Adv. Appl. Prob. **43,** 649–665 (2011)
Printed in Northern Ireland
© *Applied Probability Trust* 2011

A FORWARDS INDUCTION APPROACH TO CANDIDATE DRUG SELECTION

S. QU [*] [**] AND
J. C. GITTINS,[*] [***] *University of Oxford*

Abstract

A forwards induction policy is a type of greedy algorithm for Markov decision processes. It is straightforward to implement and is optimal for a large class of models, especially in stochastic resource allocation. In this paper we consider a model for the optimal allocation of resources in pre-clinical pharmaceutical research. We show that although they are not always strictly optimal, forwards induction policies perform well.

Keywords: Pharmaceutical research; resource allocation; forwards induction; Markov decision process

2010 Mathematics Subject Classification: Primary 90B36
Secondary 90C40; 91B32

1. Introduction

Markov decision processes (MDPs), also known as discrete-time stochastic control processes, are central to the study of sequential optimisation problems that arise in a wide range of fields. An MDP is characterised by sets of states and actions, with associated Markovian transition probabilities, and rewards. An optimal solution is a policy for choosing actions which in some sense maximises the total reward. The calculation of optimal policies is in general a complex computational challenge. The standard approaches are iterative schemes based on dynamic programming; see, for example, Puterman (2005).

Gittins (1979) (also see Gittins *et al.* (2011a)) proposed a forwards induction (FI) approach to the development of policies for MDPs. The principle of FI is that the next decision is always that which maximises the immediate expected reward rate, with no attempt to look further ahead. It is thus a kind of greedy algorithm. FI policies are straightforward to implement, and are optimal for a large class of models, especially in stochastic resource allocation. They are discussed in detail in Glazenbrook and Gittens (1993) and Glazenbrook (1995).

In this paper we investigate the application of FI policies to resource allocation problems in pharmaceutical research, where candidate drug (CD) selection is an important subproblem. The conditions for an MDP do not hold, and FI policies are not in general strictly optimal. We shall proceed as follows. In Section 2 we discuss an allocation problem in pharmaceutical research. In Section 3 we describe a simplified CD selection problem and show the optimality of FI for this problem. The use of FI for the general CD selection problem is discussed in Section 4, and its performance is assessed by simulation tests in Section 5. It turns out to perform very well.

The authors are developing the software package OPRRA (see Gittins *et al.* (2011b)). This is an acronym for *optimising pharmaceutical research resource allocation*. OPRRA is able to

Received 17 December 2009; revision received 24 February 2011.

[*] Postal address: Department of Statistics, University of Oxford, 1 South Parks Road, Oxford OX1 3TG, UK.
[**] Email address: squ@stats.ox.ac.uk
[***] Email address: gittins@stats.ox.ac.uk

implement all the algorithms described in this paper. Its development is an important motivation for the work described here.

2. Resource allocation in pharmaceutical research

2.1. Background

Typically a commercial pharmaceutical project begins with bio-scientists developing a hypothesis for the way in which a chemical intervention in the body's processes might achieve the desired results. This is likely to mean identifying a target protein (or simply *target*) with which the proposed new drug should bind. This leads to screening tests in which many compounds are examined for relevant activity.

An important stage in pharmaceutical research is the discovery of a *lead* compound. This is a compound which has sufficiently promising characteristics to be used as the starting point of a *lead series* (LS) of compounds for use in screening tests in the search for a compound which is worthy of submission for clinical trials as a CD. Because of the uncertainties involved, it is desirable after finding a first LS to find further *backup* LS. We shall describe the search for a CD as *optimisation* of the relevant LS. Any subsequent compound selected for development after the first CD is found from any particular LS is called a *backup* compound. At most 20% of CDs emerge as marketable drugs, so typically more compounds are screened while a CD is undergoing clinical trials, so that one or more backup compounds may in turn be selected for development. These may be from more than one LS, which reduces the risk that they all fail for the same reason.

The number of LS to be optimised and the number of CDs to be selected from each of them are key factors that influence the profitability of a pharmaceutical project. Yu and Gittins (2008) considered a class of (s, n) selection policies which optimise the profitability with respect to s and n, where s is the number of LS from which we hope to find CDs and n is the maximum number of LS from which we will attempt to find CDs. Charalambous and Gittins (2008) investigated explicit optimal policies for the selection of successive CDs for two restricted versions of the model considered in Yu and Gittins (2008).

In the next subsection we introduce the notation and underlying assumptions for the CD selection problem.

2.2. Preliminaries

The process of finding a candidate drug is modelled as five successive stages.

1. Before screening.
2. Looking for an LS.
3. Looking for a backup LS.
4. Looking for a first CD in an LS, also described as *optimising* the LS.
5. Looking for a backup CD from an LS which has already provided one or more CDs.

The probability that stage i reaches a successful conclusion, given the successful completion of the relevant earlier stages, we write as p_i, with $q_i = 1 - p_i$, $i = 1, 2, 3, 4, 5$. Parallelism is assumed between stages 3 and 4, and stages 3, 4, and 5 can be repeated as often as necessary, except that if stage 3 fails, we assume that it is not repeated, for the reason that any repetition would also result in failure. Reasonably realistically, we also assume that $p_5 = 1.0$. We assume that the effort rates assigned to stages 3 and 4 when they are in parallel are adjusted to ensure that the two stages both finish at the same time.

For each stage, we have an allocation u_i, $i = 1, 2, 3, 4, 5$, where u_i is measured in terms of senior scientists. Denote by β the cost per senior scientist per year.

The time taken to complete each stage is t_i, $i = 1, 2, 3, 4, 5$, except that the time required to complete stage 4, given that a first LS has already been successfully optimised, is ρt_4 $(0 < \rho \leq 1)$. An important aspect of the overall resource allocation problem for a project is to choose suitable values for the stage allocations u_i. This issue is discussed in Yu and Gittins (2008), including the role of the effectiveness function governing the relationship between u_i and t_i. In this paper, however, our primary focus is on the selection of CDs.

The drug development process involves three successive phases of clinical trials. Each phase has a duration t_i, probability of success p_i, conditional on the success of any previous phase or phases, and associated cost c_i, $i = \mathrm{I}, \mathrm{II}, \mathrm{III}$.

A CD is *successful* if it passes through all three phases of clinical trials and becomes marketable. An LS is *good* if it can produce successful CDs. The target is *achievable* if it is possible to find a good LS. Let p_a be the probability that the target is achievable, given that we have found a CD; let p_b be the probability that an LS is good given that the target is achievable; and let p_c be the probability that a CD is successful given that it is from a good LS and the target is achievable. For achievable targets, LS are assumed to be good or bad independently, and CDs from good LS are assumed to be successful or not independently. Writing p for the probability that a CD is successful, we have $p = p_{\mathrm{I}} p_{\mathrm{II}} p_{\mathrm{III}} = p_a p_b p_c$.

Since the value of a future sum of money is lower than that of the same sum of money available immediately, an exponential discount rate γ is used to calculate the expected present value of future expenditure. For income, there is additional discounting. Sales of a drug will be lower if the launch time is delayed, due to the general tendency for better drugs to become available from competitors as time goes by. This effect is described as *obsolescence* and can be represented by an increase in the discount rate. We denote by ν the obsolescence rate, and by γ_1 the discount rate including obsolescence, so that $\gamma_1 = \gamma + \nu$. The expected value of the new drug is discounted at the rate γ_1, whereas the expected expenditure is discounted at the rate γ.

Let D be the expected value at the time when it becomes available of the first new drug from the project which completes phase III clinical trials successfully. The expected value of the first CD, ignoring the cost of clinical trials and without discounting for the time taken, is pD.

The additional expected value, at the time when it becomes available, of any second or subsequent new drug is smaller than D, because of competition from the first new drug and any others which are earlier. This is a separate effect from the obsolescence effect already described, which is caused by competition from other companies' products. We assume that the additional expected present value of the nth new drug from a project is $\lambda^{n-1} D$, $0 < \lambda < 1$.

The following proposition is proved in Charalambous and Gittins (2008).

Proposition 1. *The additional expected value of a subsequent CD conditional on the set of previously selected CDs is* $pD\,\mathrm{E}(\eta^m \mid A)$*, where* $\eta = 1 - (1 - \lambda) p_c$*, m is the number of earlier CDs which are either from the same LS or from another LS which is good, and A is the event {the target is achievable}.*

The measure which we shall use for profitability is the *profitability index* (PI), which is the expected reward divided by the expected cost. The algorithms considered in this paper seek to maximise the PI for the project. This is one of the standard criteria for the profitability of investment projects (see, for example, Brealey and Myers (2000)). A portfolio of projects with high PI values leads to a high value for the total expected reward, or net present value (NPV), when the total capital available for investment is limited. Our focus is on the efficient use of resources before clinical trials start. For that reason, we evaluate the total reward as the net reward after deduction of the cost of clinical trials, and include in the denominator only the costs incurred before clinical trials start.

The PI is a reward rate, and may be defined for a CD, or for an LS, as well as for the project as a whole. In all cases the expectations are conditional on the previous history of the project. Although our setup is not a discounted MDP, we shall proceed to use PI as the basis for defining FI policies.

2.3. FI policies

First some definitions. A *decision point* is a point at which a decision must be taken either to look for a CD to send to clinical trials, in which case we must also decide on the LS from which to look for a CD, or to stop. The *youngest* LS is the LS from which the minimum number of CDs have been selected. PI(CD) is the PI for an additional CD from the youngest already optimised LS, conditional on the set of previously selected CDs. PI(LS) is the PI for starting a new LS, conditional on the set of previously selected CDs. PI(Proj) is the PI for the project as a whole, maximised over all selection policies.

There are two versions of PI(LS), depending on whether stage 3 is to be run alongside stage 4. Choosing between them involves evaluating the PI over the CDs chosen from two successive LS. The simpler version, without a concurrent stage 3, is the PI for the LS up to the number k of CDs which gives the maximum PI; k has the property that the kth CD is the last CD from the LS for which the PI is higher than the PI for the whole LS up to that point.

In an FI policy, the sequence of CDs sent for clinical trials is determined with reference to a *reference PI*, PI(ref). The purpose of the reference PI is that it works as an approximation to the optimal PI for the project, and, therefore, screens out CDs and LS which are likely to reduce the overall PI. FI policies are more flexible than (s, n) policies, and might be expected to perform better for suitable values of PI(ref).

The FI selection algorithm is as follows.

Algorithm 1. (FI selection algorithm.)

(a) *Decide a reference PI,* PI(ref).

(b) *At each decision point, compare* PI(CD), PI(LS), *and* PI(ref).

(c) (i) *If* PI(CD) *is the biggest, take an additional CD from the youngest optimised LS and go back to step (b).*

(ii) *If* PI(LS) *is the biggest, optimise a new LS.*

- *If the attempt to find a CD in this LS is successful, take the PI-maximising number of CDs from this LS, then go back to step (b).*
- *If the attempt is not successful, go back to step (b).*

(iii) *If* PI(ref) *is the biggest, stop.*

3. A simplified CD selection problem

In this simplified problem, we allow CDs to be selected from at most one LS. We also assume in this section that whenever stage 4 is carried out stage 3 is carried out in parallel. This is for simplicity. The algorithm whose performance is described in Section 5 ensures that stage 3 is only carried out if there is a possibility that the resulting LS will be used.

The FI algorithm proceeds as before except that after the first LS, to be successfully optimised, there is no longer the option to optimise a new LS. The main conclusion of this section is that, for this modified problem, FI is optimal. We start with some further simplified problems and solution algorithms which provide useful building blocks.

3.1. Infrastructure

Problem 1. For $j = 1, 2, \dots$, let r_j be real numbers and c_j be positive real numbers, and let $q_j = r_j/c_j$, where $q_j \geq q_{j+1}$ when $j > 1$. Define $R(k) = \sum_{j=1}^{k} r_j$, $C(k) = \sum_{i=1}^{k} c_j$, $Q(k) = R(k)/C(k)$, and $Q = \sup_k Q(k)$. The problem is either to show that $Q \leq 0$ or to find k such that $Q(k) = Q$.

The following proposition on ratios of real numbers is the key to solving this and similar problems.

Proposition 2. (a) *For real numbers x_1, x_2, y_1, y_2 with $y_1 > 0$ and $y_2 > 0$, the following three statements are equivalent:*

$$\frac{x_1}{y_1} < \frac{x_2}{y_2}, \qquad \frac{x_1}{y_1} < \frac{x_1 + x_2}{y_1 + y_2}, \qquad \textit{and} \qquad \frac{x_1 + x_2}{y_1 + y_2} < \frac{x_2}{y_2}.$$

(b) *The same is true with '$<$' replaced by '$=$' throughout, or by '$\leq$'.*

Proof. (a) Assume that $x_1/y_1 < x_2/y_2$, so that $x_1 y_2 < x_2 y_1$. Adding $x_1 y_1$ to both sides of this inequality we get $x_1 y_2 + x_1 y_1 < x_2 y_1 + x_1 y_1$ and, hence, $x_1/y_1 < (x_1 + x_2)/(y_1 + y_2)$. Similarly, adding $x_2 y_2$ to both sides of the inequality instead of $x_1 y_1$ gives us $(x_1 + x_2)/(y_1 + y_2) < x_2/y_2$. Now assume that $x_1/y_1 < (x_1 + x_2)/(y_1 + y_2)$; thus, $x_1 y_1 + x_2 y_1 < x_1 y_1 + x_1 y_2$, so that $x_2 y_1 < x_1 y_2$ and, thus, $x_1/y_1 < x_2/y_2$. A similar proof shows that this also follows from the inequality $(x_1 + x_2)/(y_1 + y_2) < x_2/y_2$, and so the three original inequalities are equivalent.

(b) The proofs are almost identical.

For Problem 1, there is a straightforward solution algorithm, as follows from Lemmas 1 and 2 below.

Lemma 1. *For Problem 1, if*

$$q_{k+1} \begin{Bmatrix} > \\ = \\ < \end{Bmatrix} Q(k) \quad \textit{for some } k \textit{ then} \quad Q(k+1) \begin{Bmatrix} > \\ = \\ < \end{Bmatrix} Q(k).$$

Proof. If $q_{k+1} > Q(k)$ then $r_{k+1}/c_{k+1} > R(k)/C(k)$. Using Proposition 2(a),

$$\frac{r_{k+1} + R(k)}{c_{k+1} + C(k)} > \frac{R(k)}{C(k)};$$

hence, $Q(k+1) > Q(k)$. The other two parts of the lemma also follow, in similar fashion, from Proposition 2.

Lemma 2. *Consider Problem 1.*

(a) *There is at most one k for which $q_k \geq Q(k) > q_{k+1}$. For this k, $Q(k) = Q$.*

(b) *If $Q > 0$ and $q_j < 0$ for some j, then there is a k with the above property.*

(c) *If, for some k, $Q(j) \leq 0,\ 1 \leq j \leq k$, and $q_{k+1} \leq 0$, then $Q \leq 0$.*

Proof. If $q_{k+1} \geq Q(k)$ then $Q(k+1) \geq Q(k)$. So either (i) $q_{k+1} \geq Q(k)$, and $Q(k+1) \geq Q(k)$ for all k, or (ii) there exist m such that $q_{m+1} < Q(m)$, in which case we define k to be the smallest such m. In case (ii) it follows that $q_{n+1} \geq Q(n)$ for all $n < k$, and $q_{k+1} < Q(k)$. The first of these statements implies that $Q(n+1) \geq Q(n)$ for all $n < k$. From the second statement, it follows from Lemma 1 that $q_{k+1} < Q(k+1) < Q(k)$. This in turn implies that $q_{k+2} < Q(k+1)$, and, hence, that $q_{k+2} < Q(k+2) < Q(k+1)$, and the argument extends by induction on n to show that $q_{n+1} < Q(n+1) < Q(n)$ for all $n > k$.

It follows that in case (i) $\lim_{k\to\infty} Q(k) = Q$, and in case (ii) $Q(k) = Q$. For case (ii), it is straightforward to check that the chosen k is the unique value described in the statement of the lemma. In case (i) there is no k for which $q_k \geq Q(k) > q_{k+1}$, and part (a) of the lemma is proved. For part (b) of the lemma, note that, under the given conditions, it is impossible that $Q(k+1) \geq Q(k)$ for all k, and, therefore, case (i) does not occur. Part (c) of the lemma follows from Proposition 2 and the fact that $q_j \geq q_{j+1},\ j > 1$.

From Lemma 2, and with the additional assumption that $q_j < 0$ for some j, it follows that Problem 1 may be solved by the following algorithm.

Algorithm 2. *Compute in succession $Q(k),\ k = 1, 2, \dots,$ stopping either when $q_k \geq Q(k) > q_{k+1}$, in which case $Q(k) = Q$, or when $Q(j) \leq 0,\ 1 \leq j \leq k$, and $q_{k+1} \leq 0$, in which case $Q \leq 0$.*

The following variants of Problem 1 are also of interest.

Problem 2. For x belonging to the finite index set I, which includes 1, let r_x be a real number, c_x be a positive real number, and let $q_x = r_x/c_x$. Let S denote a subset of I, and let $R(S) = \sum_{x\in S} r_x$, $C(S) = \sum_{x\in S} c_x$, $Q(S) = R(S)/C(S)$, and $Q = \max_{S\in U} Q(S)$, where $U = \{S \subset I,\ 1 \in S\}$. The problem is to find $S \in U$ such that $Q(S) = Q$.

Problem 3. This is the same as Problem 2 except that the class of index subsets U is replaced by a subclass $W \subset U$.

Problem 4. The definition is motivated by Lemmas 3 and 4 below. In these lemmas V denotes the class of index subsets of the form $\{x : q_x > \xi\} + 1$ for some ξ.

Lemma 3. *For any $S \in U \setminus V$, there exists $S' \in U$ such that either $Q(S') > Q(S)$ or $Q(S') = Q(S)$ and $|S'| = |S| + 1$.*

Proof. Suppose that $S \in U \setminus V$. Then there exist $x \in S - 1$ and $y \in I \setminus S$ such that at least one of the following statements holds.

(i) $q_y > q_x \geq Q(S)$.

(ii) $q_y > q_x < Q(S)$.

(iii) $q_y = q_x > Q(S)$.

(iv) $q_y = q_x < Q(S)$.

(v) $q_y = q_x = Q(S)$.

In cases (i) and (iii), let $S^{'} = S + y$. In cases (ii) and (iv), let $S^{'} = S - x$. In all these cases it follows from Proposition 2 that $Q(S^{'}) > Q(S)$. In case (v), let $S^{'} = S + y$ and we have $Q(S^{'}) = Q(S)$ and $|S^{'}| = |S| + 1$.

Lemma 4. *For any $S \in U$, there exists $S' \in V$, with $Q(S') \geq Q(S)$.*

Proof. We shall suppose the converse of the lemma and show that this leads to a contradiction. Thus, suppose that $S \in U$ and that there does not exist $S' \in V$ with $Q(S') \geq Q(S)$, so that in particular $S \notin V$. From Lemma 3, it follows that there exists $S_1 \in U$ with $Q(S_1) > Q(S)$. By our assumption, $S_1 \notin V$, so again applying Lemma 3 it follows that there exists $S_2 \in U$ with $Q(S_2) > Q(S_1)$. Again, by assumption, $S_2 \notin V$, and the argument may be repeated to define an infinite sequence $S_o(= S), S_1, S_2, \dots$ such that $S_\tau \in U \setminus V$ and $Q(S_{\tau+1}) > Q(S_\tau)$ for all τ. However, this is impossible as the index set I is finite, as is therefore the class of subsets U. This completes the proof of the lemma.

It follows from Lemma 4 that if W is a class of subsets of I with $V \subset W \subset U$ then $\max_{S\in W} Q(S) = \max_{S\in V} Q(S) = Q$. Problem 4 is the same as Problem 3 with the restriction that $W \supset V$.

Problems 2 and 4 may be transformed into problems of the form of Problem 1 as follows. For each triple (r_x, c_x, q_x), change the index x to one of the first $|I|$ integers, leaving the values of the triple unchanged, with the new index values chosen so that r_1, c_1, and q_1 are unchanged and $q_i \geq q_{i+1}$, $i > 1$. Let q_i be large and negative for $i > |I|$. To complete the solution, we carry out Algorithm 2, and then reverse the index value changes.

The next step in this sequence of preliminary problems is to introduce a second subscript, which allows us to model different LS.

Problem 5. For $(i, j = 1, 2, \dots)$, let r_{ij} be real numbers, c_{ij} be positive real numbers, and $q_{ij} = r_{ij}/c_{ij}$, where $q_{i1} > q_{i+1,1}$ for $i > 1$, $q_{ij} > q_{ij+1}$ for $(i, j) \neq (1, 1)$, $q_{i1} \to q < 0$ as $i \to \infty$, and $q_{ij} \to q_i < 0$ as $j \to \infty$ for all i.

Define

$$R_i(k) = \sum_{j=1}^{k} r_{ij}, \qquad C_i(k) = \sum_{j=1}^{k} c_{ij}, \qquad R(k_1, k_2, \dots, k_n) = \sum_{i=1}^{n} R_i(k_i),$$

$$C(k_1, k_2, \dots, k_n) = \sum_{i=1}^{n} C_i(k_i), \qquad Q(k_1, k_2, \dots, k_n) = \frac{R(k_1, k_2, \dots, k_n)}{C(k_1, k_2, \dots, k_n)},$$

and $Q = \sup_A Q(k_1, k_2, \dots, k_n)$, where $A = \{n > 0; k_i > 0,\ i = 1, 2, \dots, n\}$.

The problem is either to show that $Q \leq 0$ or to find $n^*, k_1^*, k_2^*, \dots, k_{n^*}^*$ such that $Q = Q(k_1^*, k_2^*, \dots, k_{n^*}^*)$.

Lemma 5. *Problem 5 may be regarded as an example of Problem 4, with the index pair* 11 *in place of* 1.

Proof. To establish this equivalence, we first note that, except that the index set $I = \{ij : 1 \leq i, j\}$ is infinite, Problem 5 is equivalent to Problem 3, with 11 (to be read as 'one-one' rather than as 'eleven') in place of 1, and W defined by the constraints that if $S \in W$ and $ij \in S$ then $k1 \in S$ for all $k \leq i$, and $il \in S$ for all $l \leq j$. To show that Problem 5 is also an example of Problem 4 (except for the infinite index set), we need to show that, for any ξ, the index subset $S_\xi = \{ij : q_{ij} > \xi\} + 11$ belongs to the subclass W.

For any given ξ, this is trivially true if $q_{ij} \leq \xi$ for all $ij \neq 11$. Suppose that $ij\ (\neq 11) \in S_\xi$, so that $q_{ij} > \xi$. It is sufficient to show that $q_{xy} \geq q_{ij}$ for every index pair xy which belongs to S_ξ because $ij \in S_\xi$, and because of the constraints defining W. If $i = 1$, we need to show that $1k \in S_\xi$, $1 \leq k \leq j$. This is true since $11 \in S_\xi$ and $q_{12} \geq q_{13} \geq \cdots \geq q_{1j} > \xi$. If $i > 1$, we need to show that $k1 \in S_\xi$, $k \leq i$, and $il \in S_\xi$, $l \leq j$. Again, this is true since $11 \in S_\xi$ and $q_{21} \geq q_{31} \geq \cdots \geq q_{i1} \geq q_{i2} \geq \cdots \geq q_{ij} > \xi$.

To solve Problem 5, our task is either to show that $Q \leq 0$, which means to show that there is no $S \in V$ with $Q(S) > 0$, or to construct the set S of index pairs which maximises $Q(S)$ for $S \in V$. For these purposes, any index pair $ij\ (\neq 11)$ for which $q_{ij} \leq 0$ may be excluded. This is because if $Q(S) > 0$ for some index set S then $Q(S+ij) < Q(S)$. We also need to note that the restraints on W do not cause the exclusion of ij to necessitate the exclusion of any index pair xy for which $q_{xy} > 0$.

Finally, we note that it follows from the inequalities satisfied by q_{ij}, q_i, and q that the number of index pairs ij for which $q_{ij} > 0$ is finite. This means that in excluding every index pair ij $(\neq 11)$ with $q_{ij} \leq 0$ we reduce the problem to one with a finite index set, and, therefore, to an example of Problem 4, completing the proof of the lemma.

Thus, we have the following algorithm for solving Problem 5.

Algorithm 3. (a) *Reduce the index set to the finite set I consisting of index pairs ij with $q_{ij} > 0$ plus the pair* 11.

(b) *Transform the resulting Problem 4 into the form of Problem 1 by changing the index set.*

(c) *Use Algorithm 2 to solve the resulting Problem 1.*

(d) *If $Q \leq 0$ for Problem 1, the same is true for Problem 5.*

(e) *If $Q > 0$ for Problem 1, the same is true for Problem 5, and we may reverse the index set changes to obtain the maximising $n^*, k_1^*, k_2^*, \ldots, k_{n^*}^*$ for Problem 5.*

3.2. The main problem and its solution

Returning to our CD selection problem, note that by restricting the selection of CDs to at most one LS at each decision point we have ensured that there is only one way in which a project may be continued. After stages 1 and 2, this is by carrying out stage 4 if there has not yet been a successful stage 4, and by selecting a new CD from the current LS if there has been a successful stage 4. The two possible continuations are never both available.

With this restriction we can write down formulae for the expected reward (after subtracting the cost of clinical trials) and expected cost (before clinical trials) for each successive CD to be selected. With discount rate γ, the cost in scientist years of employing u scientists for t years is

$$\int_0^t u\mathrm{e}^{-\gamma s}\,\mathrm{d}s = \frac{u}{\gamma}(1 - \mathrm{e}^{-\gamma t}).$$

Define

$$K_i = \beta\frac{u_i}{\gamma}(1 - \mathrm{e}^{-\gamma t_i})$$

as the total cost for stage i, $i = 1, 2, 3, 4, 5$. Recall that we are assuming that $t_3 = t_4$. Define r_{ij}, c_{ij}, and $q_{ij} = r_{ij}/c_{ij}$ to be the expected reward, expected cost, and PI for the jth CD selected from LS i. It is a consequence of the fact that there is an unique sequence of events leading to the selection of the jth CD from LS i that the unconditional expected reward and cost are the same as the expected reward and cost conditional on previous history, apart from a common factor. We may therefore, and we shall, define r_{ij} and c_{ij} to be unconditional expectations without changing the value of q_{ij}. The reason for the overlaps of notation with Problem 5 will soon become clear.

From the description given in Section 2.2, it follows that

$$\begin{aligned} r_{ij} &= p_1 p_2 (p_3 q_4)^{i-1} p_4 \\ &\quad \times [pD \exp\{-\gamma_1(t_1 + t_2 + it_4 + (j-1)t_5 + t_{\mathrm{I}} + t_{\mathrm{II}} + t_{\mathrm{III}})\}\eta^{j-1} \\ &\qquad - \exp\{-\gamma(t_1 + t_2 + it_4 + (j-1)t_5)\} \\ &\qquad\quad \times (c_{\mathrm{I}} + p_{\mathrm{I}} \exp\{-\gamma t_I\} c_{\mathrm{II}} + p_{\mathrm{I}} p_{\mathrm{II}} \exp\{-\gamma(t_{\mathrm{I}} + t_{\mathrm{II}})\} c_{\mathrm{III}})], \qquad i, j \geq 1, \end{aligned}$$

and that

$$c_{ij} = \begin{cases} K_1 + p_1 \exp\{-\gamma t_1\} K_2 + p_1 p_2 \exp\{-\gamma(t_1 + t_2)\}(K_3 + K_4), & i = j = 1, \\ p_1 p_2 (p_3 q_4)^{i-1} \exp\{-\gamma(t_1 + t_2 + (i-1)t_4)\}(K_3 + K_4), & i \geq 2,\ j = 1, \\ p_1 p_2 (p_3 q_4)^{i-1} p_4 \exp\{-\gamma(t_1 + t_2 + it_4 + (j-2)t_5)\} K_5, & \text{for all } i, \text{ and } j > 1. \end{cases}$$

With r_{ij} and c_{ij} as above, and using the fact that $\gamma_1 > \gamma$, it is easy to check that all the assumptions of Problem 5 hold, except that it is not necessarily true that $q_{i1} > q_{i2}$, $i > 1$. We shall use all the notation for Problem 5 to also refer to our CD selection problem, which we shall call Problem 3. A deterministic CD selection problem, for example, may be expressed in the form of a sequence $A = \{n > 0;\ k_i > 0,\ i = 1, 2, \dots, n\}$.

A deterministic policy is one for which the decision whether to continue or to stop depends deterministically on the number of LS which we have so far tried to optimise, and on the number of CDs so far selected. We could also consider randomised CD selection policies, for which each continue/stop decision is determined randomly. However, it is an easy consequence of Proposition 2 that if the maximum PI for a project is attainable then it is attained by a deterministic policy.

As for Problems 1 and 5, we shall present an algorithm for Problem 3 which either shows that $Q \leq 0$, so that no CD selection policy produces a positive PI, or finds a CD selection policy A which attains the optimum PI. As for Problem 5, our task is to maximise $Q(S)$ over the allowed sets S of index pairs. Each included index pair ij now corresponds to the selection of the jth CD from LS i.

As for Problem 5, we may exclude from S any index pair ij $(j > 1)$ with $q_{ij} \leq 0$. By a similar argument we may exclude any index pair $i1$ $(i > 1)$ if $q_{i1} \leq 0$ and $q_{i2} \leq 0$. These exclusions mean that we have again reduced the index set to a finite set. However, there remains the possibility that $q_{i1} \leq q_{i2} > 0$, unlike Problem 5, so we have not yet reduced the problem to an example of Problem 4.

To proceed further, we next note that, for any LS i, the conditions for Problem 1 are satisfied with $(r_i, c_i, q_i) = (r_{ij}, c_{ij}, q_{ij})$ and $Q(k) = Q_i(k) = R_i(k)/C_i(k)$, the PI for LS i when k CDs are selected. Also, $q_{ij} < 0$ for large j. We may therefore use Algorithm 2 to either find k_i^{**} such that $Q(k_i^{**}) = \sup_k Q_i(k)$, which we denote by Q_i, or to show that $Q_i \leq 0$. Note that it follows from the definitions that $Q_i(k) > Q_{i+1}(k)$ for $i > 1$ and all k, and, hence, that $Q_i > Q_{i+1}$. Note too that since the number of ij pairs for which $q_{ij} > 0$ is finite, it follows that $Q_i > 0$ for at most a finite set of i values.

Since the class of allowed sets S is finite, if $Q > 0$, there exist $n, k_1, k_2, \dots, k_n$ (all greater than 0) such that $Q = Q(k_1, k_2, \dots, k_n)$. We define $n^*, k_1^*, k_2^*, \dots, k_{n^*}^*$ to be chosen in that order to be the largest values of $n, k_1, k_2, \dots, k_n$ for which this is true. To further simplify Problem 3, we shall use the following lemma.

Lemma 6. $k_i^* \ge k_i^{**}$, $2 \le i \le n^*$.

Proof. Note first that $Q_{n^*}(k_{n^*}^*) \ge Q$. If $Q_{n^*}(k_{n^*}^*) < Q$, it follows from the fact that $Q(k_1^*, k_2^*, \dots, k_{n^*}^*) = Q$ that $Q(k_1^*, k_2^*, \dots, k_{n-1^*}^*) > Q$, using Proposition 2(a), which contradicts the definition of Q. Thus, $Q_{n^*}(k_{n^*}^*) \ge Q$.

Now suppose that $k_i^* < k_i^{**}$ and $1 < i \le n^*$. We have

$$q_{ik_i^*+1} \ge Q_i(k_i^{**}) \ge Q_{n^*}(k_{n^*}^{**}) \ge Q_{n^*}(k_{n^*}^*) \ge Q.$$

The first inequality follows from Proposition 2 and the definition of k_i^{**}. The second inequality is equivalent to $Q_i \ge Q_{n^*}$, which is true since Q_i is a decreasing function of i for $i > 1$. The third inequality is by the definition of $k_{n^*}^{**}$. Thus, $q_{ik_i^*+1} \ge Q$, so that $Q(k_1^*, k_2^*, \dots, k_i^* + 1, \dots, k_{n^*}^*) \ge Q$. This contradicts the assumed maximality of k_i^* and completes the proof.

Now note that it follows from Proposition 2(a) and the definition of Q_i that $Q_i \ge q_{ik_i^{**}+1}$. It thus follows from Lemma 6 that Problem 3 may be transformed into an equivalent problem in the form of Problem 5 as follows: $r'_{1j} = r_{1j}$, $c'_{1j} = c_{1j}$, $q'_{1j} = q_{1j}$, for all j; $r'_{i1} = R_i(1, 2, \dots, k_i^{**})$, $c'_{i1} = C_i(1, 2, \dots, k_i^{**})$, $q'_{i1} = r'_{i1}/c'_{i1}$, and the index pair ij is replaced by $i(j - k_i^{**})$, $i > 1$ and $j > k_i^{**}$.

We can now write down an algorithm which solves Problem 3.

Algorithm 4. (a) *Use Algorithm 2 to compute k_i^{**} and Q_i, $i = 2, 3, \dots,$ stopping when $Q_i <$ 0, at which point set $n^{**} = i - 1$.*

(b) *For all index pairs ij with $i > n^{**}$, replace q_{ij} by $-M$, where M is large.*

(c) *Transform the problem into one of the form of Problem 5 as just described.*

(d) *Use Algorithm 3 to solve the resulting Problem 5.*

(e) *Express the solution in terms of the index set for Problem 3.*

We conclude Section 3 by noting that the output from Algorithm 4 defines the decision to be taken at each decision point in the procedure for selecting CDs, and that the policy defined in this way is equivalent to an FI policy with PI(ref) $= Q$, the optimal PI for the project under the restrictions of this section.

4. FI for the general CD selection problem

In this section we present some properties of FI policies for the unrestricted CD selection problem. Unlike the restricted problem considered in the previous section, it is not in general true to say that an FI policy with a reference PI equal to the optimal PI for the project is itself an optimal policy. This is shown by a counterexample. Lemma 7 below, on the other hand, provides encouragement for the view that FI policies may be close to optimal, while Lemma 8 below narrows the choices which need to be considered at each step in an FI policy.

Let r_{ij} and c_{ij} now be the expected reward and cost from the jth CD in LS i, conditional on the set of previously selected CDs. Note that this is different from Section 3, where our notation referred to unconditional expectations. Here r_{ij} and c_{ij} are random variables depending on time, and on the numbers of CDs from previous LS. Let t denote the time at which the jth CD from LS i is selected.

Counterexample 1. *If* PI(CD) $=$ PI(LS) $>$ PI(ref) $=$ PI(Proj) *at a decision point P in an FI policy, it does not in general follow that taking an additional CD and trying to optimise a new LS lead to the overall maximum PI value for the project,* PI(Proj).

Proof. Let a and b be the expected reward and cost for the additional CD, and let A and B be the expected reward and cost for the new LS. Suppose that $a/b = A/B$. Let PI(ref) $= a/b - \varepsilon$, where ε (> 0) is sufficiently small to ensure that, after any of the three possible ways in which the project might be continued from point P, PI(ref) $>$ max (PI(CD), PI(LS)), so that the project then terminates under an FI policy.

The three possible continuations are

1. an additional CD is chosen from an optimised LS,
2. we try successfully to find a CD from a new LS and then select the number of backup CDs from that LS which maximises the PI for that LS,
3. we try unsuccessfully to find a CD from a new LS.

It is easy to construct numerical cases satisfying these conditions.

Let R and C be the expected reward and expected cost under an FI policy, excluding the contributions which arise after reaching the decision point P, and let θ be the probability that point P is reached. Thus, the overall PI from an FI policy is $(R+\theta A)/(C+\theta B)$ if at P we look for a CD in a new LS, and $(R+\theta a)/(C+\theta b)$ if at P we select a CD from an already optimised LS. We have assumed that $R/C < A/B = a/b$, and it therefore follows from Proposition 2 that if $A > a$ then $(R+\theta A)/(C+\theta B) > (R+\theta a)/(C+\theta b)$, completing the proof.

Lemma 7. *In a sequence of backup CDs from the same LS the PI is decreasing.*

Proof. By definition, $q_{ij+1} = r_{ij+1}/c_{ij+1}$ is the PI for the jth backup CD in LS i. Using Proposition 1, we have $r_{ij} = \mathrm{e}^{-\gamma_1 t}\eta^j v_a - \mathrm{e}^{-\gamma t} v_b$ and $c_{ij} = \mathrm{e}^{-\gamma t} K_5$. Here

$$v_a = pD\exp\{-\gamma_1(t_5 + t_{\mathrm{I}} + t_{\mathrm{II}} + t_{\mathrm{III}})\}\,\mathrm{E}(\eta^N),$$

where N is the number of previously selected CDs from good LS other than LS i, and

$$v_b = \mathrm{e}^{-\gamma t_5}(c_{\mathrm{I}} + p_{\mathrm{I}}\mathrm{e}^{-\gamma t_{\mathrm{I}}}c_{\mathrm{II}} + p_{\mathrm{I}}p_{\mathrm{II}}\mathrm{e}^{-\gamma(t_{\mathrm{I}}+t_{\mathrm{II}})}c_{\mathrm{III}}).$$

Thus, the PI for the jth backup CD in LS i is

$$q_{ij+1} = \frac{r_{ij+1}}{c_{ij+1}} = \frac{\mathrm{e}^{-\gamma_1 t}\eta^j v_a - \mathrm{e}^{-\gamma t} v_b}{\mathrm{e}^{-\gamma t}K_5} = \frac{\mathrm{e}^{-\nu t}\eta^j v_a - v_b}{K_5},$$

which is a decreasing function in j, $j \geq 1$. This completes the proof.

Lemma 7 means that within each optimised LS there is no reason for taking account of the PIs from later CDs when deciding whether or not to select one more CD. To this extent, then, the forwards induction greedy principle is correct.

Our next lemma reduces the computational complexity of implementing an FI policy. It refers to a *youngest* LS, by which we mean an LS from which the minimum number of CDs have so far been selected.

Lemma 8. *The expected value of the next CD from an already optimised LS is maximised by choosing it from the youngest optimised LS.*

Proof. It will be sufficient to show that, for any two optimised LS, we will get a higher expected value if we choose a CD from the younger LS.

Consider selection from two LS, i and j, at time t. Assume that we have so far selected x_i CDs from LS i and x_j CDs from LS j. Let ω_{ij} denote the number of CDs which have been selected from good LS other than i and j, and let

$$m_i = \omega_{ij} + x_j\mathbb{I}(j) + x_i, \qquad m_j = \omega_{ij} + x_j + x_i\mathbb{I}(i),$$

where $\mathbb{I}(k)$ is the indicator random variable for the event $\{\text{LS } k \text{ is good}\}$, $k = i, j$. Thus, using Proposition 1, and dropping the condition $|A$ from the notation, which applies to every expectation in this proof,

$$r_{ix_i+1} = \mathrm{e}^{-\gamma_1 t}\,\mathrm{E}(\eta^{m_i})v_a - \mathrm{e}^{-\gamma t}v_b, \qquad r_{jx_j+1} = \mathrm{e}^{-\gamma_1 t}\,\mathrm{E}(\eta^{m_j})v_a - \mathrm{e}^{-\gamma t}v_b,$$

where

$$v_a = pD\exp\{-\gamma_1(t_5 + t_{\mathrm{I}} + t_{\mathrm{II}} + t_{\mathrm{III}})\},$$
$$v_b = \mathrm{e}^{-\gamma t_5}(c_{\mathrm{I}} + p_{\mathrm{I}}\mathrm{e}^{-\gamma t_{\mathrm{I}}}c_{\mathrm{II}} + p_{\mathrm{I}}p_{\mathrm{II}}\mathrm{e}^{-\gamma(t_{\mathrm{I}}+t_{\mathrm{II}})}c_{\mathrm{III}}),$$

so that

$$\begin{aligned}\frac{r_{ix_i+1} - r_{jx_j+1}}{\mathrm{e}^{-\gamma_1 t}v_a} &= \mathrm{E}(\eta^{m_i}) - \mathrm{E}(\eta^{m_j})\\ &= \mathrm{E}(\eta^{m_i} - \eta^{m_j})\\ &= \mathrm{E}(\eta^{\omega_{ij}+x_j\mathbb{I}(j)+x_i} - \eta^{\omega_{ij}+x_j+x_i\mathbb{I}(i)})\\ &= \mathrm{E}(\eta^{\omega_{ij}}(\eta^{x_j\mathbb{I}(j)+x_i} - \eta^{x_j+x_i\mathbb{I}(i)})).\end{aligned}$$

Since ω_{ij} is independent of the events $\{\text{LS } i \text{ is good}\}$ and $\{\text{LS } j \text{ is good}\}$, conditional on the event that the target is achievable, we have

$$\mathrm{E}(\eta^{\omega_{ij}}(\eta^{x_j\mathbb{I}(j)+x_i} - \eta^{x_j+x_i\mathbb{I}(i)})) = \mathrm{E}(\eta^{\omega_{ij}})\,\mathrm{E}(\eta^{x_j\mathbb{I}(j)+x_i} - \eta^{x_j+x_i\mathbb{I}(i)}),$$

so that

$$\begin{aligned}\mathrm{E}(\eta^{x_j\mathbb{I}(j)+x_i} - \eta^{x_j+x_i\mathbb{I}(i)}) &= p_b\eta^{x_i+x_j} + (1-p_b)\eta^{x_i} - p_b\eta^{x_i+x_j} - (1-p_b)\eta^{x_j}\\ &= (1-p_b)(\eta^{x_i} - \eta^{x_j}).\end{aligned}$$

This is positive if $x_i < x_j$, since $p_b < 1$ and $0 < \eta < 1$. Hence, $r_{ix_i+1} - r_{jx_j+1} > 0$ if $x_i < x_j$, as required. This completes the proof.

5. Algorithm performance

In this section the performance of different algorithms for optimising the profitability index of a project are compared. These are algorithms that are used to calculate and optimise expected rewards and costs using explicit formulae ((s, n) policies, and the restricted FI policies discussed in Section 3) and simulation algorithms (unrestricted FI policies). The advantage of (s, n) policies is that we have explicit formulae for total expected rewards and costs, and can optimise directly with respect to allocation rates (for a full account, see Charalambous (2009)); however, they have the disadvantage of an inflexible procedure for selecting CDs. First we formally define an (s, n) policy.

5.1. (s, n) policies

The parameters s and n which define an (s, n) policy have the following meanings:

- s is the number of LS from which we would like to select CDs,
- n is the maximum number of LS from which we will attempt to find CDs. Obviously, $s \leq n$.

The principles of an (s, n) policy are as follows.

1. Stage 3 is carried out in parallel with stage 4, except for the nth time that stage 4 is carried out.
2. If an LS becomes the ith successfully optimised series, $1 \leq i < s$, selecting k_i CDs from it has precedence over optimising further LS. After we have obtained k_i CDs from the ith optimised LS, we will never return to this LS for more CDs.
3. If an LS becomes the sth successfully optimised LS before all n attempts have been made, the project will be terminated after selecting k_s CDs from it.
4. The project is terminated after n optimisation attempts have been made, even if the number of optimised LS is less than s.

5.2. Simulations

A simulation of any system or process in which there are inherently random components requires a method of generating numbers that are random. Thus, a good random number generator (RNG) is essential for obtaining accurate simulation results. The RNG we use in the simulation study is a mixed generator recommended in the book by Press *et al.* (2007), with a period length of 1.8×10^{19}.

A replication is a run of a simulation model that utilises a specific stream of random numbers. Different streams of random numbers are used for different replications of simulation in our study to ensure the accuracy of the estimate. Each run of the simulation results in a reward and a cost for the project. Let X_i and Y_i be the reward and cost obtained on the ith replication for $i = 1, 2, \dots, n$. Then the X_is and Y_is are independent and identically distributed random variables. Thus, the sample means

$$\bar{X} = \frac{\sum_{i=1}^{n} X_i}{n} \quad \text{and} \quad \bar{Y} = \frac{\sum_{i=1}^{n} Y_i}{n}$$

are unbiased estimators of the population means for reward and cost, and the sample variances

$$S_X^2 = \frac{\sum_{i=1}^{n} [X_i - \bar{X}]^2}{n-1} \quad \text{and} \quad S_Y^2 = \frac{\sum_{i=1}^{n} [Y_i - \bar{Y}]^2}{n-1}$$

are unbiased estimators of the population variances for reward and cost. Denote by $\overline{\mathrm{PI}} = \bar{X}/\bar{Y}$ the average PI for the project. The corresponding estimated variance for $\overline{\mathrm{PI}}$ is derived using the delta method (see, for example, Casella and Berger (2001)), which gives

$$\widehat{\mathrm{var}}\left(\frac{\bar{X}}{\bar{Y}}\right) = \frac{1}{n}\left[\frac{\bar{X}^2}{\bar{Y}^4} S_Y^2 + \frac{S_X^2}{\bar{Y}^2} - \frac{2\bar{X}}{\bar{Y}^3}\operatorname{cov}(\bar{X}, \bar{Y})\right],$$

and, therefore, the 95% confidence interval for $\overline{\mathrm{PI}}$ is

$$\left[\frac{\bar{X}}{\bar{Y}} - 1.96\sqrt{\widehat{\mathrm{var}}\left(\frac{\bar{X}}{\bar{Y}}\right)},\ \frac{\bar{X}}{\bar{Y}} + 1.96\sqrt{\widehat{\mathrm{var}}\left(\frac{\bar{X}}{\bar{Y}}\right)}\right].$$

We set the number of runs $n = 100\,000$ in all of the simulation tests.

5.3. Examples

Six artificial projects are examined in this paper. Most of the parameters were set at realistic values. The other values were chosen so as to illustrate particular features of the solution algorithms. Stages 1 and 2 have to be completed successfully before anything else of interest can happen, and were treated as one simplified stage by setting $u_1 = t_1 = p_{22} = 0$ and $p_{21} = 1$ for all projects.

The procedures used to evaluate, and in some cases also to optimise, the PI were as follows.

1. *(s, n) expectation algorithm (SNEA).* This algorithm evaluates the PI for given allocations and values of s and n. It can also optimise the allocations for given s and n using numerical optimisation procedures. The values s_{opt} and n_{opt} of s and n which lead to the overall maximum value of the PI may be determined by repeatedly running the algorithm with different values of s and n.

2. *FI simulation.* This algorithm provides a distribution of outcomes for given allocations, using FI to select successive CDs. It was used to provide estimates for PI for the two cases

   - $(s, n) = (s_{\mathrm{opt}}, n_{\mathrm{opt}})$, or FIOPTSIM,
   - $(s, n) = (1, 1)$, or FI11SIM,

   with allocations optimised using SNEA, and PI(ref) equal to the optimal PI calculated with SNEA for the corresponding s and n.

3. *Restricted FI expectation algorithm, or RFIEA.* This is the algorithm described in Section 3 for sequential CD selection. It was implemented with allocations optimised using SNEA with $(s, n) = (1, 1)$.

For many realistic projects, $(s_{\mathrm{opt}}, n_{\mathrm{opt}}) = (1, 1)$, and the optimal solutions from all solution algorithms take simple forms. To avoid these simple cases, all projects (see Table 1) other than project 1 have been defined with two distinct features. Firstly, they have been loaded with high initial costs by giving high values to u_2 and t_2. This means that more than one attempt to optimise an LS is desirable to achieve a high PI. Secondly, to ensure that the effect of later CDs on the PI is not of negligible importance compared with those discovered earlier, the discount parameter γ_1 is set to be small.

The desirability of selecting CDs from more than one LS, and, hence, the likelihood that $s_{\mathrm{opt}} > 1$ is reinforced if backup CDs are loaded with high costs (projects 3, 4, and 6), and also if λ is low (projects 3 and 6) and p_b is low (projects 3, 5, and 6).

All six projects have

$$\begin{aligned}&(p_1, p_{21}, p_{22}, u_1, t_1, c_{\mathrm{I}}, c_{\mathrm{II}}, c_{\mathrm{III}}, \rho, \beta, t_{\mathrm{I}}, t_{\mathrm{II}}, t_{\mathrm{III}})\\ &\qquad = (1, 1, 0, 0, 0, 0.4m, 1.2m, 13m, 0.8, 300\,000, 1, 2, 3).\end{aligned}$$

For projects 1 to 5, $D = 300$ m, and for project 6, $D = 600$ m. The remaining project

TABLE 1: Parameters and SNEA results for projects 1 to 6.

Project 1	
Optimal solution	$(s, n) = (1, 1)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_4) = (1, 1, 1, 1, 1, 0.5)$, $(\gamma, \gamma_1, \lambda) = (0.2, 0.3, 0.8)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (10, 10, 15, 9, 1.0, 0.9, 2.5, 1.5)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (33, 10, 40, 40, 0.33, 1.09, 1.09, 0.39)$,
	$k_1 = 4,\ \mathrm{PI} = 1.606$
Project 2	
Optimal solution	$(s, n) = (1, 4)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_3, p_4) = (0.7, 0.5, 0.7, 0.9, 0.7, 1, 0.5)$,
	$(\gamma, \gamma_1, \lambda) = (0.09, 0.14, 0.8)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (40, 10, 15, 9, 2.0, 1.0, 2.5, 1.5)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (43, 11, 52, 40, 1.9, 0.92, 0.92, 0.39)$,
	$k_1 = 10,\ \mathrm{PI} = 1.113$
Project 3	
Optimal solution	$(s, n) = (3, 12)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_3, p_4) = (0.5, 1, 1, 1, 0.5, 0.8, 0.5)$,
	$(\gamma, \gamma_1, \lambda) = (0.09, 0.14, 0.5)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (30, 5, 5, 9, 4, 0.5, 0.5, 1.5)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (39, 10, 10, 35, 3.28, 0.25, 0.25, 0.43)$,
	$(k_1, k_2, k_3) = (2, 2, 1),\ \mathrm{PI} = 1.375$
Project 4	
Optimal solution	$(s, n) = (2, 5)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_3, p_4) = (0.7, 0.5, 0.7, 0.9, 0.7, 0.6, 0.6)$,
	$(\gamma, \gamma_1, \lambda) = (0.09, 0.14, 0.8)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (40, 10, 10, 9, 3, 1, 1, 1.5)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (43, 51, 50, 41, 2.86, 0.25, 0.25, 0.38)$,
	$(k_1, k_2) = (9, 4),\ \mathrm{PI} = 1.158$
Project 5	
Optimal solution	$(s, n) = (3, 3)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_3, p_4) = (0.7, 0.5, 0.6, 0.9, 0.3, 0.8, 0.6)$,
	$(\gamma, \gamma_1, \lambda) = (0.09, 0.14, 0.8)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (20, 10, 15, 9, 2, 0.9, 1, 1.5)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (34, 24, 46, 32, 1.28, 0.4, 0.4, 0.46)$,
	$(k_1, k_2, k_3) = (4, 3, 3),\ \mathrm{PI} = 1.357$
Project 6	
Optimal solution	$(s, n) = (3, 22)$
Parameters	$(p_{\mathrm{I}}, p_{\mathrm{II}}, p_{\mathrm{III}}, p_a, p_b, p_3, p_4) = (0.5, 1, 1, 1, 0.5, 1, 0.4)$,
	$(\gamma, \gamma_1, \lambda) = (0.09, 0.14, 0.5)$
Initial inputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (40, 2, 5, 9, 4, 0.5, 0.5, 1)$
Optimal outputs	$(u_2, u_3, u_4, u_5, t_2, t_3, t_4, t_5) = (41, 4, 10, 33, 3.93, 0.25, 0.25, 0.3)$,
	$(k_1, k_2, k_3) = (2, 2, 2),\ \mathrm{PI} = 3.322$

TABLE 2: Profitability index values for projects 1 to 6.

Project	s_{opt}	n_{opt}	SNEA		FISIM		RFIEA
			OPT	(1, 1)	OPT	(1, 1)	(1, 1)
1	1	1	1.606	—	1.607	—	1.606
2	1	4	1.113	0.987	1.116	1.111	1.219
3	3	12	1.375	0.736	1.401	1.400	1.101
4	2	5	1.158	1.046	1.148	1.145	1.147
5	3	3	1.357	1.248	1.383	1.381	1.303
6	3	22	3.322	1.061	3.736	3.688	2.272

parameters, together with both the initial and SNEA optimised stage resource allocations and stage durations, are detailed in Table 1.

5.4. Results

The PI values and estimates which result from applying the different optimisation and simulation algorithms are shown in Table 2. They show the following features.

- For project 1, $(s_{\text{opt}}, n_{\text{opt}}) = (1, 1)$, so there are just three distinct procedures. They all give the same value for the PI.
- The values of $(s_{\text{opt}}, n_{\text{opt}})$ and of $\text{PI}(s_{\text{opt}}, n_{\text{opt}})/\text{PI}(1, 1)$ for the SNEA are in line with the expectations set out above.
- FIOPTSIM is always at least as good as SNEA.
- RFIEA is best when either $s_{\text{opt}} = 1$ or $\text{PI}(s_{\text{opt}}, n_{\text{opt}})/\text{PI}(1, 1)$ for the SNEA is close to 1.
- FI11SIM is almost as good as FIOPTSIM, and better than RFIEA except when either $s_{\text{opt}} = 1$ or $\text{PI}(s_{\text{opt}}, n_{\text{opt}})/\text{PI}(1, 1)$ for the SNEA is close to 1.

The 95% confidence intervals for each of the 22 PI values which were estimated by simulation are of the form *estimated PI value* $\pm x$, where $0.007 \leq x \leq 0.020$.

6. Conclusion

The most striking aspect of these results is the strong performance of FI11SIM. This means that the performance of the FI algorithm does not depend strongly on the choice of PI(ref). It also means that the PI resulting from allocations given by the SNEA with $(s, n) = (1, 1)$, followed by selection of CDs using FI11SIM, is close to the highest attainable PI. This in turn means that there is no need to determine s_{opt} and n_{opt}, which requires repeated application of the SNEA.

We plan to carry out further tests on the performance of the various algorithms and to use the results as the basis for recommendations on the use of the OPRRA software.

References

BREALEY, R. AND MYERS, S. (2000). *Principles of Corporate Finance*, 6th edn. McGraw Hill, New York.
CASELLA, G. AND BERGER, R. L. (2001). *Statistical Inference*, 2nd edition. Duxbury Press.
CHARALAMBOUS, C. (2009). Models and software for improving the profitability of pharmaceutical research. Doctoral Thesis, University of Oxford.

Charalambous, C. and Gittins, J. C. (2008). Optimal selection policies for a sequence of candidate drugs. *Adv. Appl. Prob.* **40,** 359–376.

Gittins, J. C. (1979). Bandit processes and dynamic allocation indices. *J. R. Statist. Soc. B* **41,** 148–177.

Gittins, J., Glazebrook, K. and Weber, R. (2011a). *Multi-Armed Bandit Allocation Indices*, 2nd edn. John Wiley, Chichester.

Gittins, J. C. *et al.* (2011b). *OPRRA User Guide*. Available at www.stats.ox.ac.uk/people/academic_staff/john_gittins.

Glazebrook, K. D. (1995). Stochastic scheduling and forwards induction. *Discrete Appl. Math.* **57,** 145–165.

Glazebrook, K. D. and Gittins, J. C. (1993). The performance of forwards induction policies. *Stoch. Process. Appl.* **46,** 301–326.

Press, W. H., Teukolsky, S. A., Vetterling, W. T. and Flannery, B. P. (2007). *Numerical Recipes*, 3rd edn. Cambridge University Press.

Puterman, M. L. (2005). *Markov Decision Processes*, 2nd edn. John Wiley, New York.

Yu, J. Y. and Gittins, J. C. (2008). Models and software for improving the profitability of pharmaceutical research. *Europ. J. Operat. Res.* **189,** 459–475.