added in similar generation numbers for $N = 100$, they take longer or are not even added for $N = 20$. Consequently, adding longer gene intervals may result in more additions performed later during the evolution. That is, a larger amount of growth per addition can maintain the conditions longer for more subsequent growth. Here each increase in the size of a fitness landscape supplies a number of sub-optimal gene values, thereby maintaining an overall low fitness level, which in turn may aid the acceptance of a new random sequence.

In [24] the NK model was used to show that the evolutionary optimization is more robust when additions to a genome are smaller (for $N = 16$ and $K = 2$). This effect happens because the gradual increase of dimensionality leads to continuous fitness landscape changes, without abrupt shifts. That is, a point in the landscape where the fitness is high under smaller dimensions should lead to an appropriate new, higher fitness score when dimensions increase. A similar idea was outlined in [20] under the static genome sizes of the standard NK model when altering the size of mutation steps. When the mutation operator can generate genomes greater than the correlation length in the fitness landscape, the period needed to reach an optima increases considerably.

In our work, the results demonstrate that the first instance of the lengthy gene additions is achieved within the range of K values studied, namely without limitations based on the correlation of the fitness landscape. However, as depicted in Fig. 8, for $N = 20$ a second addition instance is not acquired for $4 < K < 10$ and a possible third one does not typically happen for all K values. This is in contrast to the case with small addition intervals ($G = 1$) as depicted in Fig. 5. This could be caused by the level of correlation between the two landscapes, despite the fact that additions are successful for low values of K.

In nature, the most frequent end result of new gene additions is their dismissal. This may happen through different mechanisms such as fractionation following duplication of the genome or via mutations. We implement the possibility of gene deletion by updating the aforementioned mutation operator. An individual can undergo mutation by altering a gene allele in its existing genome or by altering the length of its existing genome, both with the same likelihood. When length alteration is randomly selected, the length can be increased or decreased with a 50-50 chance, by G genes. That is, there is a 0.5 probability of gene allele alternation, a 0.25 probability of length increase by G and a 0.25 probability of length decrease by G (for the pseudo-code refer to Supplementary Material). After running appropriate tests as before, the fitness is not significantly altered for any combination of K and N parameters, so the results are not depicted here. On the contrary, as illustrated in Fig. 9 for $G = 1$ the final lengths of the genome observed are slightly shorter than in previous cases, especially for $K > 0$. Similar results are observed for $G = 20$ (not shown).

Further investigation of the final genome length achieved with respect to the reduction and increase functions relative probabilities is performed. The results are provided in Fig. 10. The case for $G = 1$ is investigated with probabilities of the genome reduction function in the range of 0 to 0.5 (and the increase with complementary probabilities). It can be concluded that as the probability of reduction increases, the final genome length decreases at an almost steady rate. However, the decrease of the final genome length is not that significant, i.e., for $N = 20$ it is roughly one gene, whereas for $N = 100$ it is two genes for each case of K.

The findings described in this section are now used to inform explorations of novel types of NP-based cancer treatment, where the number of NP types is not pre-determined. As there is no trivial way of accurately determining the ruggedness of the real problem fitness landscape (the corresponding N and K combination of the NK model), the increasing interval of genome length will be studied for more gradual genome growth (as with the case of $G = 1$). This is due to the fact that $G = 1$ case provides a good starting point to study the observation made earlier; as G increases, the resultant new fitness landscape bears less analogies to the previous fitness landscape.

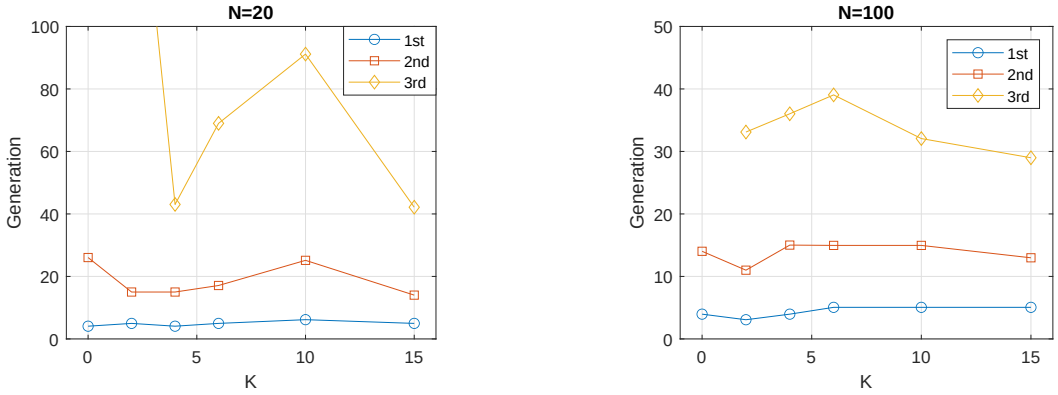


Fig. 5. Average waiting time in generation steps for each interval increase for combinations of N and K, and the case of $G = 1$. Note that for $N = 20$ and $K = 2$ the third gene addition averages at 179, while it was not added for $K = 0$ and both N values.

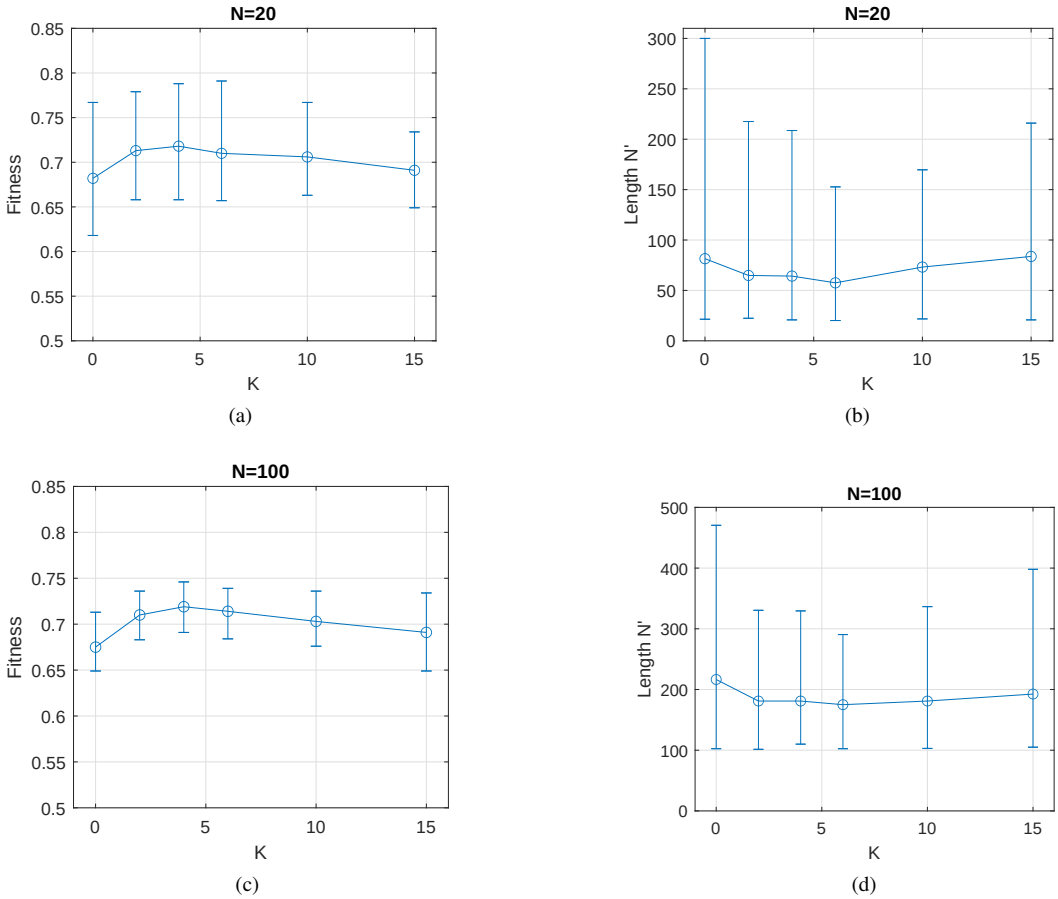


Fig. 6. (a, c) Average fitness values at 20000 generation steps for different K and N values (maximum and minimum values are indicated by vertical bars). (b, d) Average genome length values (maximum and minimum values are indicated by vertical bars) for increasing functionality by twenty genes ($G = 20$).

IV. PhysiCell

The high computational capacity of modern systems enables the incorporation of detailed mathematical models in the study of biological processes, along with laboratory experiments. More specifically, the study of cancer biology is the epicenter of many mathematical models [25], [26] that aim to be powerful tools in the hands of scientists. One of these models is PhysiCell [21], which is an agent-based, multicellular, open source simulator emulating physics and biological rules in interactions between cells. Moreover, PhysiCell utilizes BioFVM [22] to emulate the secretion, diffusion and uptake of chemical substances in the simulated area.

Employing PhysiCell as a target simulator for optimization was previously proposed in problems of designing NP-

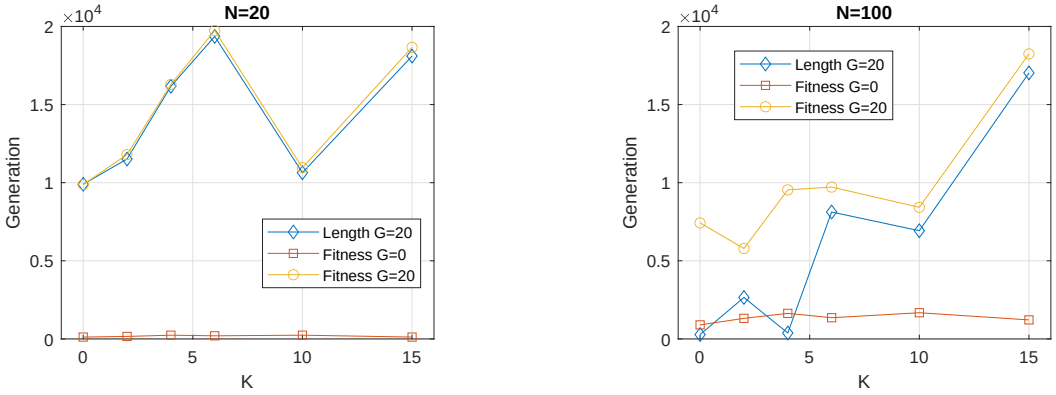


Fig. 7. Average time when the fittest individual is discovered and when the corresponding genome length is reached for combinations of N and K, and the case of $G = 20$

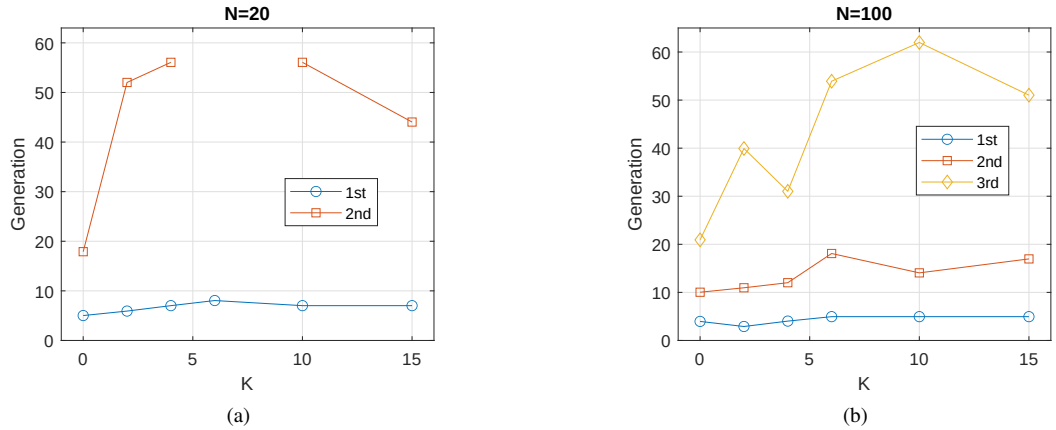


Fig. 8. Average waiting time in generation steps for each interval increase for combinations of N and K, and the case of $G = 20$.

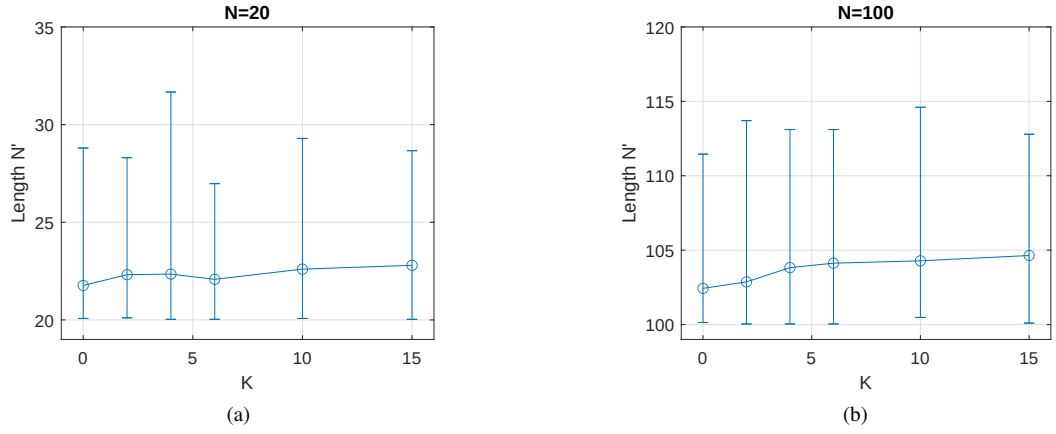
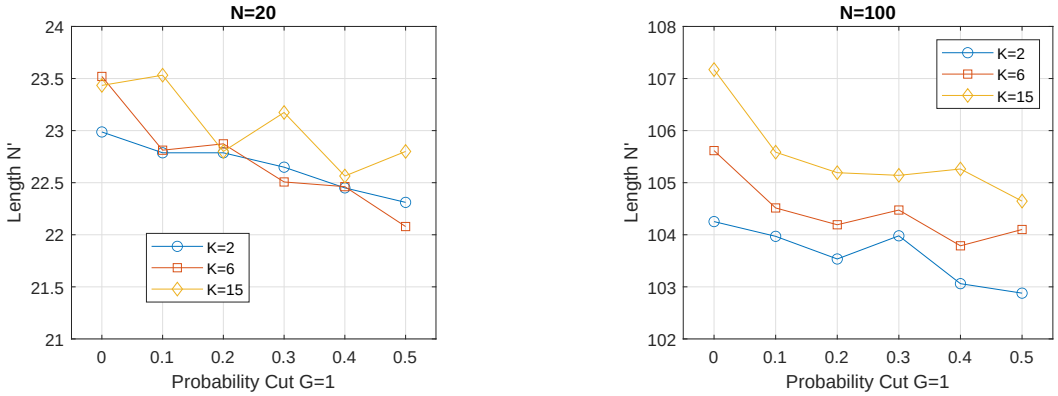


Fig. 9. Average genome length values (maximum and minimum values are indicated by vertical bars) for increasing and decreasing functionality by one gene ($G = 1$).

based drug delivery systems [27]–[30] or unveiling cancer immunotherapies [31]. The design of NPs was investigated through surrogate-assisted [27], haploid-diploid [28], differential evolution [29] and novelty search [30] evolutionary algorithms, while the most effective immunotherapy for cancer tumours was examined through a combination of active learning and EAs [31].

Specifically, the sample project "anti-cancer biorobots" (see [21] for details) of the PhysiCell simulator (v.1.6.1) is used to simulate the NP's interaction with the tumour, as in previous works [30], [32], [33]. However, here the need to simulate complex multi-NP-based treatments leads to an alternation in



(a) (b)

Fig. 10. Average final lengths for combinations of N and K values, and decrease functionality probability.

TABLE I
UNALTERED PARAMETERS OF PHYSICELL SIMULATOR.

Parameter	Value
Damage rate	0.03333 min^{-1}
Repair rate	0.004167 min^{-1}
Drug death rate	0.004167 min^{-1}
Elastic coefficient	0.05 min^{-1}
Cargo O_2 relative uptake	0.1 min^{-1}
Cargo apoptosis rate	4.065e-5 min^{-1}
Cargo relative adhesion	0
Cargo relative repulsion	5
Cargo release O_2 threshold	10 $mmHg$
Maximum relative cell adhesion distance	1.25
Maximum elastic displacement	50 μm
Maximum attachment distance	18 μm
Minimum attachment distance	14 μm
Motility shutdown detection threshold	0.001
Attachment receptor threshold	0.1
Worker migration speed	2 $\mu m/min$
Worker apoptosis rate	0 min^{-1}
Worker O_2 relative uptake	0.1 min^{-1}

the source code (Please refer to the Supplementary Material for details). Nonetheless, similar to the aforementioned works, here the design of a NP can reflect to a point in a 5-dimensional space of the following parameters (with their range in the brackets): attached worker migration bias [0,1], unattached worker migration bias [0,1], worker relative adhesion [0,10], worker relative repulsion [0,10], worker motility persistence time (min) [0,10]. The rest of the user-defined parameters are not altered from the original instance published by the developers of the simulator and are presented in Table I.

The scenario in the sample project "anti-cancer biorobots" of the agent-based PhysiCell simulator is the following (for more extensive details refer to [21]). An initial tumour with a total radius of 200 μm of cancer cells (approximately 570 cancer cell agents) undergoes growth of a simulated period of 7 days. At that point, the treatment is injected into the simulated area, comprising of 450 cargo agents and 50 worker agents, emulating the therapeutic compound and NPs, respectively. A simulated period of 3 days is then executed, where the worker agents transport the cargo agents (therapeutic compound) and deposit them near the cancer cell agents that decay and die because of the increased concentration of drug agents. The fitness function of the given point in the parameter space is the number of cancer cell agents remaining after the simulated period of 10 days.

Every simulation of these 10 days runs on an Intel® Xeon® CPU E5-2650 at 2.20GHz with 64GB RAM (using 8 of the 48 cores) and is completed at approximately 5 minutes of wall-clock time. Because of the stochastic nature of the simulator, a static sampling approach is employed, considering the mean of 5 runs with the same parameters. In order to further minimize the noise caused by the stochasticity of the simulator and, also, to speed up the optimization process, alternations in the original source code are made in order to load a single tumour at the initialization of the simulator and apply the therapy for 3 days (as described in [28]).

One specific tumour, which was derived after a simulated growth of 7 days of an initial tumour, is used as the new initial state and the treatment is injected to the simulated area at $t = 0$. Then, 3 days are simulated and the evaluation of a single solution needs 1.5 minutes of wall-clock time (7.5 minutes for the 5 runs for static sampling). Each test in the following starts with the same randomly produced population of solutions and evolution is applied for 200 evaluations of each individual/solution. Thus, the results are provided for each test after approximately a day wall-clock time of repeatedly running the simulator.

To accommodate the functionality of injecting the region around the cancer tumour with more than one types of NPs with different parameters, we made appropriate alternations in the source code of the simulator. Without loss of generality a maximum of 10 different types of NPs is defined. The amount of NPs (worker agents) is set to 50, despite the number of types of NPs. These 50 NPs are equally divided among different NP types. For instance, in testing 2 NP types, 50 worker agents are divided into two groups of 25; with 5 types of NPs, the size of each group is ten, etc. Further technical details of the additions/alternations of the initial source code can be found in the Supplementary Material.

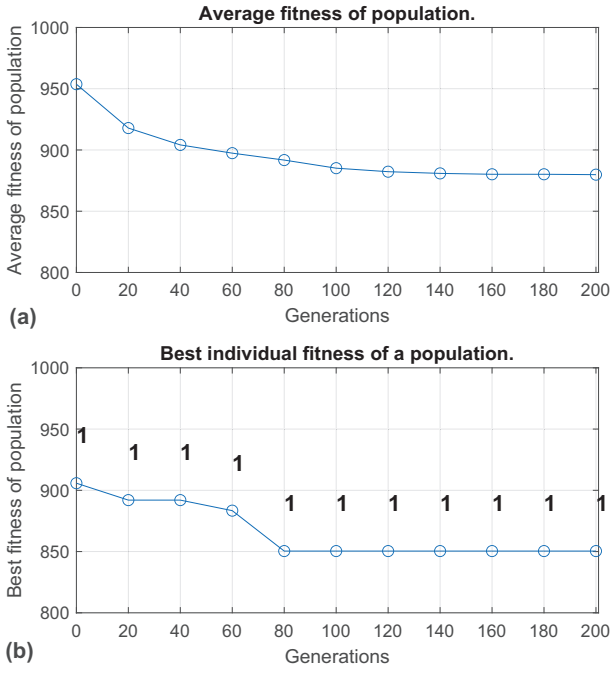


Fig. 11. Results from example run of the EA. (a) Evolution of average fitness of the population and (b) evolution of the best individual in the population (where numbers indicate the number of types NPs).

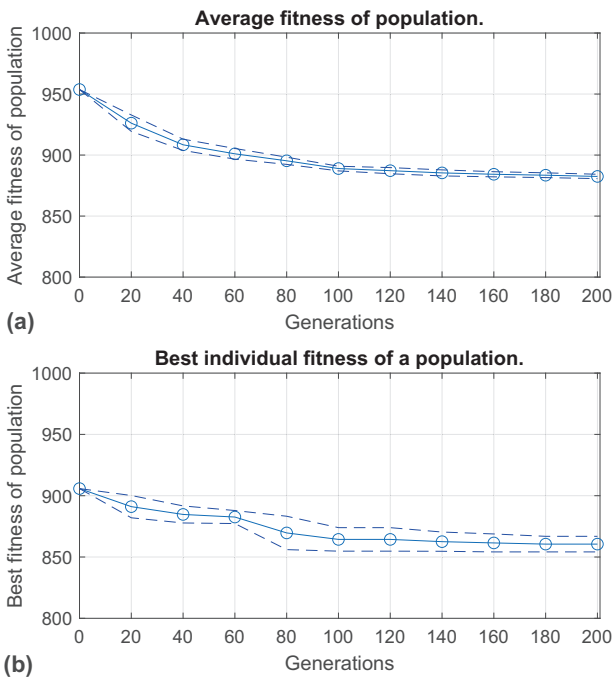


Fig. 12. Average and confidence levels (95%) results from 5 runs of EA. (a) Evolution of average fitness of the population and (b) evolution of the best individual in the population.

V. Genome growth in the simulator

The optimization of the design of NPs for drug delivery initially uses a standard steady-state EA and mutation operators only for a single NP. The population size is $P = 20$, reproduction selection is implemented as a tournament of size $T = 2$, and replacement selection is the inverse. Offspring individuals replace the selected individual only if that individual is less fit, namely it results in a higher amount of remaining cancer cell agents. The mutation procedure is executed on one randomly selected gene by modifying it with a random step size of $s = [-5; 5]\%$. The computational budget is set as 1000 evaluations with PhysiCell, thus, given the 5 run static sampling approach, the population evolves for 200 generations.

The results of the evolution with the EA are depicted in Figs. 11 (example run) and 12 (average of 5 runs). Figure 11(a) shows that the average fitness of the population (remaining cancer cells agents) converges to c. 880 agents after 200 generations of the evolution of the population, from c. 950 agents of the initial random population. That is an improvement of c. 7.3%. Quite similar results are provided by all the tests executed, as illustrated in Fig. 12(a) showing the average and 95% confidence levels of the cumulative results. Note here that the confidence levels of 95% are calculated with the mean and standard deviation of the sample and indicate the probability (here 95%) with which the estimated interval contains the true value of the parameter. For the best individual in the population the final fitness seems to converge close to 850 agents, while the one initially randomly generated has a fitness of c. 900, an improvement of 5.5%, for one example run (Fig. 11(b)). The cumulative results depicted in Fig. 12(b) reveal that during the rest of the runs the improvement is slightly smaller.

In order to enable the variation of genome length —and hence more than one types of NPs per solution— the mutation operator described in Section III is incorporated. The mutation can therefore alter a randomly chosen gene allele or add one type of NPs (with randomly chosen parameters) to the simulated treatment. Both cases have the same possibility of occurring, which is 50%. A maximum of 10 NPs types is allowed here. As explained before, despite the amount of types of NPs being tested per treatment, a total of 50 worker agents are injected in the simulated treatment. The initial population used for all tests is the same as for the previous tests (only gene allele mutation) and consisted of solutions with only one type of NPs. Note that the existence of an additional type (or multiple types) of NPs potentially alters the fitness landscape significantly due to the complex interactions with the cancer cell agents.

With the ability to add multiple types of NPs, the optimization process reaches better results within the 200 generations as depicted in Figs. 13 (example run) and 14 (average of 10 runs). In Fig. 13(a), the average fitness of the population is depicted for an example run, which converges to c. 450 agents at the end of the evolution, while the same metric of the initial random population is c. 950 agents. Thus, the alternation in the number of types of NPs injected leads to an improvement of c. 52.6%. Investigating the cumulative results of average and 95% confidence levels of all the runs that are portrayed in Fig. 14(a), there is evidently a further improvement of the average fitness of the population (average of 431 agents). Moreover, there is an improvement of 52.2% when comparing

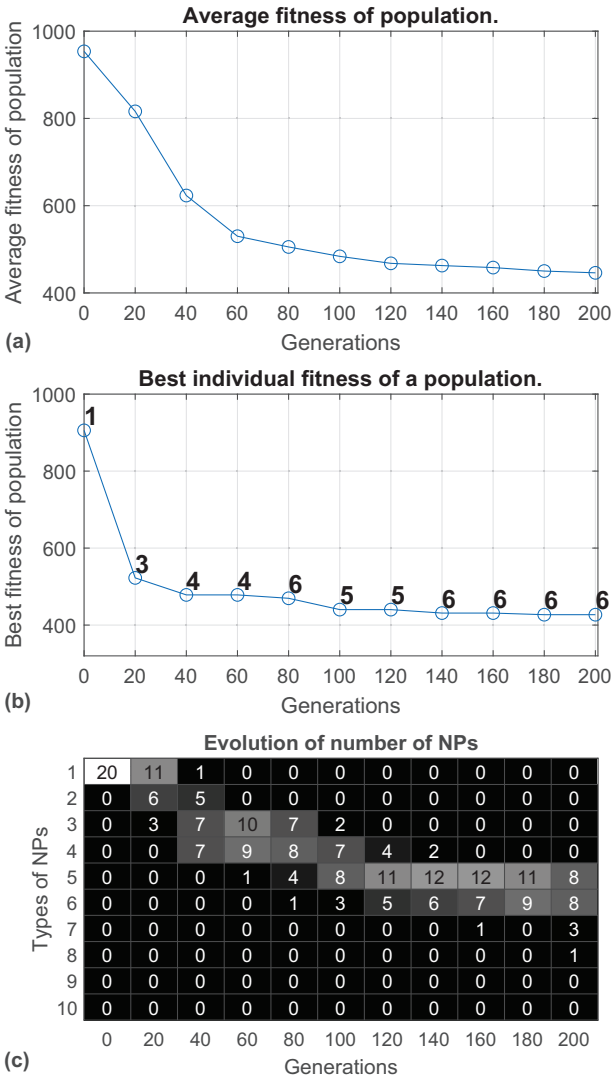


Fig. 13. Results from example run of the EA with variable-length genome. (a) Evolution of average fitness of the population, (b) evolution of the best individual in the population (where numbers indicate the number of types NPs) and (c) composition of population in terms of types of NPs.

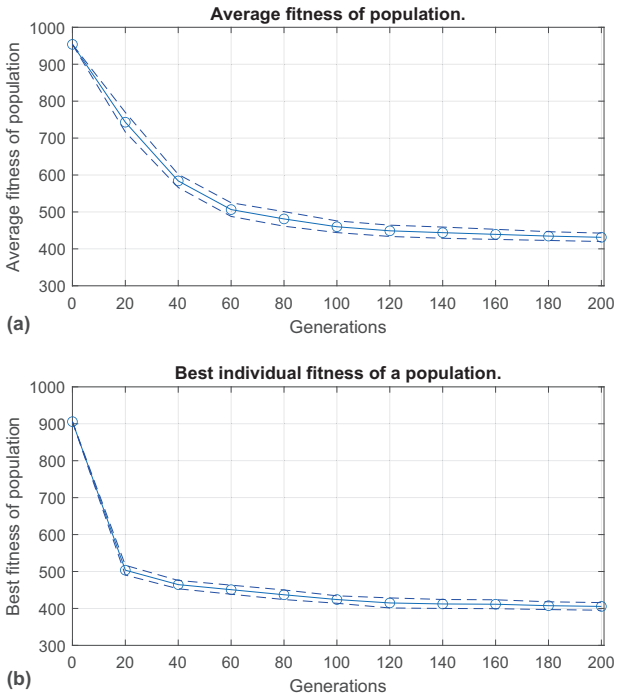


Fig. 14. Average and confidence levels (95%) results from 10 runs of EA with variable-length genome. (a) Evolution of average fitness of the population, (b) evolution of the best individual in the population.

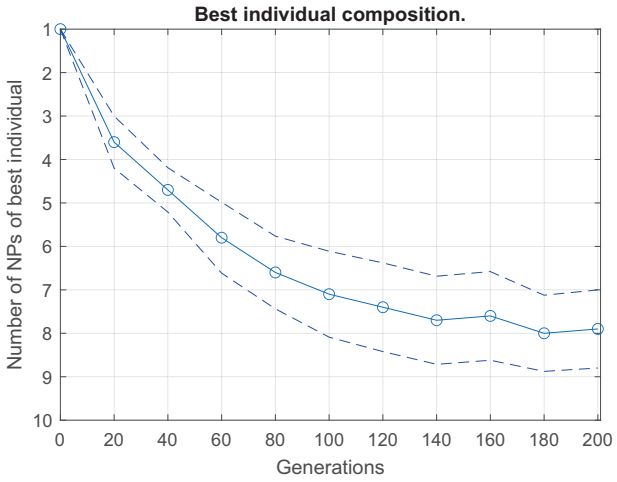


Fig. 15. Average and confidence levels (95%) results from 10 runs of EA with variable-length genome for the composition of the best solution.

the final fitness of the best individual discovered during one run (approximately 430 agents) with the best individual randomly generated in the initial generation (c. 900 identical in all runs), as outlined in Fig. 13(b). In Fig. 14(b) where all the results of 10 runs are considered, it can be seen that during the rest of the runs the improvement is similar and slightly better (average of 405 agents).

Despite the fact that the maximum number of NPs types can be 10, the composition of the best solutions usually converges to lower complexity. Taking into consideration the average of all 10 tests, solutions converge to 8 types (as illustrated in Fig. 15). Following behaviours observed in the NK model, as shown in Fig. 13(b), evolution typically adds genes (types of NPs) quite fast, while later slowly optimizing the best composition of NPs types. Similar behaviour is observed in all instances (Fig. 15).

To explore the effect of the added sequence (G) size variation, in the next experiments we allow mutation to add two new NPs types. This process reaches similar fitness results as the addition of one type of NPs per mutation event, as observed in Figs. 16 (example run) and 17 (average of 10 runs). The average fitness of the population for the example run, portrayed in Fig. 16(a), converges to c. 445 agents. That implies an improvement of c. 53.1%, compared with the same metric of the initial random population c. 950 agents. When considering the cumulative results of 10 runs, outlined in Fig. 17(a), it is evident that a further improvement of the average fitness of the population occurs (average of 434 agents). Also, the final fitness of the best individual produced during the

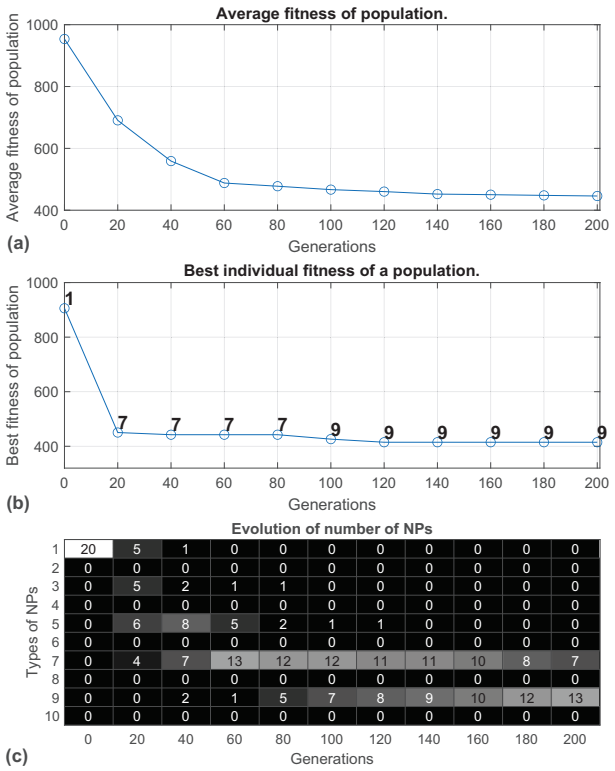


Fig. 16. Results from example run of the EA with variable-length genome adding two types of NPs. (a) Evolution of average fitness of the population, (b) evolution of the best individual in the population (where numbers indicate the number of types NPs) and (c) composition of population in terms of types of NPs.

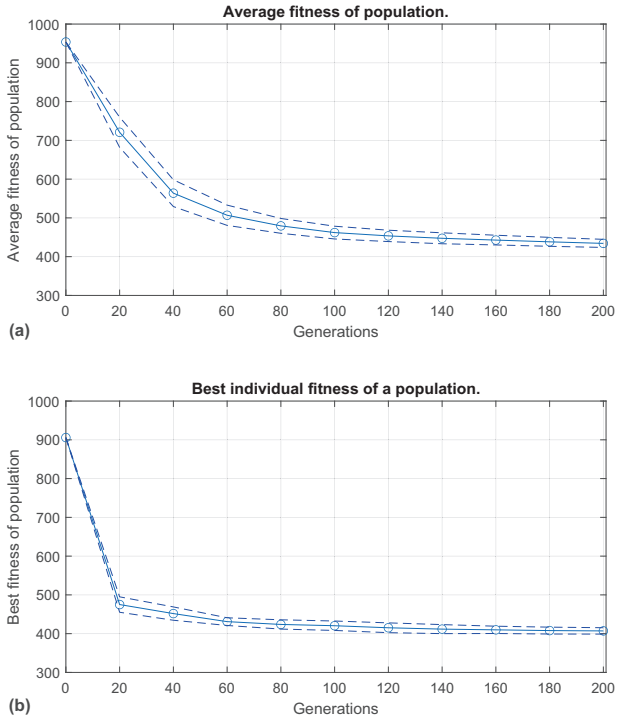


Fig. 17. Average and confidence levels (95%) results from 10 runs of the EA with variable-length genome adding two types of NPs. (a) Evolution of average fitness of the population, (b) evolution of the best individual in the population.

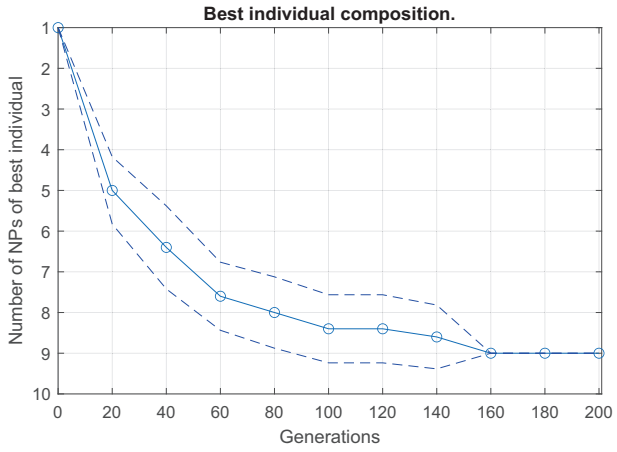


Fig. 18. Average and confidence levels (95%) results from 10 runs of the EA with variable-length genome adding two types of NPs for the composition of the best solution.

example run (approximately 415 agents) compared with the randomly generated initial one (c. 900 agents) is improved by 53.9%, as given in Fig. 16(b). In Fig. 17(b) where all the results of 10 runs are considered, it can be established that during the rest of the runs the improvement is similar and slightly better (average of 407 agents).

Here, in contrast to adding one type of NPs per mutation event, the best solutions converge to the maximum available composition (9 types), as illustrated in Fig. 18. In some of the runs the highest amount of NPs types is reached by the 100th generation (earlier than the previous case, i.e., the 120th generation), while in all runs the composition of best solutions converges to 9 by the 160th generation. As in some cases in the NK model, whilst fitness is not significantly improved by the addition of more genes per growth event, longer genomes emerge. This is an indication of the bloat effect, but one that is controllable due to the parameter G.

We would like to emphasize that our investigation of the NK model (Section III) showed that possibility of decreasing the genome length does not lead to marked difference in the evolved genome lengths and the final fitness. Therefore, this option is not tested in the computationally expensive cancer simulator.

VI. Discussion

The best solution found in this study has a fitness of approximately 405 cancer agents remaining after the three days of simulated treatment. Note that the fittest individual of the initial random population was evaluated at 900 agents, translating to an improvement of 55%. To further comprehend the fitness level of this solution, a comparison is drawn with previous studies utilizing the same simulator and similar fitness functions, but with different approaches. In [28] the best solution found indicated a 4.8% fitness improvement (best fitness of the initial random population was 420 agents, while the final best fitness was approximately 400 agents), in [29] an improvement of 13.3% (c. 450-390) and in [30] an improvement of 16.9% (c. 415-345). As it can be seen,

the best fitness (in terms of percentage reduction of tumour size) in this study largely outperforms all previous ones. However, if the fitness is expressed as the absolute number of remaining cells, the results are somewhat weaker. This can be attributed to the following side factors: the initial population in the aforementioned studies is different than this one (i.e., the initial best fitness in [30] is c. 450, whereas it is c. 900 here) and the method in this study operates on a continuously altering fitness landscape. Furthermore, using only a simple mutation operator undoubtedly limits the pace of the evolutionary search. Nevertheless, note that this is the first attempt of a simulated multi-NP treatment using the physics-based tumour simulator.

Given that the real-world problem described here needs to be under a metameric representation, the specialized crossover operators required are omitted. Instead, the simplest variation of evolution, mutation operator, is implemented. As a result, the genome length is controlled to a greater degree compared with a case of utilizing specialized metameric crossover operators. Moreover, as no significant difference arises from having both addition and deletion in the genome length while testing the abstract NK model, this is not tested on the computationally expensive simulator. As an aspect of future work, different growth and deletion schemes will be explored, such as those which self-adapt over time (after [34]), along with recombination and crossover operators.

Despite the fact that the amount of types of NPs in the evolution with addition of one type of NPs (Fig. 15) does not reach the maximum available (i.e., 10), in the case of adding two types of NPs per step (Fig. 18) the composition reaches the highest possible (i.e., 9 —because the initial composition is 1 and there are 2 added per step). Nonetheless, the best fitnesses found in both cases are quite close. Consequently, the bloat effect is obvious for the addition of two types of NPs per step, but the same can not be said for the case of adding one type of NPs.

VII. Conclusion

The effects of genome length are studied here both on an abstract model and a real-life problem of designing a NP-based anti-cancer treatment. Firstly, the influence that the ruggedness of the fitness function landscape has on the genome length through evolution is investigated with the abstract NK model. Growth is observed, with the expansion of genome lengths not obstructed by the ruggedness of the fitness landscape. On the contrary, the expansion of genome lengths can be encouraged by the topology of such landscapes, where typical peaks of low amplitude increase the possibility of higher fitness outcome per the added randomly generated sequence. It is noteworthy that no specific advantage is implemented in the abstract model for larger lengths of genomes, thus the observed limited growth (contrast to what happens during bloat situations) is explicitly due to the inherent nature of evolution over rugged fitness landscapes.

Then, by optimizing the design of NP drug-delivery systems in a cancer simulator, we investigate the increase of the genome length in a real-world problem. Despite the fact that no indication of the best treatment composition (or the number of different types of NPs) is included in the model, evolved solutions converge to treatments with eight different types of NPs, for the method that adds one type of NPs per step. For the method that adds two types of NPs per step, evolved solutions converge to slightly more complex treatments (i.e., 9 types of NPs). This general behaviour of higher growth with larger sequences added correlates well with observed behaviour in the NK model. Moreover, as deduced here and by using other versions of the NK model (after [24]), the gradual growth through small step increases in genome length appears more appropriate in the application domain. That is, whilst the fitness of the solutions found is quite similar, the higher complexity of NP-based cancer treatment drug delivery systems is harder to produce, will probably prove to be more toxic, and has the greater potential for unintended consequences when used *in vivo*.

Acknowledgment

This work was supported by the European Research Council under the European Union's Horizon 2020 research and innovation programme under grant agreement No. 800983.

References

[1] L. J. Fogel, A. J. Owens, and M. J. Walsh, "Artificial intelligence through a simulation of evolution," in *Proc. of the 2nd Cybernetics Science Symp., 1965*, 1965.

[2] M. L. Ryerkerk, R. C. Averill, K. Deb, and E. D. Goodman, "A survey of evolutionary algorithms using metameric representations," *Genetic Programming and Evolvable Machines*, vol. 20, no. 4, pp. 441–478, 2019.

[3] ——, "Solving metameric variable-length optimization problems using genetic algorithms," *Genetic Programming and Evolvable Machines*, vol. 18, no. 2, pp. 247–277, 2017.

[4] J. Serrano González, M. Burgos Payán, J. M. R. Santos, and F. González-Longatt, "A review and recent developments in the optimal wind-turbine micro-siting problem," *Renewable and Sustainable Energy Reviews*, vol. 30, pp. 133–144, 2014.

[5] Y. Chen, H. Li, K. Jin, and Q. Song, "Wind farm layout optimization using genetic algorithm with different hub height wind turbines," *Energy Conversion and Management*, vol. 70, pp. 56–65, 2013.

[6] F. Luna, J. J. Durillo, A. J. Nebro, and E. Alba, "Evolutionary algorithms for solving the automatic cell planning problem: a survey," *Engineering Optimization*, vol. 42, no. 7, pp. 671–690, 2010.

[7] C.-K. Ting, C.-N. Lee, H.-C. Chang, and J.-S. Wu, "Wireless heterogeneous transmitter placement using multiobjective variable-length genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 39, no. 4, pp. 945–958, 2009.

[8] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 550–586, 2016.

[9] K.-J. Kim and S.-B. Cho, "Automated synthesis of multiple analog circuits using evolutionary computation for redundancy-based fault-tolerance," *Applied Soft Computing*, vol. 12, no. 4, pp. 1309–1321, 2012.

[10] R. S. Zebulum, M. Vellasco, and M. A. Pacheco, "Variable length representation in evolutionary electronics," *Evolutionary Computation*, vol. 8, no. 1, pp. 93–120, 2000.

[11] Y. A. Sapargaliyev and T. G. Kalganova, "Open-ended evolution to discover analogue circuits for beyond conventional applications," *Genetic Programming and Evolvable Machines*, vol. 13, no. 4, pp. 411–443, 2012.

[12] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," *IEEE transactions on neural networks*, vol. 8, no. 3, pp. 694–713, 1997.

[13] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[14] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, vol. 10, no. 2, pp. 99–127, 2002.

[15] P. P. Palmes, T. Hayasaka, and S. Usui, "Mutation-based genetic neural network," *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 587–600, 2005.

[16] P. Angeline, G. Saunders, and J. Pollack, "An evolutionary algorithm that constructs recurrent neural networks," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 54–65, 1994.

[17] S. Ohno, *Evolution by Gene Duplication*. Springer, New York, 1970.

[18] S. P. Otto and J. Whitton, "Polyploid incidence and evolution," *Annual review of genetics*, vol. 34, no. 1, pp. 401–437, 2000.

[19] A. L. Hughes, "The evolution of functionally novel proteins after gene duplication," *Proceedings of the Royal Society of London. Series B: Biological Sciences*, vol. 256, no. 1346, pp. 119–124, 1994.

[20] S. Kauffman and S. Levin, "Towards a general theory of adaptive walks on rugged landscapes," *Journal of theoretical Biology*, vol. 128, no. 1, pp. 11–45, 1987.

[21] A. Ghaffarizadeh, R. Heiland, S. H. Friedman, S. M. Mumenthaler, and P. Macklin, "PhysiCell: an open source physics-based cell simulator for 3-D multicellular systems," *PLoS computational biology*, vol. 14, no. 2, p. e1005991, 2018.

[22] A. Ghaffarizadeh, S. H. Friedman, and P. Macklin, "Biofvm: an efficient, parallelized diffusive transport solver for 3-d biological simulations," *Bioinformatics*, vol. 32, no. 8, pp. 1256–1258, 2015.

[23] S. A. Kauffman, *The origins of order: Self-organization and selection in evolution*. Oxford University Press, USA, 1993.

[24] I. Harvey, "Species adaptation genetic algorithms: A basis for a continuing saga," in *Toward a practice of autonomous systems: Proceedings of the first european conference on artificial life*, 1992, pp. 346–354.

[25] J. Metzcar, Y. Wang, R. Heiland, and P. Macklin, "A review of cell-based computational modeling in cancer biology," *JCO clinical cancer informatics*, vol. 2, pp. 1–13, 2019.

[26] N. R. Stillman, M. Kovacevic, I. Balaz, and S. Hauert, "In silico modelling of cancer nanomedicine, across scales and transport barriers," *NPJ Computational Materials*, vol. 6, no. 1, 2020.

[27] R. J. Preen, L. Bull, and A. Adamatzky, "Towards an evolvable cancer treatment simulator," *Biosystems*, vol. 182, pp. 1 – 7, 2019.

[28] M.-A. Tsompanas, L. Bull, A. Adamatzky, and I. Balaz, "Haploid-diploid evolution: Nature's memetic algorithm," 2019. [Online]. Available: https://arxiv.org/abs/1911.07302

[29] ——, "Utilizing differential evolution into optimizing targeted cancer treatments," in *Modern Trends in Controlled Stochastic Processes:*, A. Piunovskiy and Y. Zhang, Eds. Cham: Springer International Publishing, 2021, pp. 328–340.

[30] ——, "Novelty search employed into the development of cancer treatment simulations," *Informatics in Medicine Unlocked*, vol. 19, p. 100347, 2020.

[31] J. Ozik, N. Collier, R. Heiland, G. An, and P. Macklin, "Learning-accelerated discovery of immune-tumour interactions," *Molecular Systems Design & Engineering*, 2019.

[32] M.-A. Tsompanas, L. Bull, A. Adamatzky, and I. Balaz, "Metameric representations on optimization of nano particle cancer treatment," *Biocybernetics and Biomedical Engineering*, vol. 41, no. 2, pp. 352–361, 2021.

[33] ——, "In silico optimization of cancer therapies with multiple types of nanoparticles applied at different times," *Computer Methods and Programs in Biomedicine*, vol. 200, p. 105886, 2021.

[34] L. Bull, "Coevolutionary species adaptation genetic algorithms: growth and mutation on coupled fitness landscapes," in *2005 IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2005, pp. 559–564.