OPERATIONS RESEARCH
Vol. 54, No. 5, September–October 2006, pp. 829–846
ISSN 0030-364X | EISSN 1526-5463 | 06 | 5405 | 0829

informs

DOI 10.1287/opre.1060.0320

Optimizing Chemotherapy Scheduling Using Local Search Heuristics

Zvia Agur
Institute for Medical Biomathematics, 10 Ha'Teena Street, P.O.B. 282, 60991 Bene-Ataroth, Israel, and Optimata Ltd., 11 Tuval Street, Ramat Gan 52522, Israel, agur@imbm.org

Refael Hassin
School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel, hassin@post.tau.ac.il

Sigal Levy
School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel, and The Academic College of Tel-Aviv-Yaffo, 4 Antokolsky Street, Tel Aviv 61161, Israel, levy@post.tau.ac.il

We develop a method for computing efficient patient-specific drug protocols. Using this method, we identify two general categories of anticancer drug protocols, depending on the temporal cycle parameters of the host and cancer cells: a one-time intensive treatment, or a series of nonintensive treatments. Our method is based on a theoretical and experimental work showing that treatment efficacy can be improved by determining the dosing frequency on the drug-susceptible target and host cell-cycle parameters. Simulating the patient's pharmaco-dynamics in a simple model for cell population growth, we calculate the number of drug susceptible cells at every moment of therapy. Local search heuristics are then used to conduct a search for the desired solution, as defined by our criteria. These criteria include the patient's state at the end of a predetermined time period, the number of cancer and host cells at the end of treatment, and the time to the patient's cure. The process suggested here does not depend on the exact biological assumptions of the model, thus enabling its use in a more complex description of the system. We test three solution methods. Simulated annealing is compared to threshold acceptance and old bachelor acceptance, which are less known variants to this method. The conclusions concerning the three approximation methods are that good results can be achieved by choosing the proper parameters for each of the methods, but the computational effort required for achieving good results is much greater in simulated annealing than in the other methods. Also, a large number of iterations does not guarantee better solution quality, and resources would be better used in several short searches with different parameter values than in one long search.

Subject classification: health care: treatment.
Area of review: Computing and Information Technologies.
History: Received July 2000; revisions received July 2002, January 2004, June 2005, September 2005; accepted October 2005.

1. Introduction

Optimization methods have been employed in medicine to direct clinical interventions such as radiotherapy (e.g., Wu and Zhu 2001) or surgery (e.g., Malinen et al. 2003). As will be explained below, the optimization of drug therapy, as it involves multiple drug-patient interactions, is too complex to be solved using analytical optimal control methods. The aim of this paper is to lay the ground work for a clinically implementable heuristic optimization method for drug therapy in oncology.

Oncology patients are treated nowadays using traditional and novel modes of anticancer therapy. The traditional modes include surgery, radiotherapy, and classical chemotherapy; that is, toxic, natural, or artificially synthesized nonspecific products, which inhibit growth of rapidly proliferating cells. The novel anticancer therapies involve, for example, the disruption of signal transduction pathways that are important for tumor growth, the inhibition of tumor-induced blood vessel generation (denoted *angiogenesis*), or immunotherapy, which exploits tumor-specific antigens. With the exception of surgical excision of relatively small tumors, treatment strategies employed for the great majority of the patients are currently multimodal. This multimodality and the ongoing development of new drugs or treatment approaches generate a fast growing number of different possible protocols of cancer treatment. Given the limited human and financial resources for clinical trials, optimal protocols cannot be determined empirically; that is, by trial and error alone as is presently the only existing medical paradigm. Rather, a formal method is necessary for a priori suggesting improved drug schedules, according to criteria set by the physician. These criteria may be, for example, life expectancy of the patient, time to reach a specified disease stage, side effects, quality of life, cost of treatment, etc. In this paper, we focus on the use of operations research methods for ameliorating chemotherapy, which is still the most widely applied mode of anticancer therapy.

It may be argued that no open optimization problems exist today in traditional chemotherapy, where one wishes to maximize drug total dose, while keeping toxicity below a certain threshold. This view relies on the naive assumption that today's chemotherapy treatments achieve the maximum efficacy/minimum toxicity requirement. This is not so: chemotherapy still suffers from the inability to control the efficacy/toxicity balance, and as will be mentioned below, mathematical models of the physiological and pathological cellular dynamics have been developed for identifying new strategies for splitting the total drug dose so that toxicity will be reduced without compromising efficacy. However, these strategies rely on mathematical scenarios that are too simple to enable the implementation of specific drug-disease systems. Therefore, improved optimization methods are needed, which can accommodate more complex disease-treatment models. One such method will be developed in this paper. This method, whose protocol space includes mostly new therapy schedules, is to be distinguished from the known methods using expert rules for selecting improved anticancer treatments out of a repertoire of available treatments.

Cancer progression in a patient undergoing chemotherapy is a very complex process, concurrently occurring on many levels of organization: the genomic level, the level of gene expression, that of cell-to-cell signaling, of cell populations, the level of organ (e.g., blood vessels) formation, etc. Each one of these processes itself is very complicated, and the system as a whole is even less tractable, due to the multiple interactions among the processes occurring on the different levels. For this reason, it is impossible to predict, on the basis of biological knowledge and intuition alone, the therapeutic results of changes in treatment schedules. Rather, such predictions must take account of the specific effects of the treatment schedule on the relevant cellular and molecular dynamics of the patient. Calculating these detailed dynamics, one can predict the effects of each individual treatment scenario and, subsequently, employ optimization methods for suggesting improved treatment schedules for the drugs in question.

To date, extensive effort has been invested in the theoretical investigation of cancer chemotherapy control methods. For example, following Cox et al. (1980), Swan (1987) presented two optimal control problems, which differ in the cancer growth function: drug effect is taken as linear in one, and nonlinear (saturated) in the other. The efficiency of a treatment is measured both by the overall toxicity it induces and by the overall number of cancer cells throughout the entire treatment period, and the number of cells is taken in relation to a desired lower bound. An optimal analytic solution is found to both problems in the form of a function of drug dose, tumor size, and time. In another work, Murray (1990) addressed the problem of minimizing the tumor size while limiting toxicity by keeping the host cells population above a given threshold. The optimal solution here is the administration of an initial intensive treatment, thus reducing the number of host cells to the minimum allowed, and consequently, the administering of a drug such that the host cells are always kept at their minimal level.

An optimization problem involving multiple drug chemotherapy is studied in Pereira et al. (1994). Here, the mathematical model describes the kinetics and the dynamics of anticancer drugs as well as the description of cancerous and host cells dynamics. Because an analytical solution is unobtainable in the general case, the optimization procedure involves the execution of an iterative algorithm using Pontryagin's maximum principle. The optimization problem is formulated in a rather conventional way: minimize the cost function, considering the number of surviving tumor cells, as well as minimize the concentration of the administered drugs under the conditions described by the main equation. The iterative algorithm obtains a local minimum of the cost function. The optimization problem thus formulated admits solutions which satisfy the formal requirements but are not sound biologically. Moreover, to obtain a solution, the description of the host and the cancer cell progression is much oversimplified. For example, it is assumed that cellular tissue structure is completely homogeneous, and no interaction exists between tissues.

Recently, Athanassios and Barbolosi (2000) treated cancer chemotherapy as an optimization problem, where tumor and white blood cells' (WBC) responses to chemotherapy have been considered. The optimization problem is formulated as searching a chemotherapy protocol (timing and dose), which minimizes tumor load at the end of the first chemotherapy cycle (subsequent cycles are not considered) and minimize toxicity to the WBC, as measured by their counts in peripheral blood. The model consists of ordinary delay differential equations, where the tumor compartment is controlled by Gompertz-type growth and by an additional cell kill function, depending on the drug pharmacodynamics. The peripheral WBC counts are taken as having constant production and elimination, with a constant delay effect of chemotherapy. The selected, loosely defined constraints on possible protocols are: (1) WBC counts never descending below the level of clinical leukopenia, and (2) the level of peripheral WBC never descending below a certain threshold for a time longer than some constant. The optimization is performed using nonlinear programming and numerical methods. This optimization problem cannot be applied in the clinical setting, as its tractability is based on a gross simplification of the underlying kinetics. As Athanassios and Barbolosi rightly remark, the assumption of constant tumor cell kill, and the exclusion of the multiple feedback effects of the cytokine G-CSF, the main modulator of WBC production, significantly oversimplify the model. In contrast, the consideration of more complex tumor elimination mechanisms and the introduction of the important G-CSF feedback loop, as well as many additional missing factors, while making the model clinically applicable, will probably render it too complex to enable analysis.

The models discussed above fail to retrieve the real benefit of a large class of anticancer drugs, that of cell-cycle phase-specific drugs, namely, drugs that are detrimental to the cell in specific moments of its life cycle. As will be detailed below, these drugs can be administered at times when the tumor is mostly drug sensitive (i.e., when many of the cells are in the drug-susceptible phase of their cycle). To take account of such susceptibility, one must introduce the cell-cycle structure into the mathematical model.

A mathematical model which takes account of cell-cycle dynamics of tumor and host cellular dynamics has suggested that intermittent delivery of cell-cycle phase-specific drugs, at intervals equivalent to the mean cell-cycle time, might minimize harmful toxicity without compromising therapeutic effects on target cells (The Z-Method; Agur 1986, Agur et al. 1988). Subsequently, explicit general formulas have been derived for the growth or decay of cell populations that are subjected to repeated pulse delivery of cell-cycle phase-specific drugs (Cojocaru and Agur 1992), and an algorithm has been developed for calculating the required length of treatment for this protocol (Agur and Dvir 1994). The existence of this *resonance phenomenon* has been further demonstrated for a general class of chemotherapy functions, thus supporting the underlying theory (Webb 1990, Johnson and Webb 1996; see also Dibrov et al. 1985).

Agur (1986), Agur et al. (1988), and Agur and Dvir (1994) considered a restricted case of the above-mentioned cell-cycle dynamic model, assuming that no growth occurs, and that all cells whose vulnerable period coincides with the treatment are eliminated. This applies to both the host and the target cells. In Agur (1986) and Agur et al. (1988), the goal is to minimize the ratio of target/host survival time, whereas in Agur and Dvir (1994), the goal is to totally eliminate the target cells.

The predictions of the Z-method have been verified in experiments in mice bearing lymphoma, treated by repeated pulse delivery of the anticancer drug Ara-C, and by the anti-viral drug AZT. These experiments have shown that when the rhythm of drug delivery roughly coincides with the characteristic marrow cell-cycle time, animals survive and myelotoxicity is significantly reduced. The optimal spacing of repeated treatments was determined by measurements of the kinetics of cell movement through different cell-cycle phases . These experiments showed that it is feasible to control host toxicity by rational drug scheduling (Agur et al. 1991, 1992; Ubezio et al. 1994; Agur et al. 1995).

Only periodic policies were considered in the above-mentioned models. Therefore, treatments were to be given at regular times, $t_0 + il$ for $i = 1, \ldots, n$, and a given time, l, between the onsets of consecutive treatments. It was also assumed that all treatment periods are of the same given length. Concluding from Agur and Dvir (1994), if we choose l to be a multiple of the host cells life cycle, each treatment will strike the host cell and its descendants at the same point of the life cycle, and therefore except for the damage caused to host cells by the first treatment, no further damage will be caused by the following treatments.

Optimal control problems of cell-cycle phase-specific drugs are treated by Swierniak (1995, 1996), paying attention to various degrees of complexity of the cell cycle and of the drug's phase specificity. As was previously suggested, cell-cycle dependent chemotherapy is shown to favor periodic treatment. Here, the problem is approached by minimizing the final count of cancer cells at the end of the treatment or, alternatively, by maximizing the final count of host cells at that time. This area was also reviewed by Swan (1990).

As is shown by the above-mentioned examples, to date, the approach has been mostly theoretical, using mainly control methods where, inevitably, the functions describing the involved dynamics are much simplified—the obvious trade-off being the sacrifice of medical realism for maintaining analytical rigor. In this work, we use operations research techniques for identifying improved drug schedules in any group of patients. Our focus is the general concept of the optimization method, rather than the particular treatment solution. For this reason, we chose to employ the much simplified, but already validated (see above) cell-cycle model, which underlies the Z-method. It should be emphasized that, in principle, the heuristic optimization method put forward below admits dynamic mathematical models of any desired level of complexity.

In our model, we consider two types of cells in the human body. The *host cells* denoted *h-cells*, and the *target cells* or *abnormal cells* denoted *a-cells*, which are, in fact, the tumor. Modeling cell behavior in this work relies on well-established biological research (see, for example, Marieb 2002) which shows that some types of cells have a life cycle that consists of four phases, resulting in the cells' duplication. Each one of these phases is an essential step in the cells' reproduction. Phase-specific drugs, which are used in cancer treatment (see, for example, Elledge 1996, Freaux de Lacroix and Klein 1983) are designed to interfere with the normal process of cell reproduction by preventing the cell from completing a certain phase. Because both normal cells and cancer cells follow this behavior, both types of cells may be damaged while exposed to chemotherapy. Our aim is to reduce the number of target cells, while maintaining a certain level of host cells in the body. Moreover, we assume that the lengths of the cell-cycle phases are deterministic and known, both for host and target cells. Both host and target cells are assumed to be sensitive to the chemotherapeutic agents in only one or two of the cell-cycle phases. These phases are defined here as the *critical phase* of the life cycle. If a cell is exposed to chemotherapy during part of its critical phase, there is a chance that it will be eliminated. Specifically, we assume that during each unit of time in which treatment is given, a fraction of the cells which are in their critical phase will be destroyed. Our aim is to reduce the number of target cells to a certain

fraction of their initial level. However, in order not to cause irreversible damage to the patient's body, we must schedule treatments so that the number of host cells will always remain above a fraction of its initial level. Cells of both types multiply, but not necessarily at the same rate. If the number of target cells is reduced below the desired level, we assume that the remaining target cells will not multiply anymore.

Unlike the studies mentioned previously, we assume that treatments can be of variable length and can be given at any time. A solution, or a treatment protocol, determines those time intervals in which treatment is to be given.

Our model is computationally intractable. We demonstrate this point by proving that even in an extremely simplified special case, it is NP-complete to check whether a patient can be cured. Consequently, we suggest an approach aimed at obtaining good (although not proved to be optimal) solutions. Our approach solves the model by applying local search. There are many local search heuristics that can be used. We used our numerical study to study two not very well-known heuristics, namely, *threshold acceptance* and *old bachelor acceptance*, and compare them to *simulated annealing*, which is a commonly-used heuristic. Our aim is to compare the three methods in terms of quality of the solutions and the computational effort. A necessary step that this comparison requires is the fine tuning of each algorithm's performance by choosing the best parameters for each one. We would also like to learn as much as possible about the performance of these algorithms and about how the performance depends on the various parameters of each algorithm.

To follow the dynamics of cells and their age distribution with or without the chemotherapy treatment, a computer program was written that simulates the cells' conduct. This also enabled us to assess the quality of the solutions tried and suggested by the approximation methods.

For each solution, we view the condition of the patient at a common predetermined time T. In particular, treatment must end by this time. If the patient is cured earlier, then the time until T is used by the body to recover its host cells. The quality of a solution is measured by its fitness. The fitness function assigns a value to each solution. This value is affected by the following factors:

- The number of host cells in the body at T. The fitness increases as this number grows.
- The number of target cells in the body at T. The fitness decreases as this number grows. If the patient is cured, the amount of target cells that remain in the body does not affect the fitness.
- The fitness increases if the patient is cured, and the increase grows as cure is achieved at an earlier stage.
- The fitness decreases if the patient dies as a result of the treatment protocol, and the decrease grows the earlier death occurs.

Even though all solutions that end in the patient's death are practically worthless, we need to be able to compare them. In such cases, the longer the patient lives, the better the solution is considered. The obvious reason is ethical, but another reason for this decision is that solutions that prolong a patient's life can be more easily modified into solutions that keep the patient alive.

From the biological point of view, we attempted to simulate genuine cell behavior as much as possible. The simulation and approximation methods discussed here are very flexible and can easily be modified to suit more strict assumptions.

The results suggest two types of treatment protocols—intensive and nonintensive. The initial impression is that the type of protocol depends on the relative lengths of the host and target life cycles, as well as the lengths of the critical phases. Further study can be made to characterize the biological models that yield the different patterns of treatment protocols—intensive or nonintensive.

The conclusions concerning the three approximation methods are that good results can be achieved by choosing the proper parameters for each of the methods. In threshold acceptance, only one parameter has to be chosen, and one value was found that produced best-found solutions for all problem instances that were studied. Old bachelor acceptance requires three parameters, and combinations of these parameters were found to produce best-found solutions for all instances. However, it was not the same combination for all instances. The computational effort required for achieving good results is much greater in simulated annealing than in the other methods. As for the other two heuristics, it can be seen that a large number of iterations does not guarantee better solution quality. This leads to the conclusion that computer resources would be better used in several short searches with different parameter values, than in one long search. This is particularly relevant for old bachelor acceptance, in which the number of iterations is determined by the user. Comparing the algorithms, it seems that old bachelor acceptance is the least demanding method from the computational effort point of view.

It should be noted that the problem of chemotherapy scheduling is only one of a large set of scheduling problems on which the operations research methods studied in this paper can be applied. The key to applying these methods on other problems lies in defining the proper parameters for the search heuristics.

2. The Model

Let us denote by τ_i and s_i the length of the life cycle and the critical phase, respectively, of i-cells, $i \in \{a, h\}$. At the end of a cell's life cycle, it may produce new cells. The average number of new cells produced by a single i-cell that reaches age τ_i is the growth rate in the i-cells' population, and we denote it by r_i.

If the proportion of target cells is reduced to a fraction β_a of its initial level, then the body is considered cured and the treatment may be stopped. In practice, the proportion of host cells must remain above a fraction β_h of its

initial level during the entire treatment. If the proportion of host cells falls below this level, the patient is considered dead. However, we will allow solutions that do not satisfy this condition to be tested, with the aim of achieving good solutions at a later stage of the process. In our model, the patient can only die as a result of the treatments' toxicity—death is not caused by the tumor. This model assumption would not result in solutions that leave a patient without treatment because growing cancer cells will quickly reduce the fitness to $-\infty$. Hence, solutions that result in a patients' death due to treatment have higher fitness values than solutions that do not treat the patient at all.

For each $t \in [0, T]$, the state of the system is characterized by two "density" functions, $n_i(w, t)$ $i \in \{a, h\}$, that are defined for $w \in [0, \tau_i)$. For $0 \leqslant p \leqslant q \leqslant \tau_i$, the number of i-cells whose age is in the interval $[p, q]$ at time t is $\int_{w=p}^{q} n_i(w, t)\, dw$. In particular, the total number of i-cells at t is $x_i(t) = \int_{w=0}^{\tau_i} n_i(w, t)\, dw$. We will normalize the units by which we count cells so that the initial quantity is $x_i(0) = 1$.

Initially, before treatments start, we assume that the cell ages distribute uniformly along the life cycle, that is, $n_i(w, 0) = 1/\tau_i$, $w \in [0, \tau_i]$. When chemotherapy is applied, the cell age distribution obtains some nonuniform shape, depending on the treatment schedule.

We assume that without treatment, the number of a-cells is doubled during each life cycle, that is, $r_a = 2$. In reality, we may find that the actual growth rate is lower. However, a growth rate of 2 defines the worst-case scenario, and the solutions found for this case still hold when the rate is lower. The host cells' growth rate depends on their number. The growth rate at time t is $r(x_h(t))$, where $r(x)$ is a (nonlinear) decreasing function that tends to 1 as x goes to 1. Therefore, the growth of the host cells slows as they multiply. Note that our model assumes that although influenced by the total number of host cells, their growth rate is independent of the age distribution.

Let us denote by α_i the proportion of i-cells (host or abnormal) from those which are at their critical phase, that are destroyed during one time unit of treatment. (Different rates for host and target cells suit newly developed drugs that are more aggressive to the target cells than the host cells.) A *treatment policy* (or *protocol*) consists of the times at which treatments are given.

Any policy that cures the patient without damaging more host cells than is permitted is a good policy, and can be accepted as a solution of high quality to our problem. However, defining a most desirable policy is more difficult because two factors determine the quality of a treatment—the time of cure, and the number of host cells at time T. The relative importance of each one of these factors must be defined to refine the performance of the algorithm. The fitness function constructed to meet these criteria is

$$\begin{aligned}\text{fitness}(s) = {} & (x_h(T) - \beta_h)(2 + \beta_h - x_h(T)) - x_a(T) \\ & + c_1 I_{\text{alive}} + c_2 I_{\text{cured}} + c_3 I_{\text{alive}} I_{\text{cured}} \\ & + \frac{\text{time\_of\_death}}{K}(1 - I_{\text{alive}}) - \frac{\text{time\_of\_cure}}{K} I_{\text{cured}},\end{aligned}$$

where I_{alive} and I_{cured} are indicator functions stating the patient's condition at the end of the treatment period. Detailed analysis of this fitness function follows in §5.

Our approach is to compute a protocol through numerical computations because theoretical analysis is possible only for very simplified models (Agur and Dvir 1994). To make the process computationally tractable, we will measure time and age by discrete units of a given length. To make computations reasonably quick, we divide the cells' cycle into discrete time units, and assume that the number of cells is constant over this unit. Thus, treatment policy consists of the times $t_1, \ldots, t_m \in [0, T]$, at which treatments are given.

The distribution of the cells' age is generated using the following simulation rule: When no treatment is given, all cells mature by one time unit, and the cells that have reached the end of their life cycle multiply. When chemotherapy is present, all cells mature by one time unit, cells that are in the critical phase are reduced by a given fraction, and the cells that have reached the end of their life cycle multiply.

The growth rates are calculated using the following rules: The target cells double their amount at each cycle, therefore the target growth rate always equals 2. The population of host cells, on the other hand, can never exceed its initial level—there is no uncontrolled growth in the host cells. Their growth rate is assumed to be the highest number not greater than 2 that will keep the total amount of host cells at most 1. This growth rate becomes smaller as the host cells multiply.

3. Interactive Code for Simulating a Treatment Protocol

We constructed an algorithm to simulate the body's response to chemotherapy. It implements cell growth and cell death procedures as described in the previous section. The pre-defined parameters of the program are the lengths of the host and target life cycles, the lengths of their critical phases, and a resolution factor that determines the length of a single time unit. The main part of the program are two procedures that simulate the growth of the cells during treatments and when no treatment is given. The array in which the numbers of cells are kept is updated once per time unit, whether with or without treatment present at that time.

The graphical representation of the cells' status shows us the two "density functions" on the same graph. This means that for every interval (t_0, t_1), the area under the curve between t_0 and t_1 is proportional to the number of cells whose age is in the interval. The initial state is that both types of cells have a total relative number of one, and their ages are uniformly distributed over the life cycle. When treatments are given, and when cells grow, the total number changes, and the graphs show the change in the number of cells in each point of the life cycles, with respect to the initial uniform distribution of the cells.

In the figures, we measure age not from the beginning of the conventional cell cycle, but rather from the beginning of the critical phase. The reason is that such a graph enables us to detect the most favorable time for the next treatment more easily. Let us consider an example that will be discussed in details later in the paper: in this problem, denoted P2, $\tau_a = 28$, $\tau_h = 24$, and $s_a = s_h = 10$, all in units of one hour with $\alpha = 0.05$. We also assume that the host cells' critical phase starts at age 10 hours, and the target cells' critical phase starts at age 16 hours. The treatment plan consists of treatments in times 1–6, 18–29, 42–47, 49–50, and 66–69. Figure 1 (top) shows how one time unit of treatment affects the cell distribution: the first 10 hours of both life cycles that are shown on the graph are the critical phase, and so we observe treatment effect on them—their number is reduced. The actual beginnings of the life cycles occur, in our shifted cycle clock, at time 12 for the target cells and at time 14 for the host cells. We can see that in the beginning of the life cycle, there is one time unit in which cells have begun their growth. The target cells' growth is larger at this time because many host cells are still present in the patient's body and this implies that the host cells grow relatively slowly. The growth happens along an interval which is one hour long because in our implementation, each time unit equals one hour. The target cells show uniform growth, caused by the constant growth rate of 2. The host cells, on the other hand, show an increasing growth, affected by the fact that their overall number decreases during treatment periods.

As more and more treatments are given, both functions lose their initially uniform shape. We have done interpolation in the figures describing the density functions to transform a step function to a continuous function. It can be seen that, in general, this protocol suggests treatments coinciding with many target and few host cells being in (or about to enter) the critical phase.

Figures 1 through 4 give a brief look at the cell age distribution before and after chemotherapy for the best-known solution for P2. In these graphs, the critical phases of both types of cells are the first phases on the scale (age = 0). This solution suggests a somewhat nonintensive treatment plan, and it can be seen that there is an area of unharmed

Figure 1. P2: time = 1 (top); end of first treatment time = 6 (bottom).

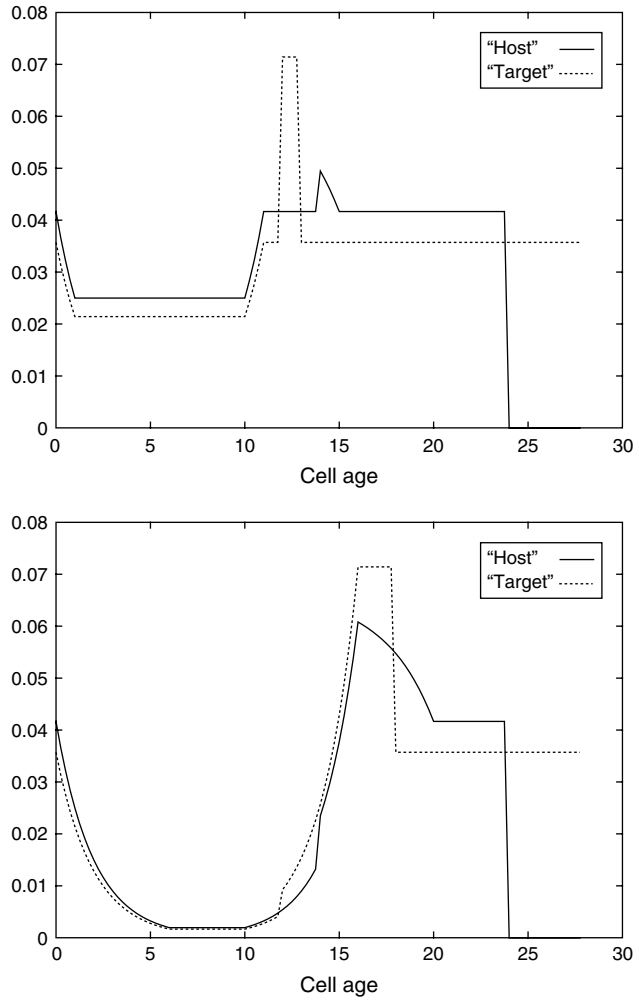


Figure 2. P2: end of first break-time = 17 (top); end of second treatment-time = 29 (bottom).

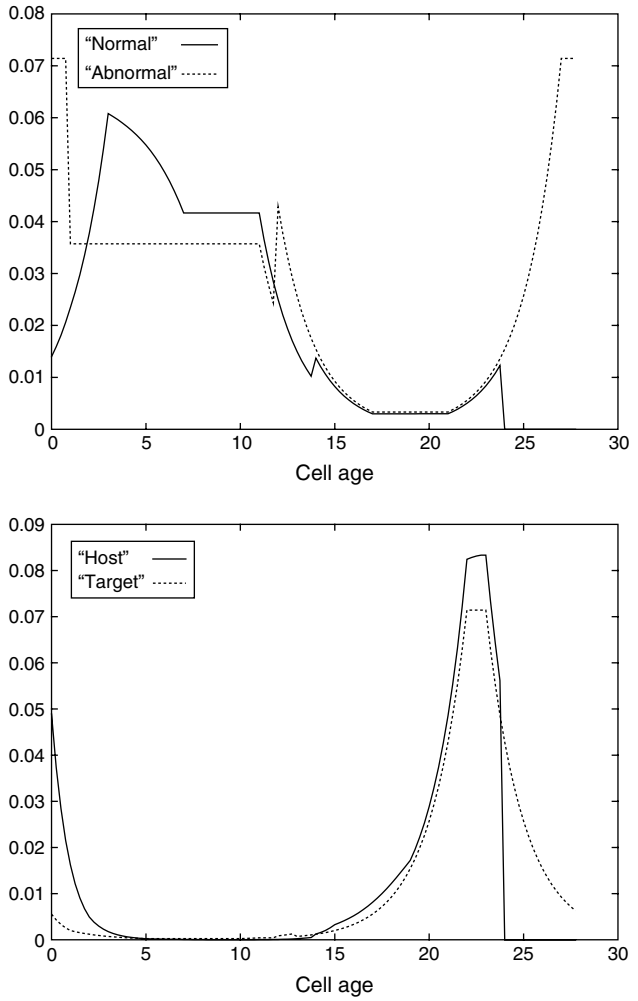


Figure 3. P2: end of second break-time = 41 (top); end of third treatment-time = 47 (bottom).

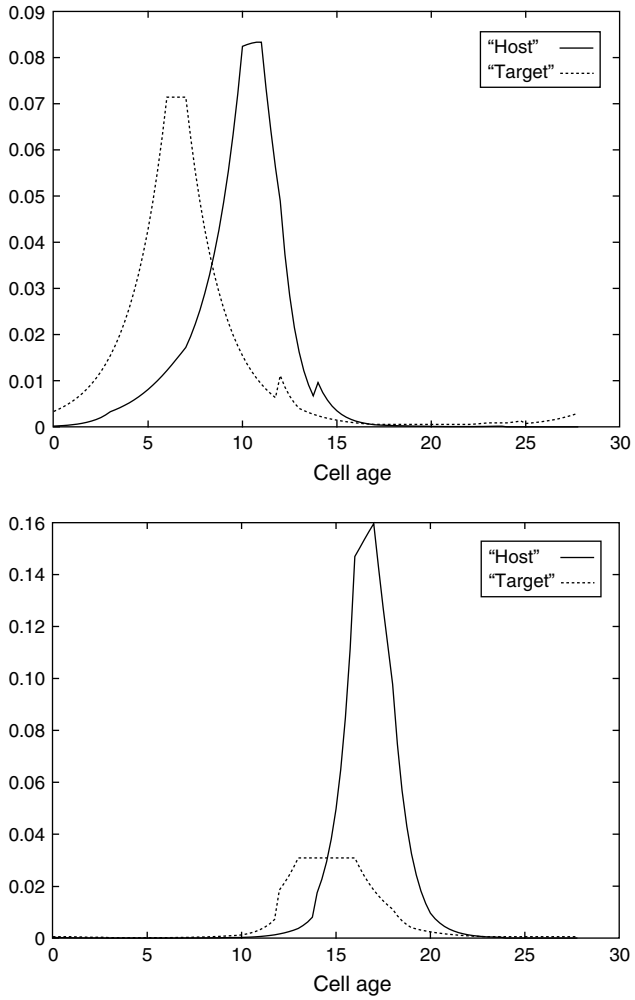
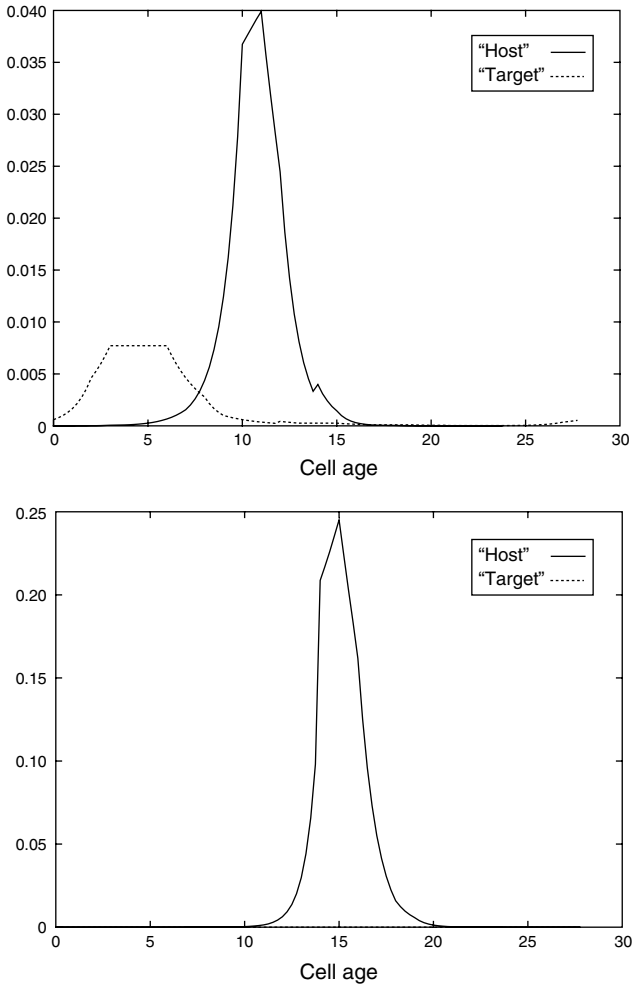


Figure 4. P2: end of fourth break-time = 65 (top); end of last treatment-time = 69 (bottom).



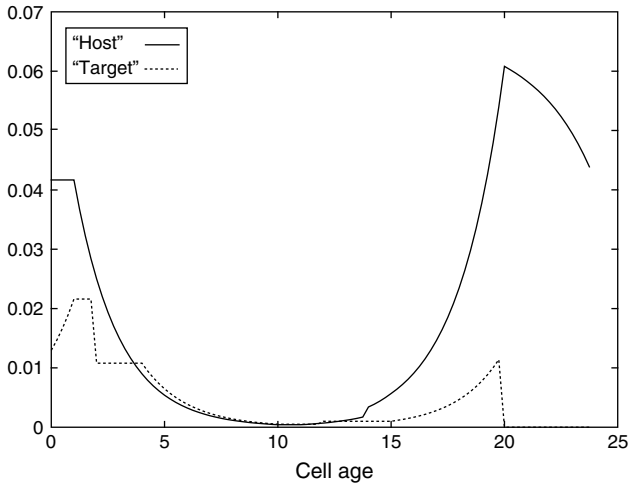
host cells, and chemotherapy is avoided when these cells are in their critical phase. Treatments are given when the critical phase has relatively many target cells and few host cells. The cells that are just about to enter the critical phase are also taken into account: treatments are given if many target and few host cells are about to enter the critical phase, and are stopped when the situation changes. This can be seen, for example, in Figure 2 (bottom): the number of host cells just before the critical phase is large, and the treatment is stopped. In Figure 3 (top), a treatment period is about to start, when the majority of the host cells has just left the critical phase, a large number of target cells is still in it, and very few cells of both types are about to enter the critical phase. At the end of this treatment period (Figure 3 (bottom)), there are hardly any cells of any type in the critical phase, and the number of target cells is considerably reduced in relation to the host cells. Note that cancer cells are not shown in Figure 4 (bottom) because their number is below the threshold β_a.

In another example denoted P5, $\tau_a = 20$, $\tau_h = 24$, $s_a = 14$, and $s_h = 10$, with the same $\alpha = 0.05$. The solution we obtained for P5 demonstrates a different type of a treatment protocol—intensive with a short one hour break. It was also noticed that the number of host cells remained relatively high throughout the entire treatment period. This can be explained by the fact that in P5, $\tau_a < \tau_h$, which is in favor of the host cells. The target cells can be destroyed within one life cycle, without damaging the host cells more than is permitted, and this implies a treatment protocol of one long treatment period. The short break in the middle of the treatment period can be explained by the fact that without it more target cells would be destroyed than is required for curing, so this time unit of treatment can be spared, leaving more host cells at the end of the treatment period. Figure 5 shows the cell age distribution at time $t = 10$, just before all target cells are destroyed.

4. Optimization

The generality of our model makes it computationally very hard. The following theorem demonstrates that even an extremely simple special case of it is NP-hard.

Figure 5. P5: end of first (and only) break-time = 10.



THEOREM 4.1. *Given the distributions of the healthy and tumor cell ages at time* 0 $(n_i(w,0),\ w = 0,\ldots,\tau_i - 1,\ i = a,h)$, *computing whether a patient can be cured is NP-complete even when the growth rates are* 0 *for both cell types and* $\tau_h = s_h = \tau_a = s_a$.

PROOF. We use a reduction from the knapsack problem: Maximize $\sum_{j=1}^{n} b_j x_j$ subject to $\sum_{j=1}^{n} a_j x_j \leqslant B$, where $x_1,\ldots,x_n$ are binary decision variables and $a_j,\ b_j,\ j = 1,\ldots,n$, are given integers. We construct an instance in which the life cycle is n for both normal and abnormal cells, and the critical phase consists of the whole life cycle. The initial distribution of healthy cells is $n_h(i,0) = a_{i+1}$ and that of the tumor cells is $n_a(i,0) = b_{i+1},\ i = 0,\ldots,n-1$. We also set $\beta_h = 1 - B$. Because $\tau_h = \tau_a$, there is a curing schedule if and only if the patient can be cured during a single cycle. Let $x_i = 1$ if treatment is given during the ith interval. The goal is to kill at least a fraction $1 - \beta_a$ of the tumor cells, while killing at most a fraction $1 - \beta_h = B$ of the healthy cells. Thus, the patient can be cured if and only if the solution to the knapsack problem is of value at least $1 - \beta_h$. This problem is well known to be NP-complete. □

Given the computational hardness of the model, we aim at computing good, although not proved to be optimal, solutions. The rest of this section contains brief descriptions of three local search heuristics that participate in our computational study.

Simulated annealing (SA) is a well-known heuristic and we refer the reader to Aarts (1997) and Johnson et al. (1989) for details. In this section, we give brief descriptions of the other two heuristics that we used in our study.

Threshold acceptance (TA) is a deterministic version of SA. The difference between the two heuristics lies in the criterion for making a downhill descent—accepting a solution s for which fitness(s) < fitness(s_0), s_0 being the current best solution. In SA, downhill descent is made with a certain probability that depends on fitness(s) − fitness(s_0) and on the temperature, that is gradually reduced as the SA process continues. In TA, the temperature is replaced by a series of descending thresholds $t_0,\ldots,t_n$. A solution s such that fitness(s) < fitness(s_0) will replace s_0 as the current solution at stage i of the process if fitness(s) > fitness$(s_0) - t_i$. We refer the reader to Dueck and Scheuer (1995) for further details.

The same fitness function and the same neighborhood were used as in SA. The TA parameters were tested by running a series of different instances of the problem, and comparing the performances of the algorithm under different parameter values. The series of thresholds taken was geometrically descending, and several values of a descent rate, noted as the *reduction factor*, were tried. The threshold was reduced after two complete searches of the entire neighborhood, which is similar to the rule used for reducing the temperature in SA. The algorithm terminates when two consecutive thresholds ended with the same fitness value, which is also similar to the SA termination rule.

Old bachelor acceptance (OBA) is a modification of TA, where the threshold does not always decrease. In this method, the threshold depends on the acceptance or rejection of the several most recently tried solutions. The heuristic is described in Hu et al. (1995), and was slightly changed to suit our specific problem—the original algorithm used $T_0 = 0$ as the initial threshold and we used $T_0 > 0$. The reason for this modification is that the first solutions tested always cause an increase in the fitness, therefore lowering the threshold rapidly. When these consecutive improvements stop, many solutions are rejected until the threshold enables another acceptance. To avoid so many rejected solutions, we set the initial threshold a bit higher than in Hu et al. (1995).

Let us now examine this algorithm more closely: OBA uses three parameters to determine thresholds and search termination:

- M—total number of iterations the procedure will perform—that is, the number of solutions that will be tested throughout the search.
- d—number of consecutive solutions that can be rejected before a solution is accepted and would still be considered a fast acceptance.
- Δ—a threshold change factor, used both for increasing and reducing the threshold.

When a solution is accepted, the number of preceding nonaccepted solutions is checked: if this acceptance is not a fast one, i.e., d or more solutions were rejected before the current solution was accepted, then the *fast_acceptances_counter*, which counts the number of consecutive fast acceptances is set to 1, and this implies a small reduction of the threshold in the next iteration. If the acceptance is a fast one, and less than d solutions were rejected between the last acceptance and this one, then the fast acceptances counter is increased by 1, which means a bigger reduction

of the threshold in the next iteration. One should note that by this way of reducing the threshold, the threshold might become negative and demand a strict improvement of the fitness to accept a solution.

As in TA, a solution s_i is accepted (at stage i out of M) if $\text{fitness}(s_i) > \text{fitness}(s_0) - t_i$. The threshold t_i is then increased (if the solution is rejected) or reduced (if the solution is accepted) by a multiple of $\Delta(1 - i/M)$. That means smaller changes in the threshold as the search advances.

Let us now examine the multiplier. When a solution is rejected, the multiplier is the constant $1/d$, and so the threshold is increased by $(\Delta/d)(1 - i/M)$. When a solution is accepted, the multiplier is the number of consecutive fast acceptances that happened prior to this solution being tested. The threshold is then reduced by fast_acceptances_counter $\Delta(1 - i/M)$. That means that acceptance becomes more difficult as many solutions are accepted quickly.

The procedure terminates once M solutions have been tested. It returns the best solution tested, which is not necessarily the last one.

When adapting the algorithm to our needs, the common parameters (fitness function, neighborhood, etc.) were taken to be the same as in SA and TA. The specific parameters that were tested were M, d, and Δ.

5. Testing for Parameters

Fitted for our purposes, the generic algorithms take the following form: Each treatment protocol is represented by a binary string of length T. Each bit in this string is equivalent to one time unit, say one hour, where a zero means that no treatment is given in this hour, and one means that treatment is given. For example, the string 110001 shows that treatment is given for two hours, then no treatment is given for three hours, and again treatment is given for a period of one hour. The length of this string can be determined by the user. Such a string is equivalent to a series of treatments, not necessarily of equal length, with variable length periods of no treatment between them.

The fitness of a solution includes several factors: $x_h(t)$ and $x_a(t)$ denote the relative numbers of host and target (abnormal) cells, respectively, at time t. We measure these numbers as proportions which are taken with respect to the initial level, so that by definition $x_a(0) = x_h(0) = 1$. When a patient is cured, we assume that as a consequence, all the target cells are eliminated. Our discrete representation of the cell age distribution implies that

$$x_h(t) = \sum_{w=0}^{\tau_h - 1} n_h(w, t)$$

and

$$x_a(t) = \begin{cases} \displaystyle\sum_{w=0}^{\tau_a - 1} n_a(w, t), & \displaystyle\sum_{w=0}^{\tau_a - 1} n_a(w, t) \geqslant \beta_a, \\ 0, & \displaystyle\sum_{w=0}^{\tau_a - 1} n_a(w, t) < \beta_a. \end{cases}$$

In addition, let us define two indicators (using obvious notation): I_{cured} and I_{alive}, that indicate the patient's status during the treatment series. I_{cured} indicates that at some point during the treatment series, the number of target cells went under the required threshold, and from this point on we considered the patient cured. As mentioned earlier, we assume that once a patient is cured, the target cells do not multiply anymore. In other words, we consider the tumor totally eliminated. The indicator I_{alive} shows that the patient was alive during the entire treatment period, and that at no time did the number of host cells go under the permitted limit. If the patient is cured, time_of_cure is the time when cure happens, and if the patient dies, time_of_death is the period until the patient's death.

Our aim is to cure the patient as quickly as possible, when a certain level of host cells must be maintained throughout the entire treatment period to keep the patient alive. Until the patient is cured, we attempt to preserve as many host cells as possible. No more treatments need to be given after the patient is cured, and it is assumed that given enough time, the host cells will recover.

As stated, if the patient is cured, we prefer that cure will occur as early as possible. Similarly, in case of the patient's death, we prefer to delay the death as much as we can. These preferences are made under the assumption that solutions that prolong a patient's life can be more easily modified into solutions that keep the patient alive.

All these considerations taken into account, the following fitness function was constructed:

$$\begin{aligned} \text{fitness}(s) = {} & (x_h(T) - \beta_h)(2 + \beta_h - x_h(T)) - x_a(T) \\ & + c_1 I_{\text{alive}} + c_2 I_{\text{cured}} + c_3 I_{\text{alive}} I_{\text{cured}} \\ & + \frac{\text{time\_of\_death}}{K}(1 - I_{\text{alive}}) - \frac{\text{time\_of\_cure}}{K} I_{\text{cured}}. \end{aligned}$$

Let us now examine how this fitness function depends on each one of the required variables. The fitness increases as $x_h(T)$ increases. However, this increase is not linear. The quadratic argument which includes $x_h(T)$ in the function is equal to 0 when $x_h(T) = \beta_h$, and is maximized when $x_h(T) = 1$. Thus, the function changes more rapidly around the critical value of β_h, where host cells are very valuable, than around the maximal value of 1, where host cells can easily be spared. Its derivative changes from 2 when $x_h(T) = \beta_h$ to $2\beta_h$ when $x_h(T) = 1$. Comparing this to the derivative of the argument representing $x_a(T)$ in the fitness function, which always equals 1, we see that many host cells can be sacrificed to eliminate one target cell when $x_h(T)$ is around 1. When $x_h(T)$ is close to β_h, we will sacrifice a host cell only if many target cells will be eliminated with it. This way, host cells affect the fitness more when they are most needed.

A "bonus" of c_1 is given if the patient survives the treatment, and c_2 if the patient is cured. In addition, if

both goals are achieved, an additional bonus of c_3 is given. (Note that cured does not necessarily mean alive. There are two thresholds, one for the host cells and one for abnormal cells, that determine the patient's status: if the host cells are reduced below their threshold, the patient is considered dead. If the abnormal cells are reduced below their threshold, the patient is cured.)

Because the effect of the time_of_death variable should never exceed the effect of any of the indicators that were mentioned earlier in the paper, its contribution to the fitness is normalized such that it will never be greater than 1. This is done by dividing the time_of_death by a constant $K > T$.

Following the same logic, if a patient is cured, we would like the curing to happen as early as possible. In this case, this demand is not just a means of comparing "good" solutions to modify them, but is an actual benefit to the patient. The contribution of the time_of_cure variable is also normalized as mentioned before for the same reasons.

Three types of neighborhoods were considered; each one is generated by allowing a different subset of the following actions.

(1) Replacing one bit of the given solution by the complementary bit (changing 0 to 1 and 1 to 0). This action enables us to change the total number of treatment hours in the solution. This way we can shorten a certain treatment, lengthen it, split it into two treatments, or combine two treatments into one. A new treatment may also be created by splitting a break into two. Similarly, a treatment may also be canceled by combining two breaks into one.

Actions 2 and 3 allow exchanging the last, or first (respectively) bits of two, not necessarily consecutive sequences—one of zeros and the other of ones. This enables us to bring forward or postpone a break between treatments, and to bring forward or postpone an entire set of treatments and breaks, without changing the length of that treatment. While doing so, it is still possible that a treatment will be split into two treatments, or that two treatments will be joined together and become one long treatment. Here are examples of the new treatment plans these actions will produce:

(2) Exchanging the last bits of two not necessarily consecutive sequences:

Bringing forward the first break:

$$1\overrightarrow{100010}111 \to 1000110111.$$

Bringing forward the second break:

$$1\overrightarrow{1000110}111 \to 1000011111.$$

Postponing the second treatment:

$$11000\overrightarrow{10}111 \to 1100001111.$$

(3) Exchanging the first bits of two not necessarily consecutive sequences:

Postponing the first treatment:

$$\overrightarrow{11}00010111 \to 0110010111.$$

Combining the second and third treatments:

$$\overrightarrow{1100010}111 \to 0100011111.$$

Actions 4 and 5 only allow the exchange of the last (or first) bits of consecutive sequences. Referring to the same example, treatment plans produced by these actions will be:

(4) Swapping ends of consecutive sequences:

Bringing forward the first break:

$$1\overrightarrow{100010}111 \to 1000110111.$$

Postponing the second treatment:

$$11000\overrightarrow{10}111 \to 1100001111.$$

(5) Swapping beginnings of consecutive sequences:

Postponing the first treatment:

$$\overrightarrow{11}00010111 \to 0110010111.$$

Combining the first and second treatments:

$$1\overrightarrow{1000110}111 \to 1110000111.$$

Actions 6 and 7 allow the exchange of the last (or first) bit of a sequence with any bit of the sequence that follows, thus allowing to split a treatment or a break in every possible place without changing the overall number of treatment time units.

(6) Swapping the end of a sequence with the consecutive sequence:

Splitting the first treatment:

$$\overrightarrow{11}00010111 \to 1001010111,$$

$$1\overrightarrow{100010}111 \to 1000110111.$$

Postponing the second treatment:

$$11000\overrightarrow{10}111 \to 1100001111.$$

(7) Swapping the beginning of a sequence with the consecutive sequence:

Postponing the first break:

$$1\overrightarrow{100010}111 \to 1110000111.$$

Bringing forward the third treatment:

$$11000\overrightarrow{1011}1 \to 1100011110.$$

Three types of neighborhoods are defined as follows:

- Type A: actions 1, 2, 3—allowing the replacement of a single bit, and swapping first or last bits of any two sequences.
- Type B: actions 1, 4, 5—replacing a single bit, swapping first or last bit of consecutive sequences.
- Type C: actions 1, 6, 7—replacing a single bit, swapping the first or last bits of a sequence with any bit of the sequence that follows.

The relations between the neighborhoods are such that both neighborhoods A and C contain neighborhood B, but no such relation exists between neighborhoods A and C. These neighborhoods were tested for the quality of solutions they produce. The results of these tests appear later in the paper.

The solutions produced by the three different actions are searched in the order that they appear in the neighborhoods' definition. That is, action 1 is performed and its results

are searched first, followed by the two other actions of which the neighborhood consists. Each action's results are searched until a pre-defined number of consecutive, non-improving solutions have been tested. In this context, an improving solution is one that has a higher fitness than the current best solution. The order of search within each action's results is cyclic, and when an improvement occurs the search will continue from the point of the last change. For example, if action 1 of the neighborhood is performed, and changing the third bit proved to be successful, then the next try will be to change the fourth bit of the new solution. The same principle is followed when searching the results of the other two actions of the neighborhood.

The initial solution can be chosen randomly or deterministically. A few ways of choosing an initial solution were tried, and the results appear later in the paper.

A series of tests was made to determine some of the parameters, joined for all approximation methods. These parameters are:

- The coefficients in the fitness function.
- The selection of an initial solution.
- The necessity of each of action in producing a neighborhood.
- The type of neighborhood to be used.

Some of these parameters were tested on small size instances, for which both the solutions quality and algorithm run times were compared. The tests produced the following results.

The indicator coefficients. The coefficients c_1, c_2, and c_3 that form the cure and survival bonuses in the fitness function ($c_1 I_{\text{alive}} + c_2 I_{\text{cured}} + c_3 I_{\text{alive}} I_{\text{cured}}$) were given several combinations of values, differing in the coefficients' absolute and relative sizes. The tests showed that c_3 is redundant—no bonus needs to be given for curing and survival, in addition to the high bonuses already given for each achievement separately.

The tests show that the values of c_1 and c_2 should be set so that their contribution to the fitness function will exceed the contribution of the number of cells. The bonus for survival should be greater than the bonus for curing, and again, the difference between the bonuses should exceed the contribution of the number of cells. The precise values of c_1 and c_2 are insignificant.

Choosing an initial solution. Both random and deterministic solutions were tried. The random starts were characterized by the probability that a certain bit of the initial solution would indicate a treatment. It seems that the initial solution may effect the quality of a solution, but the effect was inconsistent: no certain probability gave good results for all (or most of) the instances. The deterministic initial solutions, on the other hand, performed better: the options tried were an initial solution that was all "treatment," one that was entirely "no treatment," and a mixed solution. The mixed solution started with one time unit of treatment that had to be given because postponing the first treatment will damage the fitness, and the rest of the solution was "no treatment" bits. The logic of this choice is that the first treatment will be constructed by the neighborhood search at the very beginning of the search. The mixed initial solution indeed gave the best results of all. It should be noted that the combination of the initial solution and the relatively high value of c_1 are equivalent to a constraint that the patient must always be kept alive. Because the initial solution has almost no treatments, the patient is alive in the beginning, and will remain so because any solution that violates this constraint will cause a severe loss in the fitness function.

The fitness function can be now be written as

$$\text{fitness}(s) = (x_h(T) - \beta_h)(2 + \beta_h - x_h(T)) - x_a(T) + c_2 I_{\text{cured}} - \frac{\text{time\_of\_cure}}{K} I_{\text{cured}},$$

with $I_{\text{alive}} = 1$ as a constraint.

The three types of the neighborhood. All actions proved to be necessary, and lesser quality results were reached if any one of them was missing. In addition, the neighborhood type that gave the best results was type C, in which beginnings and ends of sequences were swapped with each bit of the consecutive sequence. When neighborhoods A and B were compared, it was seen that swapping between consecutive sequences was enough—further swapping was time consuming and did not produce better results. A comparison was then made between neighborhoods B and C, using TA on five selected instances with different reduction factors. The results of this comparison can be seen in Table 3 in the appendix, and the five instances will be described later in the paper. In this table and throughout the rest of the paper, the best solution for each instance is in bold type. It should be noted that changes in the fitness values of each instance may seem relatively small. This is because the fitness is increased by large constants when curing or survival are achieved, and does not suggest that the differences in the solutions quality are minor. These results show that neighborhood C produced better results with the same computational effort and for all reduction factors tested, so it was used as the neighborhood for all three algorithms.

6. Comparing the Algorithms

The three approximation methods were tried on several instances of the problem. The common parameters were set to the values as given in the previous section, and this set of instances was used to test the parameters that are algorithm specific. The parameters characterizing the host cells, τ_h and s_h, remained the same in all of these instance, whereas the target cells parameters τ_a and s_a varied. Out of the instances tried, a smaller set of five instances was chosen for further study, and the results shown in this paper relate to them. The reason for choosing these particular instances was that they make a fine representation of a variety of

Table 1. The five instances used for comparing the approximation methods.

Instance	τ_h	s_h	τ_a	s_a
P1	24	10	16	10
P2	24	10	28	10
P3	24	10	16	6
P4	24	10	28	6
P5	24	10	20	14

solutions: some of these solutions suggest a single long treatment period, or two long treatments separated by one time unit of no treatment; others suggest a few shorter treatments, separated by longer breaks. There is also one instance in which the patient was not cured. The parameters characterizing the five chosen instances are shown in Table 1. Throughout this section, those instances will be referred to as P1, ..., P5.

A comparison was made both between the methods and between different parameterization of each method.

A crucial parameter of both SA and TA is the reduction factor. Higher reduction factors mean slower reduction rate, which requires higher computational effort. SA requires slow reduction of the temperature, as stated in Aarts et al. (1997). The SA reduction factor was set to be 0.95. As will be seen later in this section, the computational effort involved in this procedure prevented further attempts to optimize the SA parameters. Several values, ranging from 0.05 to 0.85, were tried for TA, and the results show that *high reduction factors (which cause a slow descent in the threshold) do not always produce better solutions than moderate or small reduction factors*. This is as opposed to SA, where a slower descent in the temperature leads to a better performance of the algorithm. The best results were produced by reduction factors of about 0.10–0.25, which required little computational effort. A reduction factor of 0.85 did not perform as well, and higher reduction factors that were tried gave even worse results. A comparison between the reduction factors tested for TA is shown in Table 4 in the appendix. Fitness values above 6 mean that that patient is cured.

To further investigate the dependence of the solution on the reduction factor, we used TA with a series of close reduction factors ranging from 0.05 to 0.95 on P2. The results, shown in Table 5 in the appendix, support the claim that a high reduction factor does not guarantee the best results. Best results were obtained for reduction factors of about 0.25. These results show that no monotonic relation exists between the reduction factor and the fitness. It appears that the best approach for identifying a desirable solution might be to try several small reduction factors in the TA algorithm, and choose the best of the several solutions generated.

In OBA, three parameters should be determined: M, d, and Δ. The initial values for d and Δ were chosen by trial and error, and by following the parameters used in Hu et al. (1995). The value of M was taken to be similar to the number of iteration needed in TA. To see if any rule can be learned about the dependence of the best solution found on M, d, and Δ, after choosing the initial ranges, P2 was tested with a "grid" of values for the three parameters. This grid seems to be a good representation of the ranges of parameters that were found suitable for solving this problem. The results of these tests are given in Tables 6 and 7 in the appendix. Both tables show the same results, where Table 6 is arranged by values of M and d, and Table 7 is arranged by values of Δ for convenience. These results show that holding d and Δ constant, higher values of M are not necessarily associated with better solutions, and that changes in Δ are not necessarily associated with changes in solution values. Thus, it appears that small values of M should not be used with high values of d.

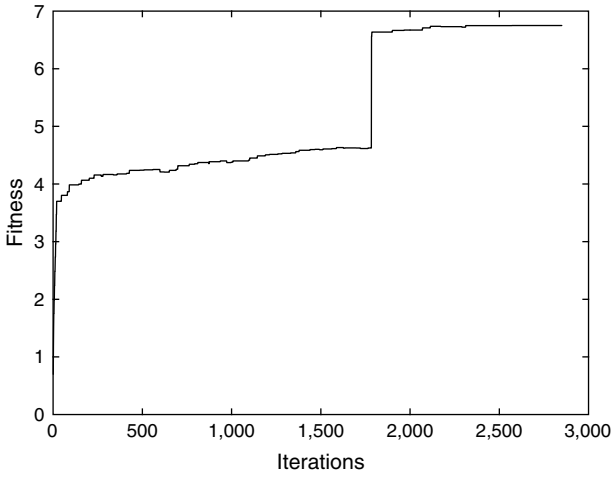
The best results for $M = 3{,}000$ were produced by $\Delta = 0.08$, and for $M = 2{,}000$ by $\Delta = 0.06$ (these solutions are not necessarily the overall best solutions, but are the best solutions for these values of M). The conclusion from these results is that it is better to use small values of M with several combinations of d and Δ than to use a high value of M once.

Several sets of parameters were tried on all five instances, resulting in similar conclusions. It can be seen in Table 8 in the appendix that big M values do not necessarily guarantee good solutions. An exceptionally high value of $M = 6{,}000$ was also taken in two sets of tests, and did not prove to perform better than the smaller M values. Furthermore, solutions whose fitness is near the best achieved so far can be obtained by using small M values, such as $M = 1{,}500$.

Table 2. The comparison of the approximation method.

	SA		TA		OBA	
Instance	Fitness	Iterations	Fitness	Iterations	3,500 Iterations	1,500 Iterations
P1	**7.588316**	14,369	**7.588316**	2,630	**7.588316**	7.582802
P2	6.746259	22,958	**6.749344**	4,630	6.733498	6.735837
P3	7.021295	24,197	**7.204986**	3,872	**7.204986**	7.199592
P4	**4.467657**	15,688	**4.467657**	2,963	**4.467657**	**4.467657**
P5	**7.620466**	15,357	**7.620466**	3,005	7.615119	7.614556

Figure 6. P2: the fitness curve as a function of time for OBA.



When considering the parameterization that gave the best results in each method, one can conclude that SA yields results that are not better than the other two methods, but the computational effort involved in reaching the solution (measured by the number of iterations used) was much higher. For this reason, no further attempts to optimize the problem by SA were made. TA gave good results using relatively small computational effort, and it should be noted that OBA came close to the best-found results using even less iterations than TA. The comparison between the best solutions is shown in Table 2.

In Figures 6 and 7, we can see how the fitness changes while the algorithms proceed. Both figures show the process of approximating P2, and both reached the highest fitness known for this instance. We can see a trend of descent in the beginning of TA, and sharp up and down moves.

Figure 7. P2: the fitness curve as a function of time for TA.

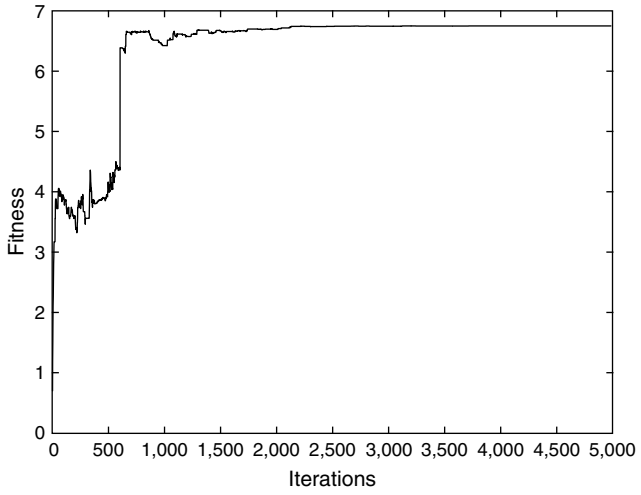
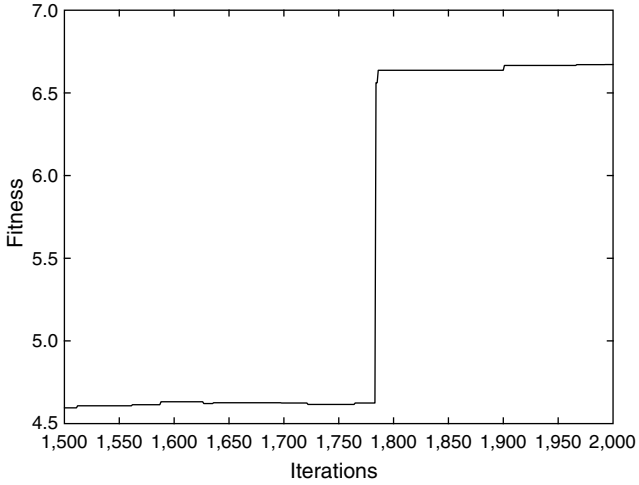


Figure 8. P2: the fitness curve in iterations 1,500–2,000 for OBA.



The fitness stabilizes when it reaches the high values. In OBA, we see small descents in the general trend of ascent. The beginning of the process shows an almost continuous ascent because the first moves are almost always improving moves. It can also be seen that OBA shows a more stable pattern, both in small fitness values (before curing) and in high fitness values (after curing). Curing solutions are reached later in the process in OBA than in TA. Note that the jump in the graphs shows the point of cure.

For a more detailed look, Figures 8 and 9 show the fitness in iterations 500 through 1,000 in TA, and iterations 1,500 through 2,000 in OBA. These stages were chosen because they are around the point of curing. Small increases and decreases in the values of the fitness function, as well as the general trend of an increase in the fitness function, may be discerned in Figures 8 and 9.

Figure 9. P2: the fitness curve in iterations 500–1,000 for TA.

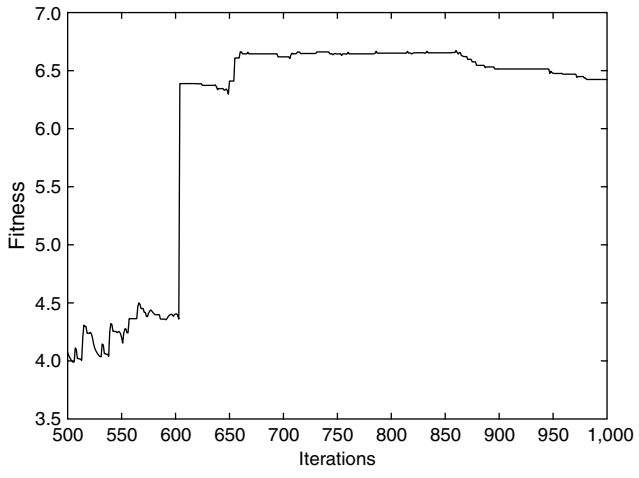
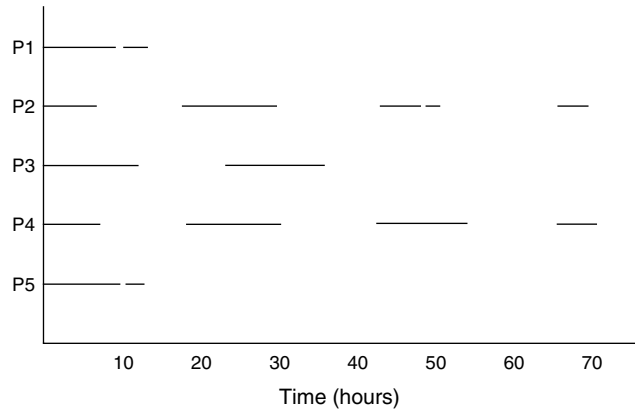


Figure 10. The best treatment protocols of the five problems.



7. Discussion

This paper addresses the problem of adapting OR heuristics to optimize chemotherapy scheduling, with the particular aim of eliminating the cancer cells while maintaining a sufficiently high level of healthy cells. This problem was defined as an optimization problem to which the possible solutions are scheduling plans represented by strings of zeros (no treatment) and ones (treatment). The possible schedules are tested by three local search-based heuristics to find a solution that will locally optimize the fitness function. The comparison between the three approximation methods shows that the three are competitive but the computational effort is much higher in SA than in the other two methods. All three methods produced solutions of similar quality, and therefore the choice among them should be done according to their computational efficiency. When optimizing by TA, we could see that a slow reduction of the threshold does not necessarily produce better solutions. Similarly, for OBA, we could find combinations of parameters that gave good results for almost any value of M. We found that several shorter TA or OBA runs that are executed with different sets of parameters give better results than one long run. The alternative of trying several initial solutions is not suitable for our purpose.

If we examine the types of best solutions the algorithms reached for these problems, they could be roughly divided into two categories: intensive treatment and nonintensive treatment. Intensive regimens are more easy to detect. The solutions to all the problems are shown in Figure 10, where time is measured in hours and the line marks a treatment period.

A central question for which we only have a partial answer is: What problem characteristics determine intensive versus nonintensive regimens? It seems that the type of treatment protocol depends on the relation between τ_a and τ_h, and on the relation between s_a and s_h. The situation favorable to the host cells is when the host cells have a long life cycle and a short critical phase. This situation may imply an exhaustive treatment.

The same problems may be studied under different model assumptions. The algorithms and implementations that were reviewed in this paper are very flexible, and can be very easily adapted to suit other assumptions, such as:

- A different growth function to the host (or target) cells. This can be very easily implemented, as the model refers to a generic growth function and does not depend on a particular growth function.
- A treatment effect that decays with time. In this case, the number of cells at each time will be calculated as a function of the overall treatment length, and not after each time unit individually.
- Stochastic life cycle lengths. The way to implement this modification is by taking a range of life cycle lengths and dividing the multiplying cells among this range. For example, we may have 20% of the cells multiplying at time 18 (and 80% reaching age 19), then 30% multiplying at age 19 (and 50% reaching age 20), and so on.
- Drugs that affect more than one phase of the life cycle. This modification might be needed to simulate treatments in which a cocktail of drugs is given, each drug affecting a different phase of the life cycle. The implementation is by defining the critical phase as a union of the two sensitive phases of the life cycle.
- Radiotherapy. Sophisticated algorithms have been developed for optimizing radiotherapy planning in each individual dosing. Here, the goal is to improve dose distribution homogeneity within the whole treated organ (Mihai 2005), or to find a compromise between the contradicting goals of delivering a sufficiently high dose to the target volume while widely sparing critical structures (Scherrer 2005). The search for such a compromise requires the computation of several plans, which mathematically means solving several optimization problems. In the case of intensity modulated radiotherapy (IMRT), these problems are large scale, hence the accumulated computational expense is very high. Mathematical models for ionizing radiation therapy, applied to multicellular populations whose cells have time-dependent radio sensitivity have been studied (see, for example, Chen et al. 1995 and Hlatky et al. 1994). However, methods for optimizing multiple dosing schedules in radiotherapy, applied over extended periods of time, are yet to be developed. An approach similar to the one described above with respect to cell-cycle specific chemotherapy can be adopted with respect to optimal regimens of ionizing radiation.

Our work concerns the selection of an optimal drug treatment by applying a new heuristic optimization method to the system of biomathematical models representing a cancer patient. This method is more elaborate and more realistic than the relatively simple one of maximizing efficacy-toxicity ratio, suggested in Agur (1986) and Agur et al. (1992).

Currently, drugs are developed to maximize the ratio of efficacy to toxicity, with the objective of improving over a "gold standard" set by the best drug in the market. The evaluation of a new drug is done by trial-and-error clinical testing and a retrospective evaluation of its efficacy and toxicity. Drug protocols that are authorized to be used by the treating physicians are those which are shown by the drug developer to satisfy the criteria of better efficacy and less or equal toxicity than the gold standard. The heuristics proposed here can be further refined and used in conjunction with clinically validated detailed mathematical models that accurately simulate the dynamics of key biological and pathological processes in a patient undergoing specific drug therapy, such as those in Skomorovski et al. (2003). The method presented in this paper will hopefully help to replace the prevailing trial-and-error paradigm by predictions-directed trials.

In this paper, we have introduced a general approach for determining chemotherapy schedules, which can satisfy a realistically complex medical optimization problem. Elaborate mathematical models of the pathology and physiology at hand have been developed to enable precise quantitative predictions and, hence, to implement this approach in the clinic. A first step in this direction was carried out and the realism of these models has been verified in the pre-clinical setting (e.g., Arakelyan et al. 2002, 2005; Skomorovski et al. 2003; Vainstein et al. 2005). Subsequently, it is suggested that replacement of a conventional breast cancer treatment schedule, which involves 100 mg/m^2 of Taxotere applied every three weeks, by a weekly dosing of 30 mg/m^2 of Taxotere, can accelerate tumor shrinkage and significantly reduce its chemotherapy-induced neutropenia in specific patients (Arakelyan et al.).

Appendix

Table 3. Comparison between neighborhood types B and C.

Neighborhood	B		C		B		C	
Instance	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration
	Reduction factor = 0.25				Reduction factor = 0.50			
P1	7.582802	3,120	**7.588316**	2,630	**7.588316**	3,582	**7.588316**	3,886
P2	6.739438	4,411	**6.749344**	4,630	**6.749344**	4,981	**6.749344**	4,984
P3	6.965311	1,882	**7.204986**	3,872	7.007004	5,970	7.204986	5,539
P4	4.460903	3,295	**4.467657**	2,963	**4.467657**	4,137	4.188200	3,297
P5	**7.620466**	2,667	**7.620466**	3,005	7.302219	4,845	**7.620466**	3,460
	Reduction factor = 0.70				Reduction factor = 0.85			
P1	**7.588316**	6,209	**7.588316**	6,073	7.582802	7,578	**7.588316**	10,307
P2	6.738903	4,951	6.716496	4,912	6.748282	5,672	6.715153	6,758
P3	7.204418	3,735	7.080932	9,020	7.204080	6,450	7.133704	4,342
P4	**4.467657**	5,411	**4.467657**	4,893	4.408935	2,542	**4.467657**	8,685
P5	**7.620466**	5,803	**7.620466**	5,590	**7.620466**	12,009	**7.620466**	12,466

Table 4. TA—Comparison of the reduction factors.

	Reduction factor							
	0.05		0.10		0.15		0.25	
Instance	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration
P1	**7.588316**	2,152	**7.588316**	2,204	**7.588316**	2,206	**7.588316**	2,630
P2	6.739438	3,231	6.737697	3,370	**6.749344**	4,148	**6.749344**	4,630
P3	**7.204986**	2,659	**7.204986**	2,973	**7.204986**	3,030	**7.204986**	3,872
P4	4.460903	2,468	**4.467657**	2,581	**4.467657**	2,737	**4.467657**	2,963
P5	**7.620466**	2,177	**7.620466**	2,151	**7.620466**	3,005	**7.620466**	12,466
	0.35		0.50		0.70		0.85	
	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration	Fitness	Iteration
P1	**7.588316**	3,103	**7.588316**	3,886	**7.588316**	6,073	**7.588316**	10,307
P2	6.736225	3,669	**6.749344**	4,984	6.716496	4,912	6.715153	6,758
P3	7.080932	4,334	**7.204986**	5,539	7.080932	9,020	7.133704	4,342
P4	**4.467657**	3,020	4.188200	3,297	**4.467657**	4,893	**4.467657**	8,685
P5	**7.620466**	3,083	**7.620466**	3,460	**7.620466**	5,590	**7.620466**	12,466

Table 5. TA results for several reduction factors on P2.

Reduction factor	Fitness	Iterations
0.05	6.739438	3,231
0.10	6.737697	3,370
0.15	**6.749344**	4,148
0.20	6.736225	4,408
0.25	**6.749344**	4,630
0.30	**6.749344**	5,402
0.35	6.736225	3,669
0.40	**6.749344**	4,880
0.45	6.723167	5,700
0.50	**6.749344**	4,984
0.55	6.724916	3,589
0.60	6.722980	3,930
0.65	6.726003	5,694
0.70	6.716496	4,912
0.75	6.739790	4,406
0.80	6.726003	9,154
0.85	6.715153	6,758
0.90	6.749257	16,532
0.95	6.661870	5,053

Table 6. Testing for the effect of M and Δ on the fitness.

d	Δ	$M = 1{,}000$	$M = 1{,}500$	$M = 2{,}000$	$M = 2{,}500$	$M = 3{,}000$	$M = 3{,}500$	$M = 4{,}000$	$M = 4{,}500$
12	0.06	6.735475	6.698936	6.723886	6.722915	6.723464	6.724006	6.748339	6.722383
12	0.08	6.709258	6.731380	6.745951	6.730176	6.749314	6.724008	6.748559	6.731344
12	0.10	6.736221	6.736100	6.737187	6.745504	6.736045	6.749315	6.745971	6.737512
16	0.06	4.642803	6.749279	6.735410	6.738925	6.734566	6.748559	6.736127	6.737365
16	0.08	6.688278	6.734002	6.659443	6.748367	6.737780	6.746024	6.738289	6.745430
16	0.10	6.722203	6.619663	6.738241	6.739790	6.715841	6.748017	6.745286	6.747786
20	0.06	6.716185	6.668162	6.736395	6.715796	6.722136	**6.749344**	6.721400	6.732031
20	0.08	6.674582	6.723643	6.615793	6.749257	6.745426	6.733498	6.749257	6.748339
20	0.10	4.643348	6.652962	6.730593	6.734861	6.738684	6.746253	6.742629	6.723137
24	0.06	4.613020	6.676059	6.748357	6.749324	6.738931	6.739438	6.723239	6.748394
24	0.08	4.541359	6.735941	6.618395	6.737556	**6.749344**	6.728144	6.735289	6.736157
24	0.10	4.623247	4.661262	6.735617	**6.749344**	6.736157	6.704856	6.723325	6.737697

Table 7. Testing for the effect of d on the fitness.

d	Δ	$M = 1{,}000$	$M = 1{,}500$	$M = 2{,}000$	$M = 2{,}500$	$M = 3{,}000$	$M = 3{,}500$	$M = 4{,}000$	$M = 4{,}500$
12	0.06	6.735475	6.698936	6.723886	6.722915	6.723464	6.724006	6.748339	6.722383
16	0.06	4.642803	6.749279	6.735410	6.738925	6.734566	6.748559	6.736127	6.737365
20	0.06	6.716185	6.668162	6.736395	6.715796	6.722136	**6.749344**	6.721400	6.732031
24	0.06	4.613020	6.676059	6.748357	6.749324	6.738931	6.739438	6.723239	6.748394
12	0.08	6.709258	6.731380	6.745951	6.730176	6.749314	6.724008	6.748559	6.731344
16	0.08	6.688278	6.734002	6.659443	6.748367	6.737780	6.746024	6.738289	6.745430
20	0.08	6.674582	6.723643	6.615793	6.749257	6.745426	6.733498	6.749257	6.748339
24	0.08	4.541359	6.735941	6.618395	6.737556	**6.749344**	6.728144	6.735289	6.736157
12	0.10	6.736221	6.736100	6.737187	6.745504	6.736045	6.749315	6.745971	6.737512
16	0.10	6.722203	6.619663	6.738241	6.739790	6.715841	6.748017	6.745286	6.747786
20	0.10	4.643348	6.652962	6.730593	6.734861	6.738684	6.746253	6.742629	6.723137
24	0.10	4.623247	4.661262	6.735617	**6.749344**	6.736157	6.704856	6.723325	6.737697

Table 8. Comparison of the OBA parameters.

M	1,500	1,500	1,500	1,500	2,000	2,000	2,000	2,000
d	6	6	10	10	7	7	12	12
Δ	0.06	0.08	0.06	0.08	0.06	0.08	0.06	0.08
P1	7.582802	7.582802	7.582802	7.582802	7.582802	7.582802	7.582802	7.582802
P2	6.744762	6.694140	6.735837	6.734866	6.699061	6.748134	6.723886	6.745951
P3	7.187378	7.173682	7.199592	7.162495	7.175924	7.171024	7.168155	7.172284
P4	4.458408	4.446000	**4.467657**	4.461132	4.464170	4.440691	**4.467657**	**4.467657**
P5	7.611470	7.601449	7.614556	7.597960	7.604000	7.615119	**7.620466**	7.608343
M	3,000	3,000	3,000	3,000	3,500	3,500	3,500	3,500
d	10	10	12	12	12	12	20	20
Δ	0.06	0.09	0.06	0.09	0.08	0.1	0.08	0.1
P1	**7.588316**	**7.588316**	7.582802	7.852802	7.582802	7.583802	**7.588316**	7.852802
P2	6.726542	6.746068	6.735837	6.737025	6.724008	6.749315	6.733498	6.724008
P3	7.168155	7.172410	7.199592	7.204080	7.204720	7.202552	**7.204986**	7.204720
P4	4.464170	4.463534	4.461972	**4.467657**	**4.467657**	**4.467657**	**4.467657**	**4.467657**
P5	7.615999	7.611483	**7.620466**	7.608449	7.610704	7.611151	7.615119	7.610704
M	3,500	4,000	4,000	4,000	4,000	6,000	6,000	
d	20	12	12	20	20	12	20	
Δ	0.06	0.06	0.08	0.06	0.08	0.06	0.08	
P1	7.584845	7.582802	7.582802	**7.588316**	7.587222	7.582802	7.587222	
P2	**6.749344**	6.748339	6.748559	6.721400	6.749257	6.732992	6.748282	
P3	7.203444	**7.204986**	7.202552	7.200144	7.203189	7.203189	7.201053	
P4	**4.467657**	**4.467657**	**4.467657**	**4.467657**	**4.467657**	**4.467657**	**4.467657**	
P5	**7.620466**	7.609602	7.612834	**7.620466**	**7.620466**	**7.620466**	7.618140	

References

Aarts, E. H. L., J. H. M. Korst, P. J. M. van Laarhoven. 1997. Simulated annealing. E. H. L. Aarts, J. K. Lenstra, eds. *Local Search in Combinatorial Optimization*. John Wiley and Sons, New York, 91–120.

Agur, Z. 1986. The effect of drug schedule on responsiveness to chemotherapy. *Ann. Acad. New York Sci.* **504** 274–277.

Agur, Z., Y. Dvir. 1994. Use of knowledge on $\{\phi_n\}$ series for predicting optimal chemotherapy treatment. *Random and Comput. Dynam.* **2** 279–286.

Agur, Z., R. Arnon, B. Schechter. 1988. Reduction of cytotoxicity to normal tissues by new regimens of phase-specific drugs. *Math. Biosci.* **92** 1–15.

Agur, Z., R. Arnon, B. Schechter. 1992. Effect of the dosing interval on survival and myelotoxicity in mice treated by Cytosine arabinoside. *Eur. J. Cancer* **28A**(6/7) 1085–1090.

Agur, Z., R. Arnon, B. Sandak, B. Schechter. 1991. Zidovudine toxicity to murine bone marrow may be affected by the exact frequency of drug administration. *Experimental Hematol.* **19** 364–368.

Agur, Z., G. Tagliabue, B. Schechter, P. Ubezio. 1995. AZT effect on the bone marrow—A new perspective on the Concorde trials. *J. Biol. Sys.* **3** 241–251.

Arakelyan, L., Y. Merbl, Z. Agur. 2005. Vessel maturation effects on tumor growth: Validation of a computer model in implanted human ovarian carcinoma spheroids. *Eur. J. Cancer* **41** 159–167.

Arakelyan, L., V. Vainstain, Z. Agur. 2002. A computer algorithm describing angiogenesis and vessel maturation and its use for studying the effects of anti-angiogenic and anti-maturation therapy on vascular tumor growth. *Angiogenesis* **5** 203–214.

Arakelyan, L., N. Dahan, A. Ianovsky, Z. Agur. A new method for optimizing chemotherapy treatment—Prospective validation in breast cancer patients. In preparation.

Athanassios, I., D. Barbolosi. 2000. Optimizing drug regimens in cancer chemotherapy by an efficacy-toxicity mathematical model. *Comput. Biomedical Res.* **33** 211–226.

Chen, P. L., D. J. Brenner, R. K. Sachs. 1995. Ionizing radiation damage to cells: Effects of cell-cycle redistribution. *Math. Biosci.* **126** 147–170.

Cojocaru, L., Z. Agur. 1992. Theoretical analysis of interval drug dosing for cell-cycle-phase-specific drugs. *Math. Biosci.* **109** 85–97.

Cox, E. B., M. A. Woodbury, L. E. Myers. 1980. A new model for tumor growth analysis based on a postulated inhibitory substance. *Comput. Biomedical Res.* **13** 437–445.

Dibrov, B., A. Zhabotinsky, Y. Neyfakh, M. Orlova, L. Churikova. 1985. Mathematical model of cancer chemotherapy. Periodic schedules of phase-specific cytotoxic-agent administration increasing to selectivity of therapy. *Math. Biosci.* **73** 1–31.

Dueck, G., T. Scheuer. 1995. Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *J. Comput. Phys.* **90** 161–175.

Elledge, S. J. 1996. Cell cycle checkpoints: Preventing an identity crisis. *Science* **274** 1664–1672.

Freaux de Lacroix, W., M. Klein. 1983. Comparative investigation on the effect of the phase specific drug vincristine (VCR) on the proliferation of a solid experimental tumor. *J. Cancer Res. Clinical Oncology* **106** 187–191.

Hlatky, L. R., P. Hahnfeldt, R. K. Sachs. 1994. Influence of time-dependent stochastic heterogeneity on the radiation response of a cell population. *Math. Biosci.* **122** 201–220.

Hu, T. C., A. B. Kahng, C. W. A. Tsao. 1995. Old bachelor acceptance: A new class of non-monotone threshold accepting methods. *ORSA J. Comput.* **7** 417–425.

Johnson, D. S., C. R. Aragon, L. A. McGeoch, C. Schevon. 1989. Optimization by simulated annealing: An experimental evaluation. Part 1, graph partitioning. *Oper. Res.* **37** 865–892.

Johnson, M., G. F. Webb. 1996. Resonances in age structured cell population models of periodic chemotherapy. *Internat. J. Appl. Sci. Comput.* **3** 57–67.

Malinen, M., T. Huttunen, J. P. Kaipio. 2003. Thermal dose optimization method for ultrasound surgery. *Phys. Med. Biol.* **48** 745–762.

Marieb, E. N. 2002. *Essentials of Human Anatomy and Physiology*. Benjamin Cummings.

Mihai, A., E. Rakovitch, K. Sixel, T. Woo, M. Cardoso, C. Bell, M. Ruschin, J. P. Pignol. 2005. Inverse vs. forward breast IMRT planning. *Med. Dosim.* **30** 149–154.

Murray, J. M. 1990. Optimal control for a cancer chemotherapy problem with general growth and loss functions. *Math. Biosci.* **98** 273–287.

Pereira, F. L., C. E. Pedreira, J. B. De Sousa. 1994. A new optimization based approach to experimental combination chemotherapy. *Frontiers Med. Biol. Engrg.* **6** 257–268.

Scherrer, A., K. H. Kufer, T. Bortfeld, M. Monz, F. Alonso. 2005. IMRT planning on adaptive volume structures—A decisive reduction in computational complexity. *Phys. Med. Biol.* **7** 2033–2053.

Skomorovski, K., H. Harpak, A. Ianovski, M. Vardi, T. P. Visser, S. Hartong, H. van Vliet, G. Wagemaker, Z. Agur. 2003. New TPO treatment schedules of increased safety and efficacy: Pre-clinical validation of a thrombopoiesis simulation model. *Br. J. Haematol.* **123** 683–691.

Swan, G. W. 1987. Optimal control analysis of a cancer chemotherapy problem. *J. Math. Appl. Medicine Biol.* **4** 171–184.

Swan, G. W. 1990. Role of optimal control theory in cancer chemotherapy. *Math. Biosci.* **101** 237–284.

Swierniak, A. 1995. Cell cycle as an object of control. *J. Biol. Systems* **3** 41–54.

Swierniak, A. 1996. Optimal control problems arising in cell-cycle-specific cancer chemotherapy. *Cell Prolif.* **29** 117–139.

Ubezio, P., G. Tagliabue, B. Schechter, Z. Agur. 1994. Increasing 1-b-D-arabinofuranosylcytosine efficacy by scheduled dosing intervals based on direct measurement of bone marrow cell kinetics. *Cancer Res.* **54** 6446–6451.

Vainstein, V., Y. Ginosar, M. Shoham, D. Ranmar, A. Ianovski, Z. Agur. 2005. The complex effect of granulocyte on human granulopoiesis analyzed by a new physiologically-based mathematical model. *J. Theor. Biol.* **234** 311–327.

Webb, G. F. 1990. Resonance phenomena in cell population chemotherapy models. *Rocky Mountain J. Math.* **20** 1195–1216.

Wu, X., Y. Zhu. 2001. A global optimization method for three-dimensional conformal radiotherapy treatment planning. *Phys. Med. Biol.* **46** 107–119.