A Simulation-Optimization Framework for Research and Development Pipeline Management

Dharmashankar Subramanian, Joseph F. Pekny, and Gintaras V. Reklaitis
School of Chemical Engineering, Purdue University, West Lafayette, IN 47907

The Research and Development Pipeline management problem has far-reaching economic implications for new-product-development-driven industries, such as pharmaceutical, biotechnology and agrochemical industries. Effective decision-making is required with respect to portfolio selection and project task scheduling in the face of significant uncertainty and an ever-constrained resource pool. The here-and-now stochastic optimization problem inherent to the management of an R&D Pipeline is described in its most general form, as well as a computing architecture, Sim-Opt, that combines mathematical programming and discrete event system simulation to assess the uncertainty and control the risk present in the pipeline. The R&D Pipeline management problem is viewed in Sim-Opt as the control problem of a performance-oriented, resource-constrained, stochastic, discrete-event, dynamic system. The concept of time lines is used to study multiple unique realizations of the controlled evolution of the discrete-event pipeline system. Four approaches using various degrees of rigor were investigated for the optimization module in Sim-Opt, and their relative performance is explored through an industrially motivated case study. Methods are presented to efficiently integrate information across the time lines from this framework. This integration of information demonstrated in a case study was used to infer a creative operational policy for the corresponding here-and-now stochastic optimization problem.

Introduction

Planning and scheduling are common to many different engineering domains. In general, the more constrained the resources, the more tightly coupled planning and scheduling will be. The Research and Development (R&D) Pipeline management problem addresses the issues of a new-product-development pipeline, where several new-product-development projects compete for a limited pool of various resource types. Each project (product) usually involves a series of testing tasks prior to product commercialization. If the project fails any of these tasks, then all the remaining work on that product is halted and the investment in the previous testing tasks is wasted. In the context of an R&D Pipeline management, planning refers to strategic planning and scheduling refers to tactical planning. Strategic planning refers to the construction of an attractive portfolio of research projects, and tactical planning refers to the temporal assignment of limited resources to tasks that are required for the actual execution of a portfolio. In its most general form, the deterministic R&D Pipeline management problem asks the following question:

Given a set of research projects, each project containing a set of activities related by generalized precedence constraints (Start-to-Start, Start-to-Finish, Finish-to-Finish, Finish-to-Start), a common pool of limited resources of various (finite) kinds and a measurement of performance, what is the best set of projects to pursue, and further, what is the best way to assign resources to activities in the chosen projects, such that the chosen measure of performance is maximized?

Clearly, planning and scheduling are inseparably connected in this context. A more realistic, and practically moti-

Correspondence concerning this article should be addressed to J. F. Pekny.

vated, problem is the preceding question in a stochastic context, that is, with uncertainty added in terms of task durations, task resource requirements, and task successes, task costs and rewards. Thus, a realistic R&D Pipeline management problem is a stochastic optimization problem combining the features of a project selection problem and generalized resource-constrained project-scheduling problem (RCPSP). The additional complexities are the task success uncertainty, duration uncertainty, and resource requirement uncertainty. An extensive review of scheduling as it applies to chemical engineering is presented by Reklaitis (1992). There also exists a large body of literature on the RCPSP, as noted in a recent review article by Brucker et al. (1999). The existing literature on the RCPSP addresses both deterministic and stochastic versions to minimize project completion time, but the stochastic version is limited to task-duration uncertainty. Task-success uncertainty, a predominant source of uncertainty in the R&D context, has not been adequately addressed in the literature, with the noted exception of Schmidt and Grossmann (1996), Honkomp (1998), Jain and Grossmann (1999), and Blau et al. (2000). Schmidt and Grossmann (1996) addressed the problem of a single project with unlimited resources while considering task-success uncertainty, duration uncertainty, and cost uncertainty with discrete stochastic scenarios. Jain and Grossmann (1999) addressed the problem of a fixed, predetermined, portfolio of projects with resource constraints and task-success uncertainty. Honkomp (1998) addressed the problem of both project selection and project scheduling with task-success uncertainty to estimate a throughput of the R&D pipeline capacity. The last two references follow deterministic mathematical programming approaches, and assume absolutely certain knowledge of the problem parameters. The failure of a task enters the preceding mathematical programming models in a fractional form. It enters the objective function, which is expressed as the expected net present value. Failure of a task also enters in a fractional form into the resource constraints, as in resource overbooking used by Honkomp (1998). While such fractional representations of failure yield plans and schedules that acknowledge failure in a broad decision-making sense, it should be noted that failure occurs in reality in a binary (whole, Fails/Succeeds) fashion, as opposed to something fractional. Such a realistic, binary representation of failure cannot be given in any *single* instance of a mathematical program, because of the chance-dependent nature of binary failure. This binary aspect of failure is important for capturing realistic operational and design possibilities in the pipeline system, since the failure of a project completely releases resources that are otherwise committed to the project. Furthermore, all of the preceding mathematical programming approaches ignore uncertainties in resource requirements and resource availability, as well as in project rewards, all of which are subject to variability in practice. Blau (1997) showed how a systems engineering approach can be used to significantly reduce the time and cost associated with the development process of new agrochemicals. More recently, Blau et al. (2000) presented a probabilistic network simulation model for the new-product development problem. They propagated uncertainties through the model in a Monte Carlo simulation sense, while assuming unlimited resources, to track the distributions of rewards and resource violations. Task scheduling was not considered within or across projects, since resource constraints were not an issue. In their approach, once a project starts, all of its tasks get executed back-to-back, that is, as soon as feasible. They applied metrics such as reward-to-risk ratio to rank-order projects and suggested rank-based staggering of project start times, along with task failure serving as a trigger event to start projects of lower priority that have not yet started.

Simulation allows for a realistic representation of failure and a detailed representation of uncertainty with relative ease. This research work integrates the merits of both optimization and simulation in a computational architecture called Sim-Opt. This article is structured as follows: The here-and-now stochastic optimization problem present in the management of an R&D Pipeline is described in its general form. Next, Sim-Opt, a computing architecture that combines mathematical programming and discrete event system simulation to assess the uncertainty present in the pipeline and to help decision making is presented. An industrially motivated case study is used to illustrate the application of Sim-Opt. Finally, methods are presented to efficiently accumulate information from this framework, and incorporate this information for more effective decision making. The literature that is related in spirit, in terms of using simulation in the context of stochastic optimization, includes Kalagnanam and Diwekar (1997), Gonzalez and Realff (1998), and Honkomp et al. (1999). Kalagnanam and Diwekar (1997) addressed the parameter-design problem as a stochastic optimization problem. It involved an optimizer in an outer loop that invoked a simulation model at each iteration to numerically estimate the multivariate probability distributions of corresponding performance statistics. Gonzalez and Realff (1998) coupled a discrete event simulator in an open loop to the output of a mixed-integer linear-programming (MILP) scheduling model for pipeless batch plants to study the sensitivity of the schedule to processing and travel time perturbations. Honkomp et al. (1999) developed a framework for directly incorporating schedules into a batch process simulator for the purposes of schedule validation and testing of rescheduling methodologies when stochastic events occur. The current work differs from these references in that it addresses a product-development, planning problem as opposed to a parameter-design problem or a batch chemical manufacturing problem. Further, it develops a framework guided by the discrete-event dynamic system perspective to introduce the concept of time lines and the integration of information across these time lines to gain design and operational insights about the system.

R&D Pipeline System as a Discrete-Event Dynamic System and the Underlying Stochastic Optimization Problem

In this section, a basic deterministic optimization formulation for the R&D pipeline problem is presented first to motivate the subsequent discussion on the underlying stochastic optimization problem and the consideration of the R&D pipeline as a discrete-event dynamic system. The deterministic optimization formulation is based on the development in Honkomp (1998). A set of research projects is given, each project consisting of a set of precedence-constrained activities (also referred to as tasks). There is a finite set of re-

sources, which are required in a certain combination for any task to be feasibly scheduled. Each task has an associated duration, cost, reward, and probability of success. The aim is to select a portfolio of projects and determine a schedule of tasks that maximizes the expected net present value (ENPV) and meets resource constraints. Both discrete and continuous variables are necessary to model the combinatorial decisions involved in the preceding project selection and scheduling problem. The variables in the basic uniform time discretization method (UDM) formulation are:

$$X_{it} \in \{0,1\} \qquad \forall i, \quad 1 \le t \le H - p_{it} \tag{1}$$

$$0 \le H_{i,i',t} \le \mathbf{1} \qquad \forall i, \quad i' \in T^I_{bi}, \qquad 1 \le t \le H - p_{it} \tag{2}$$

$$0 \le S_{rt} \le \boldsymbol{U_{rt}} \qquad \forall r, \qquad 1 \le t \le H - 1, \tag{3}$$

where

X_{it} = binary decision variable, which is 1 if task i is started at time t, and 0 otherwise
$H - 1$ = number of time periods in the planning horizon
p_{it} = time-dependent processing time of task i if it begins at time t
$H_{i,i',t}$ = variable to allow task i, which follows task i' at time t, to wait before starting
T^I_{bi} = set of tasks that immediately precede task i
S_{rt} = amount of available resource r left unused at time t
U_{rt} = maximum amount of resource r available at time t.

The allocation and precedence constraints in the formulation are Eqs. 4 and 5, respectively:

$$\sum_{t=1}^{H - p_{it}} X_{it} \le 1 \qquad \forall i \tag{4}$$

$$H_{i',i,t} = H_{i',i,t-1} + \sum_{t' | t' + p_{i't'} = t} X_{i',t'} - X_{it} \qquad \forall i \in AND_{in}, \quad i' \in T^I_{bi}, t, \tag{5}$$

where AND_{in} is a set of tasks that have an in-tree structure with *AND* logical connectivity, that is, tasks with predecessors, all of which need to be completed for feasibility. The constraints that model the demands on the end of the various projects are given below. The demands are modeled as *hard*, *soft forced*, and *soft*. Hard demands require that the task be completed by the time of the demand for the solution to be feasible. Soft-forced demands are similar to hard demands in as much as the task must be completed for the model to be feasible. However, completion later than the demand date is permitted with a penalty. The third type of demands, soft demands, are the weakest set in as much as a task with this type of demand is not required to be completed for the MILP to be feasible.

$$\sum_{t=1}^{d_i - p_{it}} X_{it} = 1 \qquad \forall i \in T_h \tag{6}$$

$$\sum_{t=1}^{H - p_{it}} X_{it} = 1 \qquad \forall i \in T_f \tag{7}$$

$$\sum_{t=1}^{d^E_i - p_{it}} X_{it} = 1 \qquad \forall i \in T_f \tag{8}$$

$$\sum_{t=1}^{d^E_i - p_{it}} X_{it} \le 1 \qquad \forall i \in T_s, \tag{9}$$

where

d_i = time of hard demand for task i
T_h = set of tasks with a hard demand
T_f = set of tasks with a soft-forced demand
T_s = set of tasks with a soft demand
d^E_i = time when the demand for task i expires

Resource utilization and capacity is modeled by discounting the requirements of every task by the compounded probability of successful finish of all corresponding predecessor tasks. This approach results in resources being scheduled at their expected usage levels.

$$\sum_{i=1}^{N\text{tasks}} \sum_{t'=0}^{p_{it}-1} P^S_i r_{i,t-t',t',r} X_{it-t'} \le U_{rt} \qquad \forall r,t, \tag{10}$$

or equivalently,

$$\sum_{i=1}^{N\text{tasks}} \sum_{t'=0}^{p_{it}-1} P^S_i r_{i,t-t',t',r} X_{it-t'} + S_{rt} = U_{rt} \qquad \forall r,t, \tag{11}$$

where

$$P^S_i = \prod_{i' \in T_{bi}} \pi_{i'} \qquad \forall i \tag{12}$$

$$P^F_i = \sum_{i' \in T_{bi} \cup \{i\}} \pi_{i'}, \qquad \forall i \tag{13}$$

and

P^S_i = parameter for the cumulative probability of task i starting
$\pi_{i'}$ = parameter for the probability of success of task i
T_{bi} = set of *all* tasks prior to task i, with a *directed path* to task i in the Activity-on-Node (AoN) directed graph (through linear or nonlinear precedence chains); it excludes tasks in the same project without a directed path to task i, in the Activity-on-Node directed graph
$N\text{tasks}$ = set of all tasks
$r_{i,t,c,r}$ = requirement for resource r by task i, which starts at time t and has completed c units of its duration

The objective function is the expected net present value of the system, and is modeled as the sum of the following terms:

$$+ \sum_{t=1}^{d^E_i - p_{it}} P^F_i C^D_{it} X_{it} \qquad \forall i \in \{T_s, T_f\} \tag{14}$$

$$+ \sum_{t=1}^{d_i - p_{it}} P^F_i C^D_{it} X_{it} \qquad \forall i \in T_h \tag{15}$$

$$+ \sum_{t=1}^{H - p_{it}} P^F_i C^D_{it} X_{it} \qquad \forall i \notin \{T_s, T_f, T_h\} \tag{16}$$

$$- \sum_{t=1}^{d^E_i - p_{it}} P^S_i C^E_{it} X_{it} \qquad \forall i \in \{T_s, T_f\} \tag{17}$$

$$- \sum_{t=1}^{d_i - p_{it}} P^S_i C^E_{it} X_{it} \qquad \forall i \in T_h \tag{18}$$

$$- \sum_{t=1}^{H-p_{it}} P_i^S C_{it}^E X_{it} \qquad \forall i \notin \{T_s, T_f, T_h\} \tag{19}$$

$$- \sum_{t=d_i}^{d_i^E - p_{it}} P_i^F \left(C_i^F + (t-d_i) C_i^V\right) X_{it} \qquad \forall i \in \{T_s, T_f\}, \tag{20}$$

where
P_i^F = parameter for the cumulative probability of task i finishing
C_{it}^D = parameter for the value of delivering task i at time t
C_{it}^E = parameter for the cost of executing task i at time t
C_i^F = parameter for fixed penalty due to missing demand for task i
C_i^V = parameter for variable penalty due to missing demand for task i.

The solution obtained with the preceding formulation is not necessarily a nondelay schedule. It may contain a delay between any two activities belonging to the same project, if, say, the objective function dictates that a delay be introduced to let a higher priority project get its resources. The notable feature in this deterministic formulation is that it accounts for the possibility of project attrition due to task failure in the pipeline, by effectively overbooking the resources as shown in Eqs. 10 and 11. This is different from the formulation of Jain and Grossmann (1999), where resource constraints are imposed in a conservative fashion without accounting for the possibility of pipeline attrition. It should be noted that the probabilistic discounting for the resource requirement of any task is done only with respect to the set, T_{bi}, which is the set of all tasks that have a *directed path* to task i, in the AoN directed graph (through linear or nonlinear precedence chains). It excludes parallel tasks in the same project that do not have a directed path to task i, in the AoN directed graph. The same argument applies for the probabilistic discounting in the expected costs and expected rewards in the objective function terms contained in Eqs. 14 to 20. This is not a serious limitation, as it can be overcome by expanding the set, T_{bi}, at every UDM time period, t, to include tasks (in the same project) that do not have a directed path to task i, but have finished before time period t. This is accomplishable at a cost in terms of increasing the number of binary variables to determine how to augment the set, T_{bi}, at every UDM time period, t. The formulation of Jain and Grossmann (1999) does not suffer from this limitation. It accounts for the probability of starting any task by considering all tasks in the same project, whether or not there exist directed paths to the task under question in the AoN graph. It does so by adding suitable binary variables. But it should be noted that their formulation considers the probabilistic implication of any task sequence only in the objective function in terms of expected costs and expected rewards, and not in the constraints. The preceding deterministic formulation as well as that of Jain and Grossmann (1999) assumes fully perfect knowledge of all problem parameters.

The practical R&D pipeline problem is complicated by the presence of significant uncertainties in the estimates of task processing times, estimates of task resource requirements, estimates of the probabilities of success/failure, and the forecasts of market returns and costs of tasks. These sources of variability, characterized as probability distributions, make the underlying problem a stochastic optimization problem. The mathematical programming formulation of this stochastic optimization problem consists of objective function and constraints that are probabilistic. A typical stochastic programming statement of the practical problem, as a chance-constrained formulation (Kall and Wallace, 1994), can be summarized as below.

Obtain a portfolio and a task schedule over the planning horizon that: maximizes the *expectation* of the net present value (NPV) of the pipeline system, subject to,

- Allocation Constraints
- $$\textit{Probability} \left\{ \begin{matrix} \textit{Satisfying Pr ecedence Constra int } s \\ \textit{Satisfying Re sourceConstra int } s \\ \textit{Satisfying DemandConstra int } s \end{matrix} \right\} \geq \alpha$$
- $\textit{Probability}\{NPV \geq \mathrm{M}\} \geq \beta$, for suitably chosen α, β, and M.

It should be noted that every constraint in the deterministic formulation that depends on random parameters is now stipulated to hold jointly only in a probabilistic sense, that is, with a joint probability of at least α. Precedence constraints and demand constraints involve random processing times, while resource constraints involve random resource requirements, random processing times, and estimates for the probabilities of success that are themselves random. An additional constraint that enters the stochastic optimization problem is the risk constraint. A practical form of the risk constraint is obtained by insisting that the NPV exceed a value M with a probability at least β. This constraint enters the problem due to variability in the estimates of market returns and costs of activities. While this article does not attempt a mathematical formulation of the preceding stochastic optimization statement, it helps to note the following about any such stochastic programming formulation. Capturing all the essentials of the preceding stochastic optimization problem in a single mathematical program will be in the form of a stochastic integer program. It will involve constructing mathematical expectations and higher moments of the net present-value distribution (objective function) and those of the probabilistic risk constraints and physical constraints. This is an arduous task, requiring multivariate numerical integration (Kall and Wallace, 1994). Given the NP-hardness of integer programming (Garey and Johnson, 1979) and the numerical complexity of stochastic programming, one monolithic stochastic integer program for the R&D pipeline problem is arguably outside the set of effectively solvable problems. Further, such a stochastic program can only have a fractional (probabilistic) representation of task success/failure. This is meaningless in the context of actual implementation, because success/failure is realized in a binary (whole) sense in the actual implementation. The output of such a stochastic optimization formulation that satisfies constraints in a probabilistic sense is somewhat disconnected from reality, in the sense that it does not say anything about any one *future* that the system could evolve along. In reality, all constraints would need to be satisfied exactly in the one and only unknown *future* that the system will evolve along. This disparity between a stochastic optimization solution and its realistic implementation needs to be addressed in a form that is practically useful. This is what is addressed next with a discrete event dynamic system view of the R&D pipeline.

Discrete-event dynamic-system view

The stochastic optimization problem in the context of R&D Pipeline management is an optimal resource-constrained project selection and task-scheduling problem in the face of significant uncertainty. The R&D Pipeline management problem can be viewed as the control problem of a performance-oriented, resource-constrained, stochastic, discrete-event, dynamic system (DEDS). For details about stochastic DEDS, refer to Cassandras (1991). The *state space* is the set of possible states of the pipeline, in terms of:

- The set, A, of active tasks, the levels of progress of these active tasks, the amounts of the various resource types that these active tasks are engaging, along with
- The set, F, of finished tasks, the success status of these finished tasks (successful finish or unsuccessful finish), the amounts of the various resource types that these finished tasks engaged, their processing times
- The feasible set, U, of tasks that are not active and have not been started. This feasible set contains tasks that are technologically feasible, but cannot be started due to resource constraints. It contains all such project tasks in the pipeline (whether in the currently chosen portfolio, or not).

It should be noted that in reality, the set, A, of active tasks always contains tasks that are simultaneously feasible with respect to resource constraints. The *event space*, corresponding to any state, is the set of events representing the finish of tasks that are active in the system at that state. The occurrence of such events is governed by the probability distribution of the task processing times. The state of the system changes at discrete points in time on the occurrence of such events. The changes are governed by the transition probabilities of tasks going from an active status to successful finish or unsuccessful finish. Such transition probabilities are themselves driven by their respective probability distributions. The *action space*, corresponding to any state, is the set of decisions corresponding to which feasible task(s) to start (which task(s) to move from set U to set A), given estimates of the task(s)'s requirements of various resource types, and the resource levels that are available in the system at that state. An action is taken at the very start of the system's evolution to establish an initial state of the system. This action establishes the portfolio with which the pipeline begins to evolve. Actions are also taken upon the occurrence of events in the system, since the finish of an active task implies disengagement of various resource types, and hence implies the capability to start new feasible tasks. These actions correspond to either the scheduling of tasks that belong to the current portfolio, or adding new projects into the portfolio. Further, actions are influenced by the probability distributions driving the resource requirements, the processing durations, and failure probabilities corresponding to the various tasks. Actions typically change the state of the system, since they change the set, A, of the active tasks. The goal of the preceding problem is to establish an initial state of the pipeline, and establish a policy for taking *actions* as the pipeline evolves through its *states*, as it traverses the stochastic state space subject to the resource constraints present in the system. Thus, the pipeline system needs dynamically changing sets of decisions due to dynamically evolving states. In higher-level terms, this translates to choosing a portfolio of projects, performing the actual task scheduling among various projects, and dynamically revising the portfolio and the schedule upon such a need.

The net present value that results from a given set of dynamic decisions, made at the here-and-now with respect to portfolio selection and task scheduling, is an indicator of the quality of decision making in the system. Uncertainties in the task processing times, task survival probabilities, task resource requirements and task rewards lead to a probabilistic distribution of the net present value of the pipeline system, for any given set of planning and scheduling decisions formed at the here-and-now. The underlying stochastic optimization problem, in the discrete event dynamic system, seeks sets of decisions at the here-and-now that maximize the expected net present value of the system, that is, maximize the mean of the distribution just described. This objective function represents the "reward" obtainable in the system. In addition, it is also desired to exercise control over the spread of the distribution, that is, control the cumulative probability of realizing a positive net present value. Clearly, the extent to which the spread of the distribution can be controlled is defined by the uncertainties inherent to the pipeline.

Sim-Opt: Computing Architecture

The discrete-event dynamic-system view of the R&D pipeline was discussed in the previous section as a practically useful way to address the differences between stochastic programming formulations of the R&D Pipeline problem and the implementation of the system in reality. In this section, a computing architecture, Sim-Opt, is presented that captures the discrete-event dynamic-system view by integrating combinatorial optimization and discrete-event system simulation. The different elements that combine and interact to create this architecture are described next, first as a general concept, and then followed by the particulars of our implementation. The optimizer, or the decision-making module, is the first element. To begin with, it establishes an initial state of the pipeline system by solving a suitably simplified deterministic version of the here-and-now stochastic optimization problem. This could be done with the use of mathematical programming or with the use of heuristics. The evolution of this initial state of the pipeline is tracked in a discrete-event system simulation module, which forms the second element. The uncertainties associated with the task processing times, task resource requirements, and task successes are modeled with appropriate probability distributions. Resource constraints are enforced in the system evolution at every point, as are the physics of the pipeline portfolio in terms of precedence relationships. As the system evolves in the constrained simulation mode to generate an artificial history, two kinds of possibilities are encountered. The first is that of a resource conflict among tasks that are simultaneously feasible within and across projects, and the second is that of a task failure that leads to attrition of projects in the pipeline. The simulation module is not equipped for resolving these possibilities. In terms of a decision-making response, it helps to think of the former as *regulatory control* action and the latter as *supervisory control* action. Both regulatory and supervisory control actions can be determined with the use of mathematical programming or with the use of heuristics. When the simulation

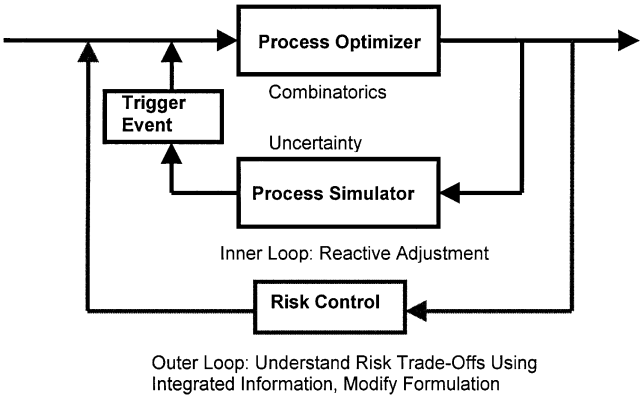

Figure 1. Sim-Opt.

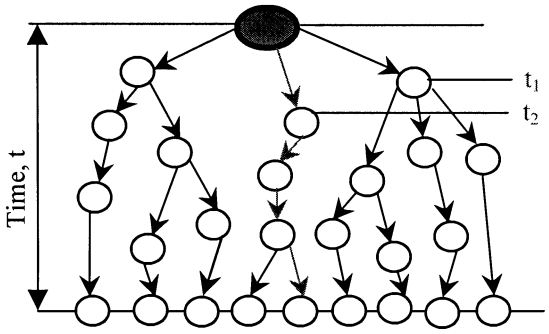

Figure 2. Multiple Monte Carlo time lines.

module encounters a need for either of the two types of control actions, it momentarily suspends itself and communicates the state of the system to the decision-making module, which solves a combinatorial problem that is appropriately modified to account for the current system state. The simulation is reprimed, with the state resulting from the optimizer, and it continues marching in time until its subsequent need for a control action. This architecture is presented in Figure 1. It should be noted that with respect to the system clock, the simulation module consumes time, while the optimizer module is an instantaneous departure for deciding on a control action. The simulation module marches in time, with an indeterminate number of departures to the optimizer, until a predetermined end of the planning horizon is reached, or a predetermined state like the exhaustion of all tasks in the pipeline is reached.

One such *controlled* walk in time through the stochastic state space constitutes a *time line*. A time line is thus a controlled trajectory that contains a unique get-together of various stochastic realizations present in the pipeline. Further, a time line contains information about the temporal coexistence of various feasible tasks, both within and across projects. This information, while being influenced by the nature of control actions exercised, is largely a function of uncertainty, and is not obtainable *a priori* from a single monolithic stochastic program due to uncertainties present in the task successes that are binary in nature, and those present in task processing times and task resource requirements. This coexistence information can be further processed to infer and identify resource types that are binding in the face of uncertainty, and to evaluate the worth of augmenting such resource types. It can also be used to obtain information about the relative tendencies of projects (and tasks) to crowd out other projects (and tasks) due to the associated variations in their resource needs. All the information that is obtained from the framework could be potentially incorporated in a suitable manner to bias the deterministic optimization formulation for the purposes of portfolio selection. Finally, multiple time lines can be explored in a Monte Carlo fashion to accumulate several unique combinations of realizations of uncertainty.

Figure 2 depicts such multiple time lines in the form of a stochastic tree. The root node can be thought to represent the initial state of the pipeline system. The internal nodes in the tree can be thought to represent calls that are triggered to the optimizer. The leaves of the tree can be thought to represent the end of the planning horizon. Furthermore, every unique path from the root node to a leaf node can be thought to represent one unique time line (a unique future) along which the R&D Pipeline could live through. Ordinary resource-constrained simulations like those implemented in packages such as Extend (Imagine That Inc.) and Awesim! (Symix Inc.) explore multiple stochastic realizations, too. Such ordinary simulations are not equipped with long-range decision-making capability. They need to use dispatching rules, in the form of a prioritized queue or some such queuing abstraction to resolve conflicts and to react to events. They are thus myopic if used alone, since such dispatching rules have available only local, current information at any decision-making point in simulation.

Sim-Opt overcomes this limitation in traditional simulation by taking the discrete-event dynamic system view of our stochastic system. Sim-Opt is, in a generic sense, the process of studying multiple *controlled* trajectories of a system through its stochastic state space. The control action within any single trajectory, or time line, is determined by solving suitable optimization formulations, which depend on the specific states that are witnessed within the time line. The optimization formulation at any state within any time line can be as simple as a set of dispatching/prioritizing rules, as is done in traditional simulation. At the same time, it can be a sophisticated mathematical program that takes into account current, local information, dynamic state-dependent information such as immediate resource competition, and the long-range effects of immediate actions by extending its considerations over a suitable horizon length. Thus traditional simulation can be viewed as one limit of the Sim-Opt approach; a limit where decisions are made in a myopic manner with local rules. Each time line in Sim-Opt is a potential evolution of reality in the system and is composed of events and corresponding decision responses that are exercised in a controlled fashion. Sim-Opt effectively generalizes traditional discrete-event simulation to embed sophisticated decision-making capability into simulation. Sim-Opt is different in terms of the treatment of the time lines. It represents a spectrum of decision making that varies according to the scope of decisions that need to be made at a given time. Sim-Opt is thus a decomposition-based approach to obtaining solutions to stochastic optimization problems. The inner loop, in Figure 1, studies multiple con-

trolled trajectories of the underlying discrete-event dynamic system in the face of rigorous uncertainty and realistic implementation. The information obtained from the multiple time lines can be integrated to exercise risk control and to obtain solutions to the underlying stochastic optimization problem. This is the intent of the outer loop, termed as the risk-control loop. The outer loop modifies the problem formulation, based on the knowledge obtained from the inner loop. It attempts to drive the controlled trajectories in the inner loop toward improving solutions with respect to probability distribution of the NPV in the system. A formal treatment of the outer loop is not conducted in this article, and will be addressed in a separate study. The inner loop and the accompanying integration of information is the focus of this study. This is demonstrated on an industrially motivated case study in a later section.

It should be noted that the principal benefit accruing from the application of Sim-Opt to the R&D Pipeline management problem is not any particular, single schedule of tasks. The foremost benefit is the assessment of uncertainty in the system. Information is obtained in terms of the relative performances of various strategic and tactical policies in the face of uncertainty and in terms of insights into the interactions resulting from resource conflicts between tasks, within and across projects. This information can be used to infer operational and design solutions for more effective management of the R&D Pipeline. Some of the questions that can be addressed with this framework are as follows:

1 What is a good set of projects to pursue?

2. What is a good way to assign resources to activities in the chosen projects?

3. What is the risk profile present in any answer to the preceding questions? What is the distribution of portfolio value?

4. How may the risk profile be "improved"?

5. Which Resources are limiting, and at what points in time?

6. What is the effect of inaccuracies in data estimates?

7. What is the value of outsourcing contracts for resources? When should we augment which type of resources to get a faster throughput of projects?

The questions that can be addressed with the Sim-Opt framework range from portfolio selection and scheduling to risk analysis and control, sensitivity analysis, and design insights.

Sim-Opt Implementation

Sim-Opt has been implemented as an object-oriented, scalable architecture using the C++ programming language. The computing recipe that the inner-loop implementation of Sim-Opt follows is given in Figure 3. First, an initial project portfolio is established. This supervisory control action is taken by solving a resource-overbooked optimization program based on the deterministic formulation presented in second section, wherein all random parameters are set to their respective expected values. The resulting mixed-integer linear program is formulated and solved in a C++ module using a commercial solver library, MILP++ (Advanced Process Combinatorics, Inc.). The solution contains a project portfolio and a project task schedule that is overbooked in terms of resources. While

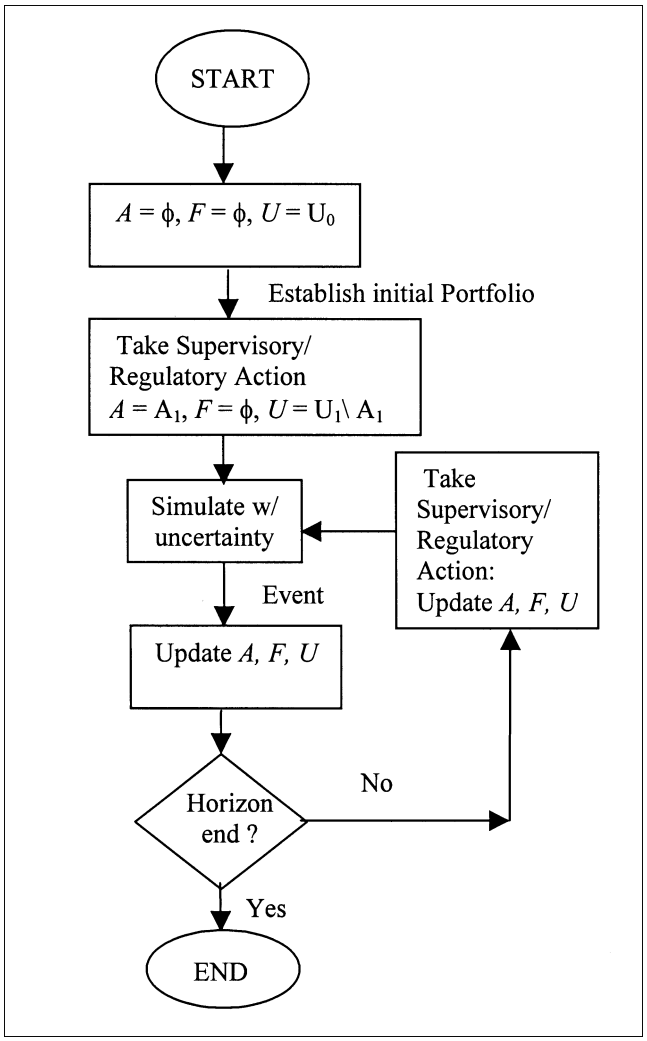

Figure 3. Sim-Opt computing algorithm.

such a task schedule is nonimplementable due to resource overbooking, the mathematical program provides decision making in terms of portfolio selection.

It should be noted that to begin with the set, $U = U_0$, is the set of all project tasks in the pipeline that have no technological predecessors. Subsequently, upon solving the MILP, the resulting scheduling solution is preprocessed to establish successor sets for all project tasks. More specifically, a successor set for any project task is the union of the set of tasks scheduled to start upon the finish of this task, and the set of tasks defined as technological successors to this task. The union of the set of tasks that are not members of any such successor set, and the set of tasks that are technologically feasible and belong to the chosen portfolio, forms the initial active set, $A = A_1$ ($A_1 \subseteq U_0$). This sequencing information, along with the pipeline problem definition, is fed into a discrete-event simulation module within the same executable. The simulation module is implemented in C++ using a commercial library, CSIM18 (Mesquite Software, Inc.). Graph-theoretic implementation of the generation of the MILP formulation as well as that of the discrete-event simulation ensures scala-

bility of the architecture, and is also key to the implementation of Sim-Opt. This is because the AoN graph data structure can be conveniently pruned and the information that it carries can be modified to correctly represent the state of the system. It can thus be effectively used to generate the corresponding MILP and simulation formulations at program execution run time.

It should be noted that the generation of the preceding formulations is determinable only at run time due to its dependence on the state of the system, and cannot be hardwired. Four types of regulatory and supervisory control actions are explored in this article. The type of control action exercised would constitute a policy for managing the R&D Pipeline. All four policies start with the initial portfolio being determined by the MILP described earlier. A description of the four policies is given below.

Policy 0: using the MILP rigorously

Policy 0 uses the MILP and the resulting scheduling solution to make both supervisory and regulatory decisions. Upon the successful finish or failed finish of an active task, that is, upon the occurrence of every event in the system, first, the sets A, F, and U are updated. The current state of the system is then communicated to the optimizer. The original MILP formulation is then modified to account for the current state of the system. This includes the following:

- Removal of all tasks belonging to projects that have failed thus far
- Removal of tasks that are contained in the finished set, F
- Addition of constraints to ensure the selection of on-going projects
- Addition of constraints to prevent the preempting of on-going tasks.
- Parameter updating for partially completed tasks in terms of estimates for processing times and resource requirements.

The solution to the modified MILP is preprocessed to form successor sets corresponding to the scheduled tasks, as described earlier. The union of tasks that are not members of any successor set, and tasks that are technologically feasible and belong to the chosen portfolio, form the active set to begin with, much like how the computing procedure began. This is conveyed to the simulation module to continue marching in time. The tasks in the active set, A, get queued for the various resource types in the simulation program, with their scheduled starting times serving as a priority key. Tasks that have the same scheduled starting times are relatively prioritized in terms of the expected NPV associated with their projects. The expected NPV is based upon the *critical path* of the corresponding AoN graph that remains at the current state of the system. The length of this critical path is calculated in an expected sense, using expected values of processing times for the AoN graph. An alternative way to calculate the critical path is to use the technique of computing the exact overall time distribution of a project with uncertain task durations, as developed in Schmidt and Grossmann (2000).

It should be noted that such an active set may not always contain tasks that are simultaneously feasible with respect to resource constraints, and hence not realistic. Resource starvation, occurring due to such resource conflicts, is tolerated for a fixed, nonzero interval of time beyond the scheduled starting times of such active tasks. Tolerating resource starvation for a nonzero interval of time is needed to prevent the degeneracy of infinite cycling between the optimizer and the simulator, since the optimizer uses nominal (mean) estimates of resource requirements for the scheduling problem, while the simulator uses values that are realized from the corresponding distributions. If starvation continues beyond the threshold of tolerance just given, it is treated as an event that needs decision making, and the current state of the system is communicated to the optimizer.

Policy I: using the MILP for resolving resource conflicts and for reacting to failure

Policy I is very similar to Policy 0 in that it uses the MILP and the resulting scheduling solution to make both supervisory and regulatory decisions. While Policy 0 makes a departure to the optimizer upon the occurrence of every event in the system, Policy I differs in the following manner. First, upon the successful finish of an active task, the sets A, F, and U are updated. Policy I augments the active set, A, with the union of the successor set (resulting from the most recent departure to the optimizer) of such a finished task, and that subset of the updated set, U, that belongs to the current portfolio. This updated input is sent to the simulation program that continues marching in time much like Policy 0. Resource starvation that may occur due to resource conflicts among tasks in the active set is handled as in Policy 0. The event of unsuccessful finish of an active task is handled entirely with the MILP by solving a formulation that is modified as described before. It proceeds much like how the computing procedure began.

Policy II: using a static knapsack problem for resolving resource conflicts and the MILP for reacting to failure

Policy II uses a combination of heuristics and mathematical programming to make both regulatory and supervisory decisions. First, upon the successful finish of an active task, the sets A, F, and U are updated. Policy II solves a 0–1 knapsack problem on that subset of the updated set, U, that belongs to the current portfolio. The reward coefficients used for the elements (tasks) of this subset are the expected NPVs that the tasks will lead to if started at the current state of the system. This expected NPV is based upon the expected *critical path* of the corresponding AoN graph that remains at the current system state. The amounts of the various resource types available at the current state of the system serve as righthand-side coefficients for the knapsack constraints, and the mean values of resource requirements of the tasks serve as lefthand-side coefficients. It is ensured that partially completed tasks are included in the knapsack solution. The tasks chosen in the knapsack solution get augmented into the active set, A, and get queued for the various resource types in the simulation program, with their reward coefficients serving as a priority key. Resource starvation that may occur due to uncertainty in the resource requirements (since the knapsack was solved with mean estimates serving as lefthand-side constraint coefficients), is tolerated until resources become available in an order determined by the priority key given

earlier, while not permitting any preemption of partially completed tasks. Task failure is handled entirely with the MILP, by solving a formulation that is modified as described in Policy 0.

Policy III: priority emphasis

Policy III uses a "greedy" heuristic to make both regulatory and supervisory decisions. First, upon the successful finish of any active task, the sets A, F, and U are updated. Policy III augments the active set, A, with that subset of the updated set, U, that belongs to the current portfolio. The tasks in the active set, A, get queued for the various resource types in the simulation program, with a priority key equal to the absolute-reward values attached to successful completion of the corresponding projects. Resource starvation occurring due to resource conflicts among the tasks in the active set is tolerated until resources become available in an order determined by the priority key, while it does not permit any preemption of partially completed tasks. When faced with supervisory de-

Table 1. Case Study Task Data

	Custom Distribution								
	Duration (Weeks)		RI (Units)		R2 (Units)		*Triangular Distribution Probability Success*		
Task	Value	Probability	Value	Probability	Value	Probability	Min	Most Likely	Max
I1	1	0.295	4	0.29	2	0.28	0.74	0.80	0.86
	2	0.375	5	0.44	3	0.44			
	3	0.190	6	0.21	4	0.28			
	4	0.110	7	0.06					
	5	0.030							
I2	2	0.10	4	0.06	1	0.22	0.7	0.75	0.8
	3	0.18	5	0.21	2	0.56			
	4	0.44	6	0.46	3	0.16			
	5	0.18	7	0.21	4	0.06			
	6	0.10	8	0.06					
P1	3	0.32	10	0.06	2	0.29	0.8	0.85	0.9
	4	0.40	11	0.12	3	0.44			
	5	0.18	12	0.55	4	0.21			
	6	0.10	13	0.21	5	0.06			
			14	0.06					
I3	1	0.335	3	0.05	1	0.23	0.7	0.8	0.85
	2	0.415	4	0.20	2	0.52			
	3	0.166	5	0.45	3	0.20			
	4	0.084	6	0.25	4	0.05			
			7	0.05					
I4	3	0.123	4	0.23	1	0.23	0.55	0.6	0.65
	4	0.203	5	0.52	2	0.55			
	5	0.335	6	0.20	3	0.16			
	6	0.228	7	0.05	4	0.06			
	7	0.111							
P2	4	0.32	10	0.23	2	0.23	0.75	0.8	0.85
	5	0.44	11	0.55	3	0.53			
	6	0.16	12	0.16	4	0.16			
	7	0.08	13	0.06	5	0.08			
I5	2	0.26	2	0.23	1	0.23	0.7	0.8	0.85
	3	0.54	3	0.55	2	0.53			
	4	0.14	4	0.15	3	0.17			
	5	0.06	5	0.07	4	0.07			
I6	3	0.10	4	0.23	1	0.225	0.7	0.75	0.8
	4	0.20	5	0.53	2	0.565			
	5	0.36	6	0.17	3	0.155			
	6	0.28	7	0.07	4	0.055			
	7	0.06							
P3	4	0.32	12	0.225	2	0.225	0.85	0.9	0.95
	5	0.40	13	0.555	3	0.555			
	6	0.18	14	0.160	4	0.160			
	7	0.10	15	0.060	5	0.060			
I7	3	0.335	3	0.23	2	0.23	0.85	0.9	0.95
	4	0.415	4	0.55	3	0.55			
	5	0.165	5	0.15	4	0.15			
	6	0.085	6	0.07	5	0.07			
I8	1	0.335	5	0.23	1	0.23	0.55	0.6	0.65
	2	0.415	6	0.57	2	0.57			
	3	0.165	7	0.15	3	0.15			
	4	0.085	8	0.05	4	0.05			

Table 1. Case Study Tasks Data (Continued)

	Custom Distribution						*Triangular Distribution Probability Success*		
	Duration (Weeks)		RI (Units)		R2 (Units)				
Task	Value	Probability	Value	Probability	Value	Probability	Min	Most Likely	Max
I9	1	0.32	4	0.23	1	0.22	0.65	0.7	0.75
	2	0.40	5	0.57	2	0.56			
	3	0.18	6	0.15	3	0.16			
	4	0.10	7	0.05	4	0.06			
P4	4	0.07	9	0.23	2	0.23	0.75	0.8	0.85
	5	0.17	10	0.53	3	0.53			
	6	0.52	11	0.17	4	0.17			
	7	0.17	12	0.07	5	0.07			
	8	0.07							
I10	3	0.33	4	0.23	3	0.23	0.7	0.85	0.85
	4	0.41	5	0.53	4	0.54			
	5	0.20	6	0.17	5	0.17			
	6	0.06	7	0.07	6	0.06			
II1	2	0.297	6	0.23	3	0.23	0.38	0.38	0.48
	3	0.456	7	0.54	4	0.53			
	4	0.197	8	0.17	5	0.17			
	5	0.05	9	0.06	6	0.07			
I12	1	0.32	5	0.23	1	0.23	0.38	0.38	0.48
	2	0.42	6	0.53	2	0.53			
	3	0.18	7	0.17	3	0.17			
	4	0.08	8	0.07	4	0.07			
P5	3	0.325	11	0.23	2	0.23	0.7	0.75	0.8
	4	0.375	12	0.53	3	0.53			
	5	0.150	13	0.17	4	0.17			
	6	0.100	14	0.07	5	0.07			
	7	0.050							
I13	3	0.305	4	0.23	3	0.23	0.7	0.75	0.8
	4	0.440	5	0.53	4	0.53			
	5	0.200	6	0.17	5	0.17			
	6	0.055	7	0.07	6	0.07			
I14	3	0.32	6	0.23	3	0.23	0.4	0.45	0.5
	4	0.42	7	0.53	4	0.53			
	5	0.18	8	0.17	5	0.17			
	6	0.08	9	0.07	6	0.07			
P6	3	0.088	13	0.20	3	0.22	0.6	0.65	0.7
	4	0.200	14	0.57	4	0.54			
	5	0.380	15	0.23	5	0.17			
	6	0.280			6	0.07			
	7	0.052							
I15	3	0.325	4	0.22	3	0.22	0.7	0.85	0.85
	4	0.425	5	0.54	4	0.54			
	5	0.175	6	0.17	5	0.17			
	6	0.075	7	0.07	6	0.07			
I16	1	0.283	6	0.22	5	0.22	0.45	0.5	0.55
	2	0.433	7	0.54	6	0.54			
	3	0.284	8	0.17	7	0.17			
			9	0.07	8	0.07			
I17	2	0.29	6	0.22	2	0.22	0.3	0.35	0.4
	3	0.44	7	0.54	3	0.54			
	4	0.21	8	0.17	4	0.17			
	5	0.06	9	0.07	5	0.07			
P7	3	0.29	13	0.20	3	0.22	0.45	0.5	0.55
	4	0.44	14	0.57	4	0.54			
	5	0.21	15	0.23	5	0.17			
	6	0.06			6	0.07			

cision making, Policy III augments the current portfolio with a new project for each failure encountered, in an order determined by the absolute-reward values attached to projects. The active set, A, is augmented with that subset of the updated set, U, that belongs to the updated current portfolio. The computing procedure returns to the simulation module and continues, as before, until the next control action is needed.

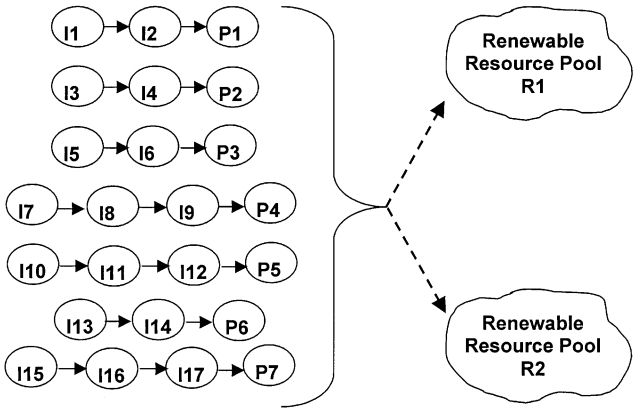

Figure 4. Case study AoN graph.

Sim-Opt Case Study

The implementation described in the previous section is demonstrated on an industrially motivated case study (Honkomp, 1998) in this section. The case comprises seven projects, with the Finish-to-Start precedence constraints within each project, as shown in the AoN graph in Figure 4. There are two renewable resource types, 16 units of type, R1, and 8 units of type, R2. The processing times (in weeks), the resource requirements, and the probabilities of success are given in Table 1 for each of the tasks. The rewards attainable upon successful completion of each of the projects are given in Table 2. All the reward is obtained upon successful completion of the final step. A planning horizon of 14 weeks is considered in this problem. As the time scale of the problem is in terms of weeks, a discount value of 1.73% per week is used for the calculations. Two classes of activities are considered for the calculation of rewards. For activities that have finished successfully, their rewards are time discounted to get a net present value. For activities that belong to on-going projects in the pipeline that have started but not conclusively completed (with respect to success/failure), an expected NPV is calculated that is based upon the *critical path* of the corresponding AoN graph that remains at the state of the system corresponding to the end of the planning horizon.

Triangular distributions are used in the case study because they are the easiest estimates to obtain from practitioners, who tend to have an idea about the maximum, minimum, most likely values of parameters that are open to variability. Custom distributions are also practically useful, as they can be constructed to represent historical information. Figures 5, 6, 7 and 8 show the frequency plots of rewards that are attainable from 15,000 time lines if the decision-making module adopts Policy 0, Policy I, Policy II, and Policy III, respectively. Using the idea of common random numbers (Law and Kelton, 1991), it is exactly guaranteed that the same unique 15,000 sets of random numbers are used for all four policies to ensure comparability. The computing times required for the four runs on a Windows NT Workstation with a Pentium II Processor, 256 MB RAM, and 450-MHz processor speed are ~ 79 h for Policy 0, ~ 32 h for Policy I, ~ 17 h for Policy II, and ~ 1.1 h for Policy III. An interesting piece of information available with Sim-Opt is the Value *with* Perfect Information (VPI). The VPI is defined as the value of the system assuming perfect prediction, that is, taking decisions with perfect knowledge of the future. This is done in Sim-Opt in any single time line by using the sampled values from various distributions for all sources of randomness, in the decision-making module. Clearly, if a project were to be fated to fail in any time line, it would not even be considered for portfolio selection and task scheduling in the case of VPI. The frequency plot of VPI is shown in Figure 9. It was noted before that the reward corresponding to any time line is obtained at the finish of the planning horizon from two terms: first, the

Table 2. Project Reward Data

Project	Reward ($)
P1	30,000
P2	20,000
P3	15,000
P4	40,000
P5	50,000
P6	40,000
P7	60,000

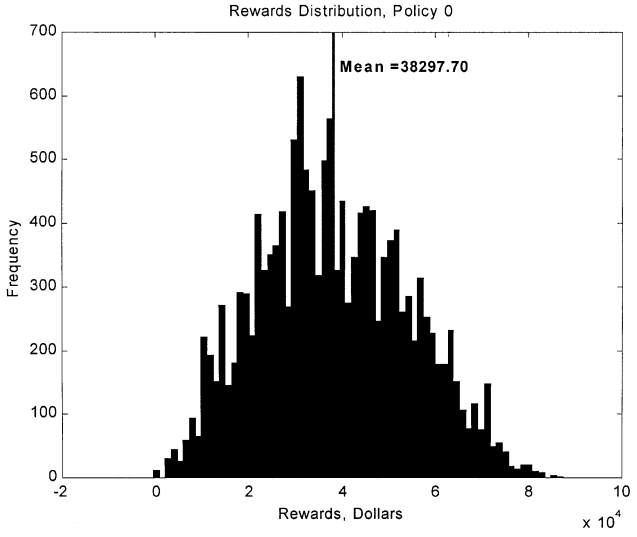

Figure 5. Results with Policy 0.

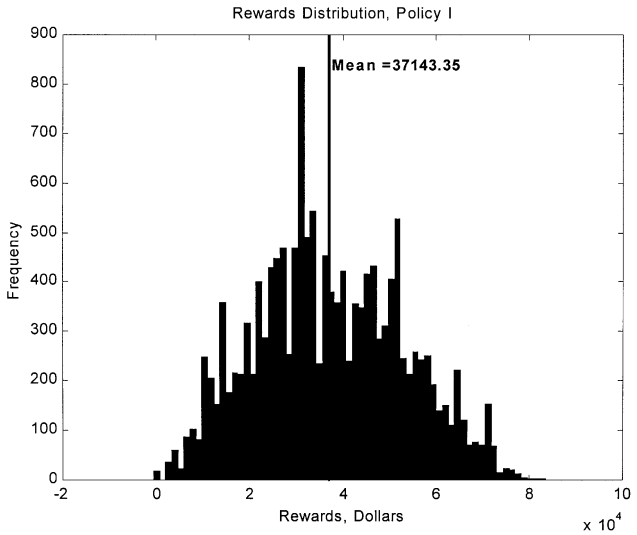

Figure 6. Results with Policy I.

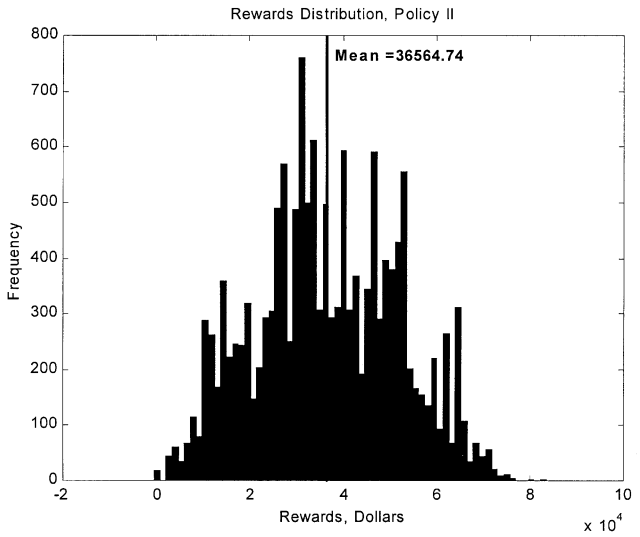

Figure 7. Results with Policy II.

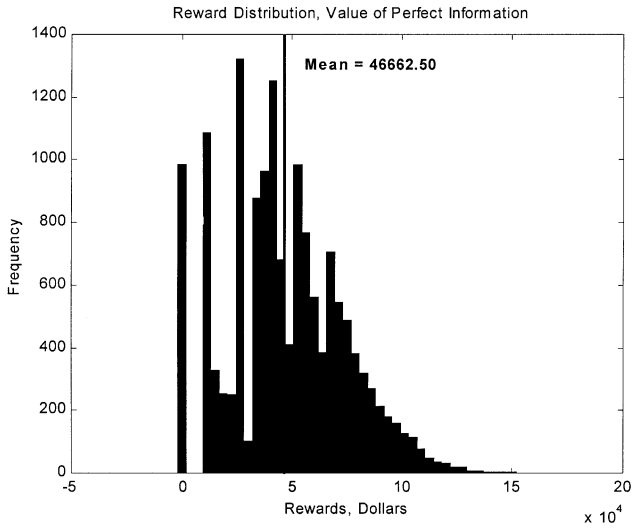

Figure 9. Reward distribution, value with perfect information.

realized value of successfully finished projects, and second, the expected value of on-going projects that have not completed conclusively with respect to success or failure.

The frequency of zero reward is notably lower in the distributions in Figures 5, 6, 7 and 8, as compared to Figure 9, due to the second term that is considered in the evaluation of the reward. For example, consider a time line that has upon the finish of the planning horizon an on-going project that has not completed conclusively, and is fated to fail in that time line. Such a project would not even be present in the case of VPI (it contributes a reward of zero), but will contribute an expected reward value in the case of Policies 0, I, II, and III. The frequency of zero reward in the VPI distribution represents the probability of all projects failing in the pipeline system. The distribution of VPI represents the variability inherent to the performance of the pipeline, and also represents an upper bound to the best performance obtainable from the pipeline.

Cumulative frequency plots are shown in Figure 10 for the rewards obtained from 15,000 time lines using Policies 0, I, II, III and VPI, respectively. At any reward value, the plot shows the frequentist cumulative probability of obtaining rewards less than or equal to that reward value. It can be seen that the greedy policy, Policy III, performs poorly as compared to the other policies when viewed in terms of the first moment (mean), as well as in terms of the cumulative probabilities. In terms of the mean, Policy 0 is better than Policy I, which in turn is better than Policy II. In terms of the cumulative probabilities, Policy 0 marginally outperforms Policy I, which in turn marginally outperforms Policy II.

Policies 0, I, II and III clearly differ in the amount of information that they use in determining the control action as a function of the corresponding state. Policy 0 differs from Policy I in the frequency with which reactive planning is done with the entire planning horizon taken into consideration.

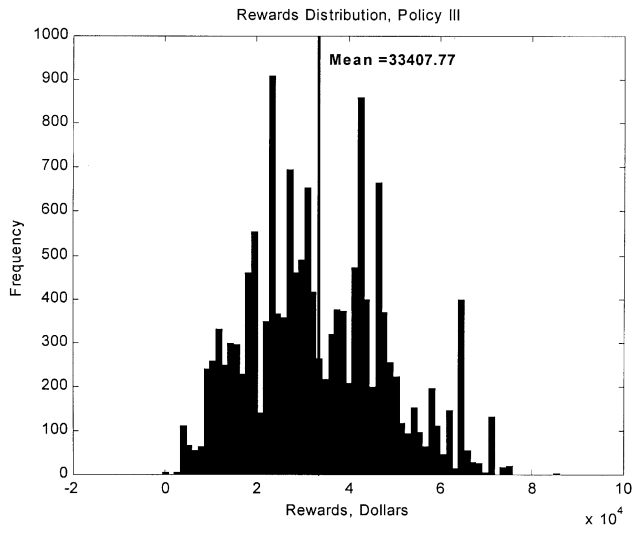

Figure 8. Results with Policy III.

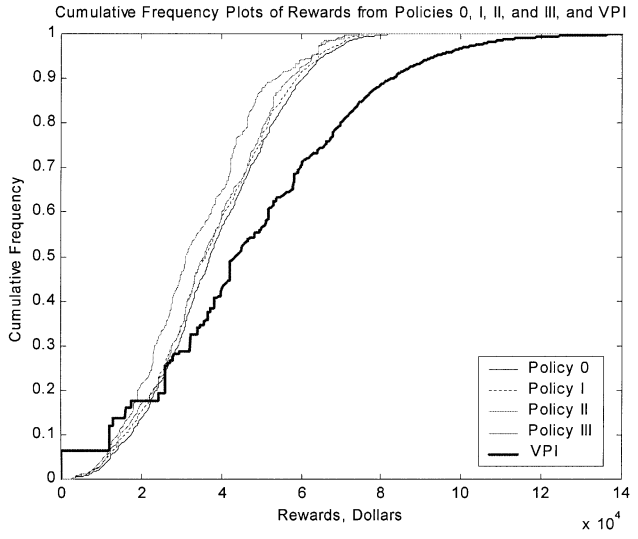

Figure 10. Cumulative frequency plots of rewards.

With respect to regulatory action, Policy 0 and Policy I differ from Policy II in that they consider the entire planning horizon in addition to the immediate resource competition among tasks that are feasible at the current state. In other words, they consider the temporal aspect in determining the control action, by considering the potential future implications of the current action in an expected sense. This is done by means of the horizon-length, resource-constrained, MILP. Policy II ignores this temporal aspect for determining control action, while considering only the immediate resource competition among feasible tasks by means of the knapsack formulation. With respect to supervisory action, Policies 0, I, and II are identical. Policy III ignores both the temporal aspect and the resource competition aspect in determining both types of control actions by making decisions in a "greedy" sense. Policy II is clearly appealing from the combined point of view of both computational effort and system performance.

Sim-Opt Information

In this section, we present methods for accumulating information from time lines. The information that can be retrieved from a time line centers on the coexistence in time of various feasible tasks, both within and across projects. This information, while being influenced by the nature of decision making that is exercised, is a function of uncertainty as well. Information can be obtained about which constraints (on resource types) are binding at which periods in time, before we can investigate the worth of augmenting such resource types.

Figure 11 shows the tasks that coexist in time, that is, tasks that have a nonzero, overlapping interval. Consider the following notation. Let set R be the set of renewable resource types, r and ρ_r be the maximum amount of renewable resource of type r, $\forall r \in R$. Define L as,

$$L([t_l,t_h]) = t_h - t_l, \tag{21}$$

that is, L gives the length of a time interval, where t_h and t_l represent the higher and lower termini of the time interval, respectively. Let A_j be an activity that attains technological feasibility at time, t_{l,A_j}, and has its processing finished at time, t_{h,A_j}, while requiring resource amounts, ρ_{r,A_j}, $\forall r \in R$, and let set, J, be the index set of such activities. If the length of the time line is restricted to a fixed planning horizon, $[0, H]$, there could be activities, A_j, that attain feasibility within the interval [0, H], but do not finish (or even start) within the interval $[0, H]$. This would be due to incomplete availability of the combination of resource types that the activity would need. For such activities, we use $t_{h,A_j} = H$, for the purposes of integrating the resource-binding information across the time lines. It should be noted that the attainment of feasibility of an activity coincides with the successful finish of its predecessor if it has predecessors, and coincides with the start of the planning horizon if it has no predecessors. Further, let A_i be an activity that starts processing at time, t_{l,A_i}, and finishes at time, t_{h,A_i}, while engaging resource amounts, ρ_{r,A_i}, $\forall r \in R$, and let set, I, be the index set of such activities. There could be activities, A_i, that start within the interval $[0, H]$, but do not finish within the interval $[0, H]$ because of their processing duration. For such activities, we use $t_{h,A_i} = H$, for the purposes of integrating the resource information across the time lines.

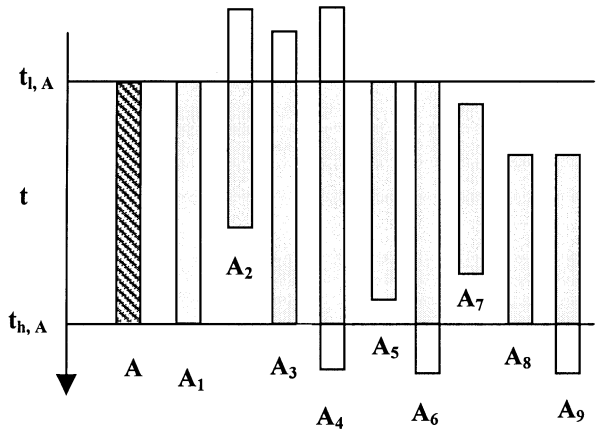

Figure 11. Coexisting tasks.

Define sets B_J^s, B_J^f, B_J, and C_J such that,

$$B_J^s = \left\{t_{l,A_j} | j \in J\right\} \tag{22}$$

$$B_J^f = \left\{t_{h,A_j} | j \in J\right\} \tag{23}$$

$$B_J = B_J^s \cup B_J^f \tag{24}$$

$$C_J = \{1, 2, 3, \ldots, 2|J|\}. \tag{25}$$

Let $f_J{:}C_J \rightarrow B_J$ be a map such that f_J is a one-to-one mapping and $f_J(1) \le f_J(2) \le \cdots \le f_J(2|J|)$. Note that f_J sorts the elements of set B_J in an ascending order. Let S_J be an ordered set of intervals such that

$$S_J = \{[f_J(i), f_J(i+1)] f_J(i), f_J(i+1) | i \in C_J,\ i+1 \in C_J,\ L([f_J(i), f_J(i+1)]) > 0\}, \tag{26}$$

and the ordering is a sorting with respect to the left terminus of the intervals. Note that set S_J contains intervals that demarcate resource engagement and disengagement. Let A_s^D be sets of activities such that

$$A_s^D = \left\{A_j | j \in J, L\left(s \cap \left[t_{l,A_j}, t_{h,A_j}\right]\right) > 0\right\} \qquad \forall s \in S_J. \tag{27}$$

We define a resource profile, termed as "desired profile," as a map $D_r{:}S_J \rightarrow \Re$, $\forall r \in R$, where $\Re$ is the set of real numbers given as

$$D_r(s) = \sum_{A \in A_s^D} \rho_{r,A}, \tag{28}$$

where A_s^D is defined in Eq. 27. We also define a resource profile termed as "utilized resource profile" as a map $U_r{:}S_I \rightarrow \Re, \forall r \in R$ in an identical manner by using set I, instead of

set J in Eqs. 22–28. This is given as

$$U_r(s) = \sum_{A \in A_s^U} \rho_{r,A}, \tag{29}$$

where A_s^U is defined as

$$A_s^U = \left\{ A_i | i \in I, L\left(s \cap \left[t_{l,A_i}, t_{h,A_i}\right]\right) > 0 \right\}, \qquad \forall s \in S_I. \tag{30}$$

Note that the definition of the "desired resource profile" given earlier is an estimate of the resource profile that we would ideally desire if we were to adopt the corresponding policy in the face of uncertainty, in order to push projects "as-fast-as-possible" for a given time line. The "utilized resource profile" is the actually engaged resource profile if we were to adopt the corresponding policy. Profiles D_r and U_r, $\forall r \in R$ can be accumulated across the time lines, and subsequently averaged to get mean "desired resource profiles" and "utilized resource profiles."

The Sim-Opt architecture tracks a fairly involved amount of information. This is needed, both within a time line to manage the temporal march that is punctuated by moving information backward and forward between the simulation and decision-making modules, and across time lines to manage the integration of time-line information. This necessitates efficient data structures to keep both the time complexity and space complexity of the implementation within control. The state of the discrete-event system is tracked with a set data structure that is implemented as a search tree (Cormen et al., 1990) to support operations Insert, Delete, and Member with time $O(\log n)$, operations Empty and Size with time $O(1)$, and operation Clear with time $O(n)$, where n is the current size of the set. The construction of resource profiles for the purposes of time-line integration from the Gantt-chart information that is obtained from a time line is done as follows. The Gantt-chart schedule is a collection of

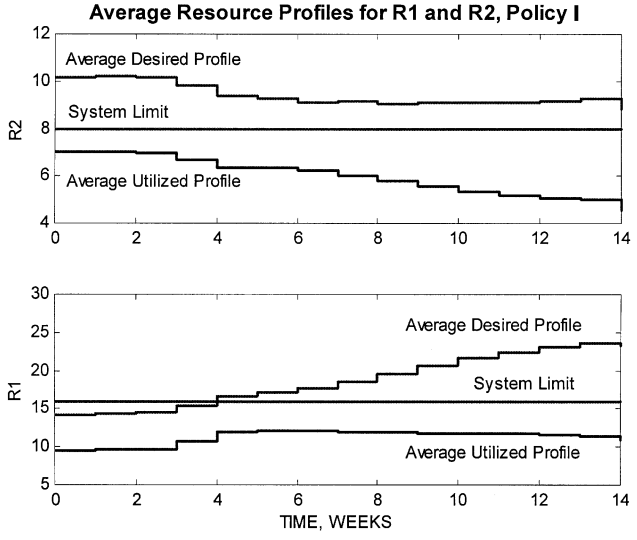

Figure 12. Resource profiles integrated across 15,000 time lines using Policy I.

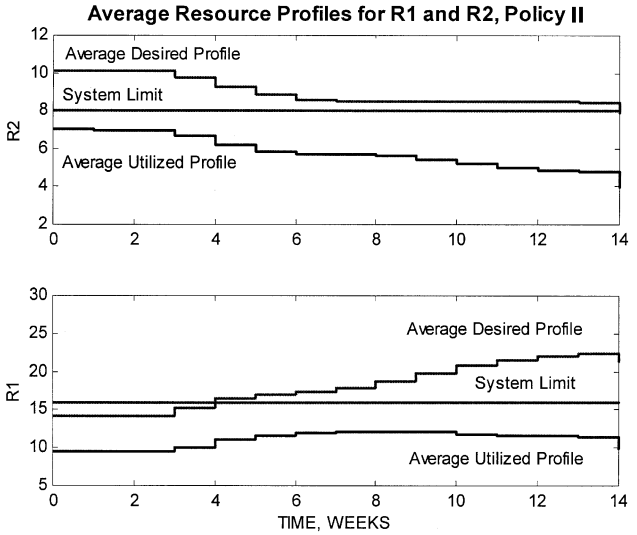

Figure 13. Resource profiles integrated across 15,000 time lines using Policy II.

segments that carry information about the amounts of various resource types that they need. This is viewed as a collection of points in a two-dimensional (2-D) space that has as axes (or components) the left terminus and the right terminus of the corresponding segment. Information required in Eq. 27 or Eq. 30 is obtained by storing these 2-D points in a geometric data structure, namely, the 2-D range tree (Van Kreveld et al., 1997), and by making a sequence of orthogonal range queries. Operations Insert, Lookup, and Delete take time $O(\log^2 n)$, operation Intersection takes time $O(k + \log^2 n)$, where k is the size of the returned list, and operation Clear takes time $O(n \log n)$, where n is the current size of the range tree. This data structure is also used to create the successor sets corresponding to any activity from the scheduling solution that results from the optimizer, as described in the fourth section. An alternative geometric data structure for efficiently finding all intervals that overlap a point is the interval skip list, as developed in Hanson (1991). It should be noted that the interval skip list is a randomized data structure with an expected performance guarantee, while the 2-D range tree is a deterministic data structure with a deterministic performance guarantee. Further, the simulation module carries in memory (or implements) only the technologically feasible portions of the R&D pipeline, that is, it simulates only tasks that are technologically feasible at any given point in time. This helps both the time and space complexities of the simulation module.

Integrating time-line information for resource profiles

We present results for the "desired resource profiles" and the "utilized resource profiles" that are accumulated and averaged across the 15,000 time lines for the example case study. Figures 12 and 13 show the profiles for the two resource types, R1 and R2, corresponding to Policy I and Policy II. It should be noted that the area between the system limit and the average utilized profile represents a measure of idle resources due to resource interaction and conflicts across projects.

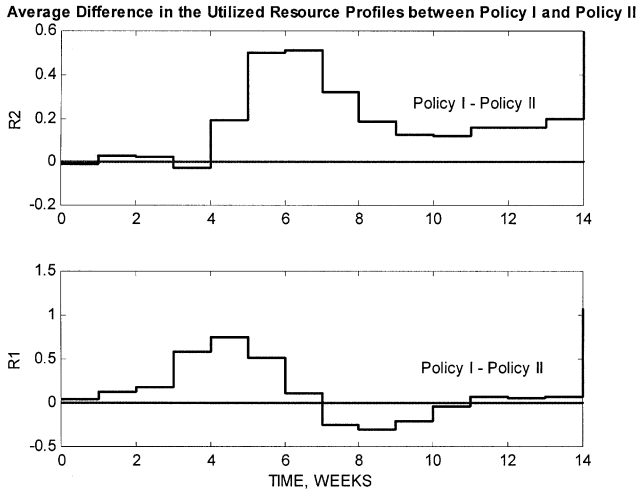

Figure 14. Average difference in resource utilization between Policy I and Policy II.

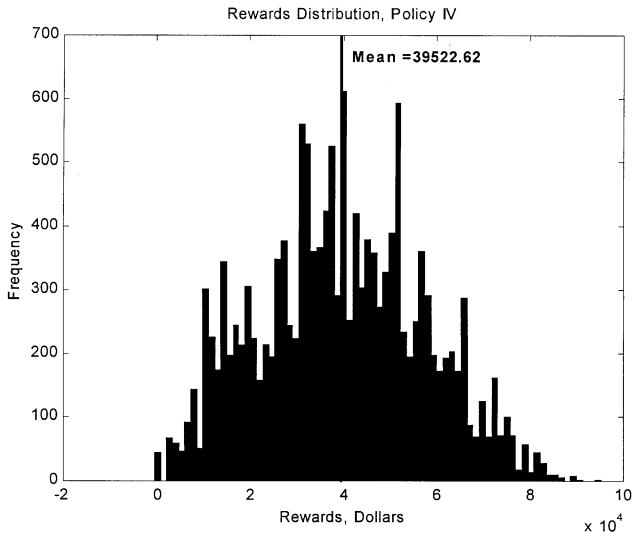

Figure 15. Results with Policy IV.

The plots in Figures 12 and 13 show the dynamics of combinatorial interaction between activities across projects in the portfolio. This interaction is due to resource competition. It can be seen that with respect to resource type, R1, the average "desired profile" stays within the system limit of 16 units during the earlier time periods in the planning horizon. But the actual utilization of resource type, R1, is below the desired level, as exhibited by the "utilized profile." While this may appear counterintuitive, it is because during the corresponding time periods, the average "desired profile" with respect to resource type R2 is well above the system limit of eight units. This prevents effective resource utilization, since the two resource types are required in the right combination. Activities in the pipeline are eligible for active processing only if the resource types R1 and R2 are available in the right combination of amounts. The plots can thus be viewed as the dynamics of interaction between resource types R1 and R2. During earlier time periods, resource type R2 is binding, while at later time periods, resource type R1 becomes binding as well. This combinatorial interaction at the current levels of availability of the resource types leads to poor utilization of the system resource levels. This knowledge can be incorporated to plan more effectively from both a design perspective and an operational perspective. The latter is illustrated with an example in the following section. Figure 14 highlights another subtle difference between Policy I and Policy II. Policy I gives a slightly better resource utilization on average, when compared with Policy II, as a result of the increased amount of information that it uses in terms of the temporal aspect to resource interaction between activities, in deciding on regulatory control action.

Example of using the information integrated across time lines

The resource profile information integrated across the time lines in Figures 12 and 13 revealed the underutilization of the resource types R1 and R2 due to their combinatorial interaction and how this interaction evolves in time. Having gained this insight, we can evaluate creative operational decisions, such as accumulating underutilized resource levels of a resource type from lean periods, and utilizing them for tight periods when the constraint on that resource type becomes binding. This is like underworking a resource type for some time periods, in return for overworking the same resource type for some other time periods. It is feasible under certain industrial circumstances, such as the use of contracted resources. Underutilized contracts can be utilized suitably at later points in time. We implement this operational strategy in Policy IV, which is same as Policy II in all other aspects. In Policy IV, accumulated resource levels are assigned to tasks along with actually present resource levels (if any), only if such an assignment fully satisfies the actual resource needs of the task for its entire processing duration. In particular, we accumulate underutilized resource levels, corresponding to resource types R1 and R2, in units of R1-Weeks and R2-Weeks, respectively. Figure 15 shows the frequency plot of rewards obtained from 15,000 time lines corresponding to the same unique 15,000 sets of random numbers, as used before. A cumulative frequency plot of the rewards obtained using Policy IV is shown in Figure 16, along with that corresponding to Policy II for comparison purposes. (Note that Policy IV can be thought of as Policy II implemented with an additional operational strategy, as described earlier.) Policy IV outperforms Policy II in terms of the mean as well as in terms of the cumulative frequencies. Figure 17 shows the "desired resource profiles" and the "utilized resource profiles" that are accumulated, and averaged across the 15,000 time lines explored using Policy IV. While the dynamics of combinatorial interaction between the resource types at the current resource levels in the system continue to prevent effective utilization of the available resource levels, the extent of underutilization has improved significantly over what was witnessed in Policy II (see Figure 13). This can be seen from Figure 18, which shows the difference in the average resource utilization between Policy IV and Policy II. This is an example of how the insight obtained with the integration of information across time lines can be utilized effectively to

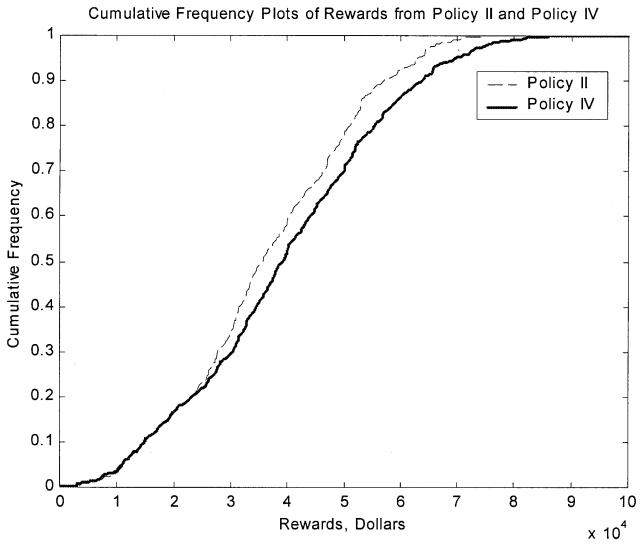

Figure 16. Cumulative frequency plots of rewards for Policy IV and Policy II.

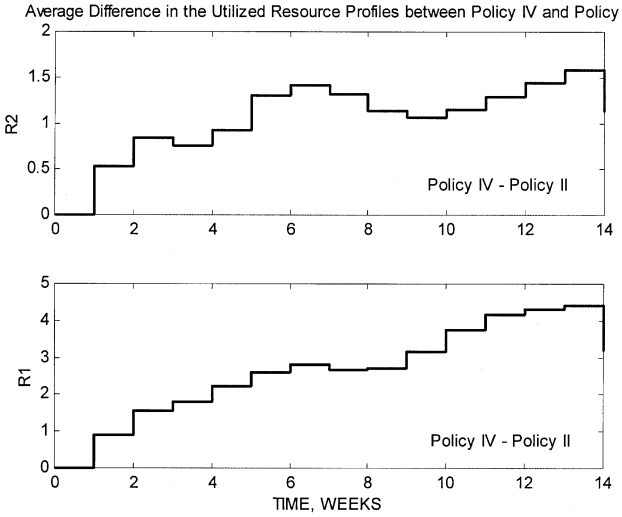

Figure 18. Average difference in resource utilization between Policy IV and Policy II.

influence and improve the quality of time lines that the pipeline system can witness.

Conclusions and Future Study

The here-and-now stochastic optimization problem inherent to the management of an R&D Pipeline has been described in its most general form as the control problem of a performance-oriented, resource-constrained, stochastic, discrete-event, dynamic system (DEDS). With the preceding perspective serving as guidance, a computing architecture, Sim-Opt, has been developed that combines combinatorial optimization and discrete-event system simulation to assess the uncertainty and control the risk present in the pipeline. The concept of time lines that studies multiple, unique realizations of the controlled evolution of the discrete-event pipeline system has been described. Sim-Opt presents a decomposition-based, practical approach to obtaining realistic solutions to effectively managing the underlying stochastic discrete-event system. It effectively generalizes traditional discrete-event simulation by adding sophisticated decision-making capability to simulation. Four different implementations of the determination of regulatory and supervisory control actions in Sim-Opt have been described and studied through an example case study. Finally, methods have been presented to integrate information across the time lines from this computing apparatus, in terms of binding resource types that present bottlenecks. An example has been presented that evaluates operational decisions, such as accumulating underutilized resource levels of a resource type from lean periods, and utilizing them for tight periods when that resource type becomes binding. This is like underworking a resource type for some periods in return for overworking the same resource-type for some other periods. With the help of the preceding time line integration, we can identify time periods when a certain resource type becomes binding. We can evaluate design decisions, such as the value of acquisition of any resource type(s), the value of entering into outsourcing contracts for binding resource types. and estimate the timing of these future contracts at the here-and-now. Operational decisions such as partially satisfying the resource needs of resource-starved activities, accompanied by a proportional increase in their processing times, can also be studied with the time-line integration within the Sim-Opt architecture. Finally, all such information can be utilized toward more effective decision making in order to improve the quality of time lines that the pipeline system can witness.

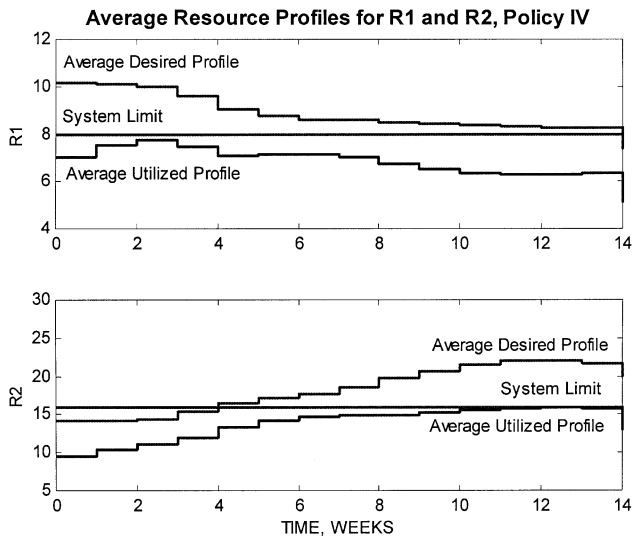

Figure 17. Resource profiles integrated across 15,000 time lines using Policy IV.

Literature Cited

Advanced Process Combinatorics, Inc., available on the Web at http://www.combination.com/

Blau, G. E., "A Systems Engineering Approach to New Product Development," CAST Communications, (1997).

Blau, G. E., B. Mehta, S. Bose, J. F. Pekny, G. Sinclair, K. Kuenker, and P. Bunch, "Risk Management in the Development of New

Products in Highly Regulated Industries," *Comput. Chem. Eng.*, **24**(2–7), 659 (2000).
Brucker, P., A. Drexl, R. Mohring, K. Neumann, and E. Pesch, "Resource-Constrained Project Scheduling: Notation, Classification, Models, and Methods," *Eur. J. Oper. Res.*, **112**(1), 3 (1999).
Cassandras, C. G., *Discrete Event Systems: Modeling and Performance Analysis*, Aksen:Irwin, Homewood, IL (1993).
Cormen, T. H., C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, McGraw-Hill, New York, p. 263 (1990).
Garey, M. R., and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of Np-Completeness*, Freeman, San Francisco (1979).
Gonzalez, R., and M. J. Realff, "Operation of Pipeless Batch Plants —I. MILP Schedules," *Comput. Chem. Eng.*, **22**(7–8), 841 (1998).
Hanson, E. N., "The Interval Skip List: A Data Structure for Finding all Intervals that Overlap a Point," *Lect. Notes Comput. Scie.*, **519**, 153 (1991).
Honkomp, S. J., "Solving Mathematical Programming Planning Models Subject to Stochastic Task Success," PhD Thesis, School of Chemical Engineering, Purdue Univ., West Lafayette, IN (1998).
Honkomp, S. J., L. Mockus, and G. V. Reklaitis, "A Framework for Schedule Evaluation with Processing Uncertainty," *Comput. Chem. Eng.*, **23**(4–5), 595 (1999).
Jain, V., and I. E. Grossmann, "Resource-Constrained Scheduling of Tests in New Product Development," *Ind. Eng. Chem. Res.*, **38**(8), 3013 (1999).
Kalagnanam, J. R., and U. M. Diwekar, "An Efficient Sampling Technique for Off-Line Quality Control," *Technometrics*, **39**(3), 308 (1997).
Kall, P., and S. W. Wallace, *Stochastic Programming*, Wiley, New York (1994).
Law, A. M., and D. W. Kelton, *Simulation Modeling and Analysis*, 2nd ed., McGraw-Hill, New York, p. 582 (1991).
Mesquite Software, Inc., available on the Web at http://www.mesquite.com/
Reklaitis, G. V., "Overview of Scheduling and Planning of Batch Process Operations," NATO ASI on Batch Process Systems Engineering, Antalya, Turkey (1992).
Schmidt, C. W., and I. E. Grossmann, "Optimization Models for the Scheduling of Testing Tasks in New Product Development," *Ind. Eng. Chem. Res.*, **35** (10), 3498 (1996).
Schmidt, C. W., and I. E. Grossmann, "The Exact Overall Time Distribution of a Project with Uncertain Task Duration," *Eur. J. Oper. Res.*, **126**(3), 614 (2000).
Van Kreveld, M., M. Overmars, O. Schwarzkopf, and M. de Berg, *Computational Geometry: Algorithms and Applications*, M. De Berg, ed., Springer-Verlag, Berlin (1997).

Manuscript received July 19, 2000, and revision received Apr. 23, 2001.