Accelerated Optimisation of Chemotherapy Dose Schedules Using Fitness Inheritance

Robert Barbour, David Corne and John McCall

***Abstract*— Cancer treatment by chemotherapy involves multiple applications of toxic drugs over a period of time. Optimising the schedule of these treatments can improve the outcome for the patient. A schedule of treatment and its effect on the tumour can be simulated by a mathematical growth model. However, when used in conjunction with an Evolutionary Algorithm (EA) to search for effective treatment schedules, the frequent use of the model can become computationally onerous. One approach to improve the efficiency of EAs is to use 'fitness inheritance', in which, for a proportion of candidate solutions, simple means are used to guess the fitness, rather than use the computationally intensive model. We investigate two versions of fitness inheritance for the chemotherapy schedule optimisation problem, and demonstrate the significant improvement in efficiency that can be achieved. In particular, we find that the Averaged Inheritance strategy is highly effective in this case, and is strongly recommended for use in further investigations of chemotherapy optimisation using population-based search.**

I. Introduction

Cancer is a major cause of death in modern society. Chemo-therapy is the treatment of cancer using highly toxic drugs which have been developed by pharmaceutical research. These drugs attack and kill cancer cells, but also damage healthy cells. To control damage to vital organs and tissue the drug doses have to be limited. A dose of drug(s) is given, then a period for the body to recover is followed by another dose, and so on. Dose schedules are developed from clinical trials and the experience of oncologists in practice. They have become more complex with the increase in the number of drugs available and in the use of multi-drug treatments. Different drugs affect different organs, so limiting the dosage to control side effects also becomes more complicated. With a treatment schedule of ten or more dose/rest periods and ten or more drugs available in different doses, there are a huge number of treatment combinations possible, far too many for clinical trials to evaluate. Thus it is almost certain that the drugs available are not being used to best advantage. EAs have been investigated in this context, but since EAs typically require tens or hundreds of thousands of fitness evaluations, the computational cost can be prohibitive in practice. In this paper we investigate whether fitness inheritance strategies ([14]) can yield a useful improvement in efficiency in this case, without unduly adverse effects on solution quality. In the remainder, Section II introduces the problem, Sections III and IV describe our experiments and results, and we provide a concluding discussion in Section V.

II. The Problem: Background

A. Cancer and Chemotherapy

Cancer is not a single disease; the biology of cancers is described in [3], [15]. Cancerous tumours are characterised by rapidly dividing cells. Chemotherapy is the application of drugs in an attempt to reduce or eliminate the tumour, whilst managing damage to the immune system and avoiding unacceptable side effects to the patient. The schedule of drugs and doses to be given is thus a balance between killing cancer cells and limiting damage. For some time the use of computer simulations of chemotherapy has been developing in this field as a method of finding alternative or novel treatments.

Modelling Tumours and Chemotherapy

Modelling of chemotherapy includes:

- tumour growth;
- tumour cells killed by drug(s);
- constraint(s) on drug application, dose rates and amounts etc.;
- meeting the objective(s) of treatment.

A well-known growth equation due to Gompertz is:

$$f(N) = \lambda N - \mu N ln N \tag{1}$$

where N is the number of tumour cells and λ and μ are growth parameters. This equation is the basis for much work in this field[16].

A Basic Chemotherapy Response Model

To a growth equation, a 'cell kill' or 'cell loss' term is added to model the effect of the drug on the tumour cell population.

If κ is the effectiveness and c the concentration of a cytotoxic drug, adding the 'kill' term $-\kappa c N$ to the Gompertz growth model Martin and Teo [4] show that :

$$\frac{dN}{dt} = N(t)\lambda ln\left(\frac{\Theta}{N(t)}\right) - \kappa c(t) N \tag{2}$$

where $N(t)$ is the number of tumour cells at time t, $c(t)$ is the drug concentration at time t and λ, Θ are tumour parameters.

Equation (2) can be extended to cover multiple drugs by summing the kill terms for each drug.

Robert Barbour and David Corne are with the Department of Computer Science, School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh, EH14 4AS, UK; email: barbour_rl@yahoo.co.uk; dwcorne@gmail.com

John McCall is with the IDEAS Research Institute, Robert Gordon University, Aberdeen, UK; email: j.mccall@rgu.ac.uk

Constraints

Due to the toxicity of anti-cancer drugs, there are limits on doses to protect healthy cells and control side effects. Constraints may also be applied to meet treatment objectives, such as keeping the tumour below a certain size or maximising patient suvival time.

Chemotherapy Optimisation and related work

When EAs are applied to the cancer chemotherapy scheduling problem, each individual in the population represents one set of drug doses and rest periods. Fitness is estimated via a mathematical model of tumour growth over time and the effect of the drugs. This tumour model simulates the interaction of the drugs throughout the treatment period.

McCall and Petrovski [5] present a decision support system called the Oncology WorkbenCH (OWCH) for oncologists seeking novel treatment schedules. The main theme is the constrained optimisation of multi-drug chemotherapy using a model with the Gompertz equation. The treatment schedule is encoded in the EA by a control vector (chromosome) of binary values. This is simply the dose for each drug for each time interval, coded as four bits representing a dose of between 0 and 15 units. Constraints are set to limit drug concentration, total exposure and tumour size during treatment. The toxic side effects on other organs are also constrained using a potency parameter for each drug. Two EAs are used. An objective function minimising tumour size determines if a cure is possible. If this is unsuccessful a second EA determines a palliative schedule. Both EAs apply penalties to their objective functions based on violations of constraints.

In [9] Petrovski and McCall further explore the problem of optimisation posed by multi-drug chemotherapy with EAs using exponential, von Bertalanffy, Verhulst and Gompertz growth equations. Experiments are described which use all four growth models and both treatment goals. These are compared to a common chemotherapy schedule of three drugs and show better performance. The authors conclude the various growth models do not affect the EA's ability to derive treatment schedules.

A Particle Swarm Optimisation (PSO) algorithm is investigated by Petrovski et al. [11] in experiments simulating and optimising chemotherapy schedules. They showed that PSO was effective for this problem, with speed and quality improvements over the EAs with which it was compared. Finally, we note the work of Ochoa et al. [7], which investigated drug chemotherapy using one drug and a computationally expensive tumour and immune system response simulation, in an EA which takes several days for a single run, thereby excluding comprehensive statistical analysis.

Accelerating Evolutionary Algorithms

Acceleration of optimisation in this context is clearly a major issue, and there have been a number studies so far that try to address it. E.g., Petrovski et al. [12] shows that the mutation and crossover probabilities can be significant in EA performance, and in [8] Petrovski et al. showd that online tuning of mutation probability and step size was worthwhile in achieveing faster and better performance. Meanwhile, use of parallel hardware resources is obviously applicable, as studied for chemotherapy optimmisation in Tan et al. [17].

However, considering the computational demands of complex simulation models, as well as the likely resources available in practice, more significant speedup of serial implementations is needed. Along these lines, one area of research is the general speedup that can be available when EAs are combined with machine learning strategies (e.g. [6]), or when techniques are used to wisely bias the genetic operators towards good areas. Along these lines, Petrovski et al. [10] compare the performance of an EA with Population-based Incremental Learning (PBIL), finding that a large reduction in fitness evaluations is possible. meanwhile, Godley et al. [1] use parent fitness to bias the crossover operator in a single drug chemotherapy problem. Fitness function evaluations were found to be significantly less than an EA using either single point or uniform crossover.

One highly promising technique that has not yet been explored for chemotherapy schedule optimisation is Fitness Inheritance, originally suggested by Smith et al.[14], and [2]. Smith et al. proposed that computationally expensive fitness calculations could be done only for a portion of the population. In contrast with Grefenestette and Fitzpatrick's proposal to only partially evaluate fitness (for all th epopulation), Smith et al. suggests full evaluation of only part of the population, with some of the population having their fitness values derived in a simple way from the fitnesses of their parents. Two varieties are described: *averaged inheritance* – the arithmetical mean of the parents' fitness; and *proportional inheritance* – a weighted average based on the contributions of genes from each parent. Tests conducted on *OneMax* showed that efficient convergence is achieved even with only 10% actual fitness function evaluations. On a more realistic problem (aircraft routing), Smith et al found that a proportional inheritance version of fitness inheritance achieved excellent results even when less than 1% of the population was subject to proper fitness evaluation. Meanwhile, Sastry et al.[13] investigated fitness inheritance from a theoretical viewpoint, and verified the potential speedup available through fitness inheritance in certain circumstances, albeit on a simple (analysable) problem landscape.

Fitness Inheritance is clearly an attractive candidate for investigating in the context of chemotherapy schedule optimisation, and has not yet been investigated for this purpose to our knowledge. It is attractive partly for its simplicity, but also with excellent results reported on other problems. We therefore investigate fitness inheritance in this paper, and examine what speedup is achievable and how this speedup interacts with the quality of solutions.

III. Experiments: Evaluating Fitness Inheritance Strategies

The Gompertz growth model is chosen because it is commonly used in this field and the Java source code for an implementation is available. Modelling of the tumour growth and chemotherapy drug interaction is an implementation of equations (1) and (2) and the constraints described in Section II-A due to Petrovski et al. [10]. The encoding of genes for this model is the integer-based drug dose array described in [10]. For each of 10 drugs, during a 10 week schedule, a chromosome simply comprises a list of 100 integers constrained in the range $0 \ldots 15$. Each successive group of 10 numbers relates to the dose concentrations for each of 10 specified drugs over a specific week. When the chromosome is interpreted, an integer value is scaled appropriately between zero and a maximum dose, according to the specific drug represented.

A. Control Experiment

As a control, we use a standard EA which uses the Gompertz tumour growth model for fitness evaluation, and which employs no fitness inheritance. This EA is otherwise the same as that used in the fitness inheritance experiments, and uses a steady-state, replace-worst reproduction strategy. A population of randomly generated individuals is evolved for up to 50,000 generations (equivalent in the steady state model to 50,000 evaluations). Binary tournament selection is used, along with standard two point cross-over and Gaussian mutation of a single randomly chosen gene.

Cross-over and mutation are used with probability of 1.0, and if the same parent is selected twice this is effectively mutation only. This schema is chosen for its simplicity and is not tuned in any way. However, some preliminary work was done to establish a suitable population size and the number of runs required to produce results and find a reasonable number of feasible solutions to the problem.

B. Termination Criteria

In the approach we undertake in this paper (following [11], [5], [9]), the model returns fitness values that indicate the size of the tumour following the end of the treatment schedule, and indications of the damage caused to various organs. These are combined into a single objective fitness value, however a ‘good’ fitness value does not necessarily mean the solution is feasible, since it may unduly tradeoff a small tumour size for a high level of damage. The criteria for a solution to be deemed feasible are:

- a positive value of fitness;
- no penalties applied for side effects on organs;
- no penalties for exceeding instantaneous or cumulative drug dose limits;
- tumour does not exceed its original size during the treatment period.

These criteria apply to all experiments described in this paper. Clearly, multi-objective approaches need to be studied for this problem, and this is the topic of future work. Here we focus only on the potential benefits of fitness inheritance, and maintain a single-objective approach for simplicity.

C. Measuring Performance

In the control and all subsequent experiments when a chromosome is found which meets the termination criteria, it is evaluated using the model and the total number of evaluations required to achieve this is recorded and the algorithm terminates. This is described as finding the first feasible solution. The count of model evaluations does not include those made at the initialisation of the starting population, which is the same for all experiments.

D. Inherited Fitness Experiments

Fitness inheritance is an extremely simple concept: the fitness of a child is derived from the fitness of its parents. Smith et al. [14] define two kinds of inheritance: averaged and proportional. Using averaged inheritance is to simply assign the arithmetical average of the fitness of the two parents to the child’s fitness. In proportional inheritance a weighted average is calculated from the contribution of genetic material by each parent – the larger the number of a parent’s genes that are used in the crossover combination, the more the average is weighted towards the fitness of that parent. The implementation of this is straighforward, given the two-point crossover has a probability of 1 in the EA.

A modified version of the EA allows a controlled percentage of the generated children to have their fitness estimated by inheritance instead of evaluated by the tumour model.This reduces the number of times the model is used. The fraction of children whose fitness is estimated by inheritance varies from zero i.e. the base EA, in steps of 5% up to 95%.

Both types of inheritance are used in separate experiments. Actual model evaluations are counted until the termination criteria given in Section III-A are met. The actual fitness of that first solution is evaluated using the model, recorded and the run ended. Model evaluations to initialise the population are not included in the count for the first feasible solution.

IV. Results

Experiments are presented in chronological order in this section and the results are discussed.

A. Control Experiment

Notice that, given the criteria discussed in section III-B it is expected that some runs will not find a feasible solution at all. Meanwhile, in order to ensure that sample sizes of results are sufficiently large for statistical testing without execution times being excessively long, an experimental regime of groups of 500 runs each with a population of 500 evolving for up to 50,000 generations is chosen.

A single set of 2,000 runs was executed to produce a larger sample to establish the mean and standard deviation of key parameters for the experiments. Table I shows the results of this effort. In this table, the number of model evaluations is that required to find the first feasible solution in the run.

For each of the the two types of inheritance a secondary baseline run is conducted, in which the parameters were set to make it equivalent to the baseline EA (i.e. inheritation proportion set at zero). These checks validated that the fitness inheritance code variants were performing correctly. The results of these runs are referred to later as baselines for the associated experiments, and are also shown in the lower part of Table I.

TABLE I

Results of baseline EA runs. For each of the three EA variants, 2,000 runs were done, each for a total of 50,000 evaluations, and the table reports the mean and standard deviation of the number of evaluations taken to find the first feasible solution (over runs which achieved at least one feasible solution), and similar statistics for best fitness obtained. T-tests confirm that the EA variants with inheritance proportion set at 0 are no different from the standard EA.

	Model Evals[1]		Fitness[1]		t-test
Reference		Std.		Std.	p value
Experiment[2]	Mean	Devn	Mean	Devn	rel to EA
Base EA	10,603	2354	0.396	0.045	1.000
Prop. Inhrt.	10,460	2438	0.403	0.046	0.625
Aver. Inhrt.	10,529	2375	0.395	0.043	0.661

The Student's t-test p-values in Table I show that these secondary baseline figures for the number of evaluations and fitness are representative of the EA and are not low probability outlying samples which could skew results.

B. Fitness Inheritance: Model Savings

Experiments were performed with the amount of fitness estimation by inheritance varying from 0% in 5% increments for both proportional and averaged inheritance up to 95%. For each inheritance method and inheritance proportion, 500 trial runs were performed and the figures provided are averaged over the successful runs. Results are also obtained at very high averaged inheritance rates. Table II shows the results.

The Evals column shows the number of model evaluations to find a feasible solution (as defined in section III-B) averaged over the number of runs which found solutions. Fitness values are also averaged over the runs which found feasible solutions. The Success Rate is defined as the number of runs which found a feasible solution expressed as a percentage of the total number of runs (500). For example (clarifying the meaning of the table) we note that when Proprtional Inheritance is used and 65% of children receive an inherited fitness (rather than a time consuming model evaluation), the first feasible solution is found on average after 7,593 evaluations, and 40% of runs achieve a feasible solution; when Averaged Inheritance is used, and 85% of children receive an inherited fitness, the corresponding figures are 3,830 evaluations and 42%. It is clear from Table II that substantial savings in model evaluations are achieved at high rates of inheritance, with very little effect on fitness values. The value of Proportional Inheritance drops sharply beyond the region of 80% inheritance – although there remain significant savings in model evaluations, the success rate becomes very low. In contrast, Average Inheritance does not suffer a sharp drop in success rate until around 90% inheritance.

TABLE II

Mean evaluations until first fasible solution, and success rates, for both proportional inheritance and averaged inheritance, for inheritance proportions ranging from 0 to 95%

%	Proportional Inheritance			Averaged Inheritance		
Est.	Evals	Fitness	Succ	Evals	Fitness	Succ
0	10,460	0.403	49	10,529	0.395	50
5	10,178	0.395	47	9,947	0.394	49
10	9,550	0.396	46	10,061	0.400	51
15	9,695	0.395	51	9,354	0.397	49
20	9,186	0.402	48	9.079	0.401	50
25	9,262	0.402	48	8,804	0.400	51
30	8,451	0.396	47	8,470	0.397	50
35	7,750	0.396	47	8,235	0.399	48
40	8,132	0.396	49	7,825	0.401	49
45	7,910	0.395	46	7,517	0.400	54
50	7,574	0.394	40	6,996	0.401	43
55	7,544	0.395	41	6,611	0.400	49
60	7,663	0.401	40	5,977	0.400	48
65	7,593	0.396	40	5,757	0.398	50
70	7,909	0.395	38	5,255	0.395	46
75	7,559	0.398	30	4,896	0.401	43
80	6,901	0.389	25	4,409	0.396	42
85	5,559	0.394	14	3,830	0.393	42
90	4,219	0.383	4	3,029	0.396	35
95	0	0	0	1,945	0.384	20
96				1,662	0.391	11
97				1,230	0.379	4
98				949	0.418	1
99				337	0.345	0

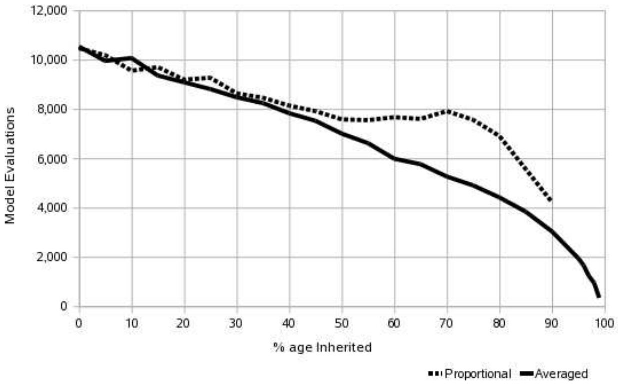

Fig. 1. Visualising the savings in model evaluations achieved using different levels of inheritance, for both Averaged and Proportional inheritance.

Figure 1 illustrates the data from Table II. Averaged inheritance can produce large savings: 80% saving of model evaluations at 95% inheritance. This is a significant reduction in computational burden. When the success rate is examined, however, there is another aspect to consider.

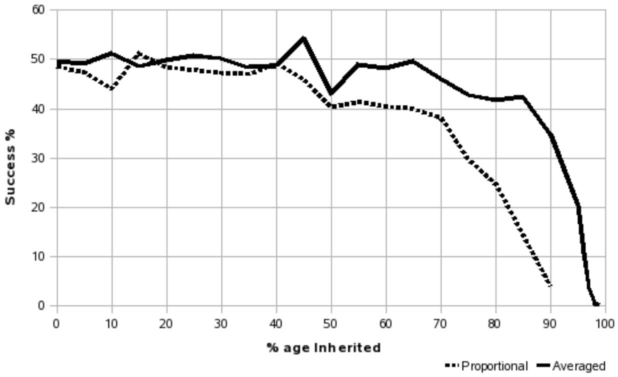

Fig. 2. The success rates (percentage of runs that achieve a feasible solution) as a function of the level of inheritance, shown for both Averaged and Proportional inheritance

It can be seen from Figures 1 and 2 that inheritance rates up to about 60% can achieve significant savings and still find almost as many solutions as the control EA. However the large savings achieved by values of averaged inheritance greater than 80% are associated with a rapidly declining success rate. There is clearly a trade-off between model saving and success rate.

V. Discussion and Interpretation

A. A Measure of Success

A lower success rate means fewer solutions in a given number of runs. The definition of success rate given in Section IV-B makes no distinction between a solution which uses only a few evaluations and one which requires a large number of model evaluations to arrive at a feasible solution.

It is useful to introduce a normalising measure which combines success rate and model evaluations. If n_{fs} is the number of feasible solutions found in N_{runs} and ϵ is the average number of evaluations to find a feasible solution and

$$\delta = \frac{n_{fs} * 10^6}{\epsilon * N_{runs}}$$

then δ (delta) represents the number of solutions per million run evaluations. The scale factor of 10^6 renders the values for δ as userfriendly numbers in the order of 10 to 100 feasible solutions per million run-evaluations.

B. Significance of Results

Detailed results are subjected to significance testing and shown in Table III. Student's t-test p-values are relative to the control EA i.e. the top (zero inheritance) row of the table.

C. Proportional Inheritance

The t-test figures shown in Table III show that the reduction in the number of evaluations is statistically significant, as the p-values are generally less than 10^{-6}. Significance testing of fitness values (not shown) is inconclusive.

D. Averaged Inheritance

Table III p-values show that the reduction in evaluations is statistically significant for averaged inheritance. Significance testing of fitness (not shown) finds that fitness values are as good as the control EA.

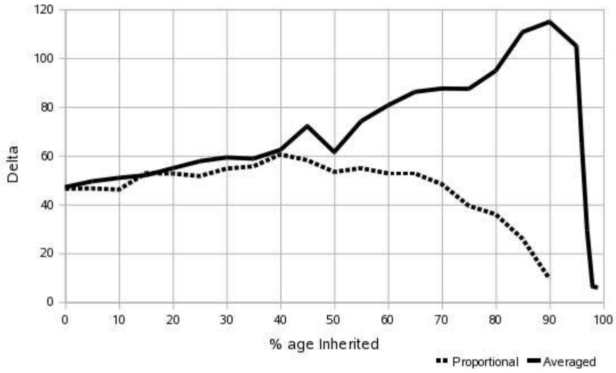

Fig. 3. Delta Values (normalised success rates) for Averaged and Proprtional Inheritance, as a function of the level of inheritance.

Whilst success rates in Figure 2 are generally lower than in the control experiment EA, the delta values in Figure 3 rise noticeably above the control: quite dramatically up to 90% averaged inheritance. Additional runs to 99% inheritance are conducted and a rapid decline in delta is observed.

In [14] Smith reports "*Surprisingly, the best results were obtained with only one individual actually evaluated per generation.*" Our best result of 337 model evaluations to find a solution also occurs at the highest rate of inheritance in the experiments.

E. Comparison

From Figure 2 it is clear that for each type of inheritance the success rate curves are of similar shapes, with averaged inheritance performing better above 40%. The delta curves in Figure 3 show a different picture. Although similar at low inheritance rates, proportional values decline above 40% in contrast to averaged inheritance, whose delta increases dramatically.

For inheritance rates up to 50%, proportional inheritance is as successful as the control EA and achieves delta values above the EA, so could possibly be a useful tool in the chemotherapy scheduling problem. However at rates above 60%, averaged inheritance is much better than proportional and achieves large model savings and yields high numbers of solutions per million run-evaluations at inheritance rates up to 95%. Exploiting these properties to reduce the computational load in EAs looks to be a promising prospect.

VI. Conclusions

A summary of good results from the experiments is shown in Table IV. Ranking results by the number of model evaluations shows that averaged inheritance is productive in finding feasible solutions, and gives the best savings by a large margin. As well as being a very simple method to implement,

TABLE III

RESULTS NORMALISED IN TERMS OF NUMBER OF FEASIBLE SOLUTIONS FOUOND PER FULL MODEL EVALUATION, AND ASSOCIATED STATISTICS.

% age Inheritance	Proportional Inheritance				Averaged Inheritance				% age Inheritance
	Model Evals	No. of Solns. Found[1]	t-test p val. Evals	Delta Solns/10^6 run-evals	Model Evals	No. of Solns. Found[1]	t-test p val. Evals	Delta Solns per 10^6 run-evals	
0	10,460	243	1.000000	46	10,529	248	1.000000	47	0
5	10,178	237	0.183803	47	9,947	246	0.004426	49	5
10	9,550	220	0.000017	46	10,061	256	0.018719	51	10
15	9,695	256	0.000267	53	9,354	243	$< 10^{-6}$	52	15
20	9,186	242	$< 10^{-6}$	53	9,079	249	$< 10^{-6}$	55	20
25	9,262	239	$< 10^{-6}$	52	8,804	254	$< 10^{-6}$	58	25
30	8,630	236	$< 10^{-6}$	55	8,470	251	$< 10^{-6}$	59	30
35	8,451	235	$< 10^{-6}$	56	8,235	242	$< 10^{-6}$	59	35
40	8,132	246	$< 10^{-6}$	61	7,825	244	$< 10^{-6}$	62	40
45	7,910	230	$< 10^{-6}$	58	7,517	271	$< 10^{-6}$	72	45
50	7,574	202	$< 10^{-6}$	53	6,996	215	$< 10^{-6}$	61	50
55	7,544	207	$< 10^{-6}$	55	6,611	245	$< 10^{-6}$	74	55
60	7,663	202	$< 10^{-6}$	53	5,977	241	$< 10^{-6}$	81	60
65	7,593	200	$< 10^{-6}$	53	5,757	248	$< 10^{-6}$	86	65
70	7,909	191	$< 10^{-6}$	48	5,255	230	$< 10^{-6}$	88	70
75	7,559	149	$< 10^{-6}$	39	4,896	214	$< 10^{-6}$	87	75
80	6,901	124	$< 10^{-6}$	36	4,409	209	$< 10^{-6}$	95	80
85	5,559	72	$< 10^{-6}$	26	3,830	212	$< 10^{-6}$	111	85
90	4,219	20	$< 10^{-6}$	9	3,029	174	$< 10^{-6}$	115	90
95		0		0	1,945	102	$< 10^{-6}$	105	95
96					1,662	54	$< 10^{-6}$	65	96
97					1,230	18	$< 10^{-6}$	29	97
98					949	3	$< 10^{-6}$	6	98
99					337	1	$< 10^{-6}$	6	99

TABLE IV

SUMMARY OF THE BEST RESULTS ACHIEVED FROM EACH OF AVERAGED AND PROPRTIONAL INHERITANCE, COMPARED WITH THE CONTROL EA

Algorithm	% age Est.	Model Evals	Delta δ	Success %
Averaged Inheritance	90	3,029	115	35
Proport'l Inheritance	90	4,219	9	4
Control EA	0	10,529	47	50

the averaged inheritance technique itself has an extremely low computational burden and therefore is potentially a very useful technique in chemotherapy scheduling.

It is worth emphasising that the δ speedup values are useful for estimating the effective speedup that can be obtained when we execute the optimisation on serial resources. However, with proper use of multicore hardware (or other parallel resources) the effective speedups, in terms of wall-clock time to achieve first feasible solution, become closer to those suggested in the Model Evals column in table III. E.g. if we use 97% inheritance in the Averaged Inheritance method, and execute (say) 50 runs in parallel, speedup of eight to nine-fold is achievable in finding first feasible solution, when compared to the (similarly implemented in parallel) control EA.

It is interesting to speculate on the relative performance of Averaged and Proprtional Inheritance. Averaged Inheritance was clearly more successful in maintaining valid progress in the search process, while we might view the relatively poorer performance of Proprtional Inheritance as a sign that the inherited fitnesses in this case were more likely to be misleading. Consider a combination of a quite fit and a quite poor parent; in the case of Averaged Inheritance, the fitness assigned to the child will always be the modest average of the two parents; such a solution will not unduly influence the search, since we can expect a fair proportion of the population to be fitter, and hence more likely to be selected. However, in Proprtional Inheritance, it is possible that such a child may be quite poor in real' fitness, but – if it happens to inherit mostly the genes of the fitter parent – be given a fairly influential fitness value, which will then mislead the search. The key point is that proportional inheritance is likely to be very sensitive to the ruggedness of the landscape in question; where it is possible for near-neighbours to have significant differences in fitness (which is quite so in this case), the use of Proportional Inheritance is prone to the possibility of yielding unduly influential poor chromosomes. With Average Inheritance, the potential damage from such chromosomes is more limited by a stricter bound on the fitness value they can inherit.

Ongoing research (some of which we mention earlier in this paper) is investigating a range of alternative optimisation algorithms for the cheotherapy scheduling problem, as well as the use of a range of alternative tumour growth models; active and promising areas in this field include further investigation of particle swarm optimisation approaches, as well as further investigation of multi-objective approaches (and of course approaches that may combine the two). An obvious avenue for further study will be to investigate fitness inheritance in the context of these approaches.

Meanwhile, this study has concentrated on trying to efficiently find the first feasible solution on each run. This was a convenient measure to form the basis of comparison, although the first feasible solution is itself rarely fit enough to ensure a treatment effective enough for use. Further studies are needed to test whether the continued use of fitness inheritance, beyond the point of first feasible solution, is able to maintain the speed advantage without compromising the ultimate fitness values achieved.

Finally, it is interesting to speculate on the relative performance of Averaged and Proprtional Inheritance. Averaged Inheritance was clearly more successful in maintaining valid progress in the search process, while we might view the relatively poorer performance of Proportional Inheritance as a sign that the inherited fitnesses in this case were more likely to be misleading. Consider a combination of a quite fit and a quite poor parent; in the case of Averaged Inheritance, the fitness assigned to the child will always be the modest average of the two parents; such a solution will not unduly influence the search, since we can expect a fair proportion of the population to be fitter, and hence more likely to be selected. However, in Proprtional Inheritance, it is possible that such a child may be quite poor in real' fitness, but – if it happens to inherit mostly the genes of the fitter parent – be given a fairly influential fitness value, which will then mislead the search. The key point is that proportional inheritance is likely to be very sensitive to the ruggedness of the landscape in question; where it is possible for near-neighbours to have significant differences in fitness (which is quite so in this case), the use of Proportional Inheritance is prone to the possibility of yielding unduly influential poor chromosomes. With Average Inheritance, the potential damage from such chromosomes is more limited by a stricter bound on the fitness value they can inherit. One corollary to this argument, suggesting a particular line of future study, is that a stricter limit on inherited fitness may be even more beneficial. For example, fitness could be the lower of the Proportional and Average values, or the child may simply inherit the fitness of its weakest parent.

Acknowledgements

We are grateful to Andreas Petrovski for the use of his codebase in implementing the tumour model, and we are grateful to Manjula Dissanayake and to Sebastian Collins for discussions on this topic while they were undertaking related projects.

References

[1] P. M. Godley, D. E. Cairns, J. Cowie, and J. McCall. Fitness directed intervention crossover approaches applied to bio-scheduling problems. *IEEE*, 2005.

[2] J. J. Grefenstette and J. M. Fitzpatrick. Genetic search with approximate function evaluations. In *Proceeding of an International Conference on Genetic Algorithms and Their Application*, pages 112–120, 1985.

[3] F. MacDonald and C. H. J. Ford. *Molecular biology of Cancer*, chapter 1. BIOS Scientific Publishers, 1997.

[4] R. B. Martin and K. L. Teo. *Optimal Control of drug administration in cancer chemotherapy*. World Scientific, 1994.

[5] J. McCall and A. Petrovski. A decision support system for cancer chemotherapy using genetic algorithms. In M. Mouhammadian, editor, *Proceeding of the International Conference on Computational Intelligence for Modelling, Control and Automation*, volume 1, pages 65–70, Vienna, Austria, 1999. IOS Press : ISBN 90-5199-474-5.

[6] R. S. Michalski. Learnable evolution model: Evolutionary processes guided by machine learning. *Machine Learning*, 38(1-2):9–40, January 2000.

[7] G. Ochoa, M. Villasana, and E. K. Burke. An evolutionary approach to cancer chemotherapy scheduling. *Genetic Programming and Evolvable Machines*, 8(4):301–308, December 2007.

[8] A. Petrovski, A. Brownlee, and J. McCall. Statistical optimisation and tuning of ga factors. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 1, pages 758–764, Edinburgh, Scotland, 2005. IEEE Press.

[9] A. Petrovski and J. McCall. *Soft Computing Techniques and Applications*, chapter Computational Optimisation of Cancer Chemotherapies Using Genetic Algorithms, pages 117–122. Physica Verlag, 2000.

[10] A. Petrovski, S. Shakya, and J. McCall. Optimising cancer chemotherapy using an estimation of distribution algorithm and genetic algorithms. In *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*, volume 1, pages 413–418, Seattle, Washington, USA., July 8-12 2006. ACM Press.

[11] A. Petrovski, B. Sudha, and J. McCall. Optimising cancer chemotherapy using particle swarm optimisation and genetic algorithms. In X. Yao, editor, *Parallel Problem Solving from Nature VIII, 8th International Conference Proceedings, Lecture Notes in Computer Science*, volume 3242, pages 633–641. Springer–Verlag ISBN: 3-540-23092-0 ISSN: 0302-9743, 2004.

[12] A. Petrovski, A. Wilson, and J. McCall. Statistical identification and optimisation of significant ga factors. In *Proceedings of the Fifth Joint Conference on Information Sciences, JCIS 2000*, volume 1, pages 1027 – 1030, Atlantic City, USA, 2000.

[13] K. Sastry, D. E. Goldberg, and M. Pelikan. Don't evaluate, inherit. In L. Spector, E. D. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 551–558, San Francisco, California, USA, 7-11 2001. Morgan Kaufmann.

[14] R. E. Smith, B. A. Dike, and S. A. Stegmann. Fitness inheritance in genetic algorithms. In *Proceedings of the 1995 ACM symposium on Applied computing ISBN:0-89791-658-1*, pages 340–345, Nashville, Tennessee, United States, 1995. ACM.

[15] R. Souhami and J. Tobias. *Cancer and its Management*, chapter 3. Blackwell Science, 1998.

[16] G. W. Swan. Role of optimal control theory in cancer chemotherapy. *Mathematical Biosciences*, 101:237–284, 1990.

[17] T. C. Tan, T. H. Lee, J. Cai, and Y. H. Chew. Automating the drug scheduling of cancer chemotherapy via evolutionary computation. In *Proceedings of the 2002 Congress on Evolutionary Computing*, volume 1, pages 908–913, May 2002.