

Optimal Resource Investment and Scheduling of Tests for New Product Development

*Christos T. Maravelias, Ignacio E. Grossmann**

Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA15213

April 2003 / July 2003

Abstract

Previous optimization models for the scheduling of tests for pharmaceuticals and agrochemicals, assume that the resources available for testing such as laboratories and scientists are constant throughout the testing horizon, and that all testing tasks have fixed cost, duration and resource requirements. In order to be able to handle more effectively a number of potential products in the R&D pipeline a company may consider the option to install/hire additional resources. Also, the type and amount of resources can be chosen for some tests to reduce their duration. In this paper we propose a new scheduling MILP model that addresses these issues to optimize the overall costs. More specifically, the model (a) allows for installation of new resources during the course of testing, and (b) handles cost and duration of tests as functions of the type and amount of the resources assigned to each test. To enhance the solution of the model we develop, (a) a reformulation with disaggregated variables, (b) a procedure for the tightening of the big-M constraints, (c) a procedure that fixes some of the sequencing binary variables, and (d) logic cuts. The solution of a single large-scale problem is avoided with a heuristic decomposition algorithm that relies on solving a reduced mixed-integer program that embeds the optimal schedules obtained for the individual products. The proposed algorithm is shown to be one to two orders of magnitude faster than the full space method, yielding solutions that are optimal or near optimal.

1. Introduction

In highly regulated industries, such as agrochemical and pharmaceutical, new products have to pass a number of regulatory tests to gain FDA approval. If a product fails one of these tests it cannot enter the market and the investment in previous tests is wasted. Depending on the nature of the products, testing may last up to 10 years (Gardner et al., 2003), and the scheduling of tests should be made with the goal of minimizing the time to market and the expected cost of the testing. The problem of scheduling testing tasks in new product development has been studied by several authors. Schmidt and Grossmann (1996) proposed various MILP optimization models for the scheduling of testing tasks with no resource constraints. The basic idea in this model is to use a

* Author to whom correspondence should be addressed. E-mail: grossmann@andrew.cmu.edu, Phone: 412-268-2230

discretization scheme in order to induce linearity in the cost of testing. Jain and Grossmann (1999) extended these models to account for resource constraints. Honkomp et al. (1997) addressed the problem of scheduling R&D projects, which is very similar to the one of scheduling testing tasks for new products. Subramanian et al. (2001) proposed a simulation-optimization framework that takes into account uncertainty in duration, cost and resource requirements and extended this model to account for risk (Subramanian et al., 2003). Maravelias and Grossmann (2001) proposed an MILP model that integrates the scheduling of tests with the design and production planning decisions. Rogers et al. (2002) and Papageorgiou et al. (2001) have addressed related problems. A literature review of optimization approaches in the supply chain of pharmaceutical industries can be found in Shah (2003).

A common assumption in all these approaches is that the available resources are constant throughout the testing period. In Schmidt and Grossmann (1996), Jain and Grossmann (1999), Subramanian et al. (2001), Maravelias and Grossmann (2001), (2003), and Subramanian et al. (2003) the authors develop models for the optimal scheduling of testing tasks assuming that the available resources are constant, and usually that there is only one resource for each task. In Papageorgiou et al. (2001) and Rogers et al. (2003) the authors develop models assuming that enough resources are always available. In reality, however, the availability of resources is often the bottleneck for launching new products in a timely fashion. Therefore, a company may decide to hire more scientists or build more laboratories to handle more effectively a great number of potential new products in the R&D pipeline. As another option the company may have to outsource the tests often at a high cost. Another assumption common in all these approaches is that the cost and the duration of one test do not depend on the amount of resources allocated to one test. In reality, however, there is sometimes the option of allocating more resources to some tests in order to reduce the length of the critical path that determines the duration of the testing process.

In this work we propose a MILP optimization model that addresses the above issues. Specifically, the proposed model allows: (a) “installation” of new resources (e.g. hiring of new scientists) for a given set of products that are at different stages of the testing, with the option of outsourcing also included; (b) it handles the type and level of resources assigned to a test as a decision variable. This in turn implies that the cost and the duration of each test may be variables. The main trade-off is between the higher cost of testing (caused by the “installation” of additional resources, the utilization of outsourcing and the allocation of more resources) and the higher income from sales due to shorter completion times that result from eliminating bottlenecks.

It should be noted that the duration, cost and probability of success of testing tasks are highly uncertain in practice. A multistage stochastic MILP that accounts for all the uncertain parameters can in principle address this issue. However, such a model is very difficult to formulate and solve (Birge and Louveaux, 1997). Even a simpler two-stage optimization model with multiple scenarios is non-trivial. Therefore, in order to eventually solve in an efficient way a stochastic optimization model it is essential to first have an efficient deterministic model which is the focus of this paper. Specifically, in order to improve the efficiency of solving the MILP model we propose to tighten the big-M parameters through time-window calculations and fix subsets of binary

variables via a graph-theoretic preprocessing and logic implication. We show that tighter LP relaxation can be obtained by a reformulation with disaggregated variables and through the addition of logic cuts. Finally, we build on the decomposition algorithm by Maravelias and Grossmann (2003) that was proposed for the standard resource-constrained scheduling problem and that yields order of magnitude improvements in the computation. Several examples are presented to illustrate the application of the proposed model and the computational performance of the proposed method.

2. Motivating Example

To illustrate the trade-offs and the importance of resource management, consider the example of an R&D pipeline with two potential pharmaceutical products. Product A has passed successfully Phase I clinical trials, while product B is a new drug that passed successfully pre-clinical studies. Currently, due to limited resources, two tests of the same type cannot be performed simultaneously. To overcome this barrier, we assume that the company can install additional laboratories. We also assume for illustration purposes that outsourcing is not feasible. The expected duration of the remaining clinical trials for both drugs are given in Table 2, while the testing costs are given in Table 2.

Table 1: Durations of tests for the motivating example.

Duration (months)	Phase I	Phase II	Phase III
A	-	8	14
B	6	10	12

Table 2: Testing costs for the motivating example.

	Phase I	Phase II	Phase III
Fixed Cost for product A (\$10 ⁶)	-	5	15
Fixed Cost for product B (\$10 ⁶)	3	6	12
Cost of installing new resources (\$10 ⁶)	-	5	10

If no additional laboratories are installed, the testing completion time of drug A is 22 months and the completion time of drug B is 34 months. The total non-discounted cost of testing (assuming that Phase I and Phase II clinical trials will always be successful) will be \$41,000,000 (the schedule of this case is shown in Figure 1a). If additional laboratories are installed for Phase III clinical trials, the completion time of drug B can be shortened to 30 months, but the total cost of testing increases to \$51,000,000 (Figure 1b). Finally, if additional laboratories are installed for both Phase II and Phase III clinical trials, the completion time for product B is reduced to 28 months and the total testing cost is \$56,000,000. In all cases reduced completion times will potentially yield larger income from sales and larger market share.

This example then shows that, in general, there is a trade-off among the (expected) cost of testing, the installation of new resources and the completion time of testing. The problem becomes even more difficult when outsourcing is available for certain tests, and resource constraints and technological precedences are also to be

taken into account. Thus, to maximize the Net Present Value of simultaneous projects, we need a systematic optimization model that takes into account all these aspects.

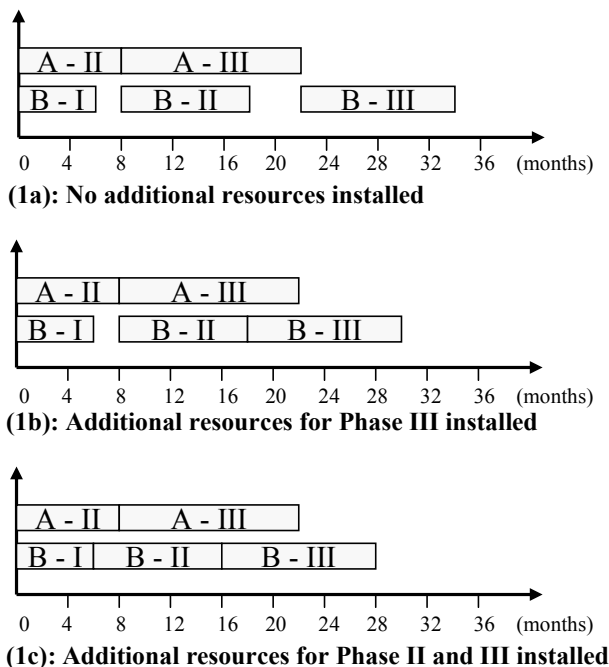


Figure 1: Alternative schedules for the motivating example.

3. Problem Statement

Given is a set of potential products that are in various stages of the company’s R&D pipeline and a set of resources to complete the testing tasks. Each potential product is required to pass a series of tests. Each test has a given duration, cost and probability of success that is assumed to be known a priori. In some cases the duration and the cost of each test may be a function of the resource allocation. If needed, a test may be outsourced at a higher cost. Resources can also be installed at a known cost. Resources are discrete in nature (e.g. laboratories and scientists) and each resource unit can handle only one task at a time. Tests are assumed to be non-preemptive. Due to technological precedences some tests can only be performed after the completion of other tests. There are three main decisions regarding the resource management and the scheduling of tasks: (a) the decision to install a resource unit ($w_q = 1$ if resource q is “installed”, $b_q =$ time of installation), (b) the sequencing ($v_{kk'} = 1$ if test k finishes before test k') and timing ($s_k =$ start time of test k) of tests, (c) the amount of resources allocated to a test ($N_{rk} =$ number of resource units of category r allocated to test k) and the assignment of resource units to tests ($x_{kq} = 1$ if resource unit q is assigned to test k). The option of outsourcing is also considered by postulating additional resource units. The objective is to maximize the Net Present Value (NPV) of multiple projects, i.e. the income from sales minus the expected cost of testing.

It should be noted that other tasks, besides testing tasks, have to be completed before the commercialization of a new agrochemical or pharmaceutical product. Such tasks are the process development stage, the procurement

and validation of the equipment, the construction or retrofit of the manufacturing facilities and the qualification runs. These non-testing tasks have to be carried out in parallel with (or after) the testing tasks and resource scarcity for these tasks would also lead to delays in the commercialization of the new products. To account for this we model them as testing tasks with specific characteristics. The process development stage, for instance, can be modeled as a testing task that requires a specific type of resource (process development group), with a probability of success equal to 1 and for which outsourcing is not allowed. Qualification runs can be modeled as testing tasks that should be performed after the completion of all the other tasks. Note that in a case study by Subramanian et al. (2003), prelaunch, ramp up sales and mature sales are also modeled as testing tasks with specific resource requirements, duration, costs and revenues. It is because of some of these tasks that the variability in resource allocation should be included in a general testing model. Specifically, while clinical trials may often have fixed duration and resource requirements, many of the non-testing tasks have variable duration/resource requirements. The process development stage and the construction of manufacturing facilities, for example, can be expedited if more resources are used. The latter one, specifically, appears to become a very important task whose timing and duration are very important.

We would also like to add that the proposed model is quite general and applicable to a number of industries. For instance, if applied to an agrochemical company, where each product has to undergo many short tests whose duration and cost are usually well known, the model can be simplified by assuming constant test durations and resource requirements. If applied on a pharmaceutical company, where the sequence of tasks for products close to commercialization can assumed to be known, the model can be simplified by ignoring sequencing decisions.

4. Proposed Model

The proposed model is a generalization of the resource-constrained MILP model by Jain and Grossmann (1999). It includes the following additional features:

- (a) The available resources are not fixed. Additional resources can be installed during the course of testing if needed.
- (b) The duration and the cost of each test are not necessarily constant parameters but can be functions of the amount and type of the allocated resources; the duration of a cost can be reduced by allocating more resources at the expense of a higher cost.

In the proposed model, J is the set of new products, K is the set of tests, R is the set of resource categories, Q is the set of resource units, and $K(j)$ represents the set of tests to be performed for product $j \in J$ ($K = \cup_{j \in J} K(j)$). The set of tests that belong to the same product as test k is denoted by $KK(k)$. Each resource unit $q \in Q$ belongs to a resource category, and $Q(r)$ is the set of resource units that belong to category $r \in R$ ($Q = \cup_{r \in R} Q(r)$). The set of tests that can be assigned to unit q is denoted by $K(q)$. Outsourcing is represented through dummy resource units for which there are no resource constraints (i.e. many tests can be outsourced at the same time). QE is the set of existing resources, QN is the set of potentially newly installed units, and QD are the dummy units for outsourcing ($Q=QE \cup QN \cup QD$). The nomenclature is given at the end of the paper.

4.1. Timing and Sequencing Constraints

Constraint (1) ensures that the testing completion time, T_j , of product j is greater than the completion time of any task k required for product j , where d_k is the duration of test k . Constraints (2) fix sequencing binaries $y_{kk'}$ for pairs (k, k') for which a technological precedence exists (i.e. $(k, k') \in A$). Time sequencing is enforced through constraints (3) and (4), where U is a big-M parameter. The duration of test k , d_k , is calculated in equation (5) as a function of the amount of resources N_{rk} allocated to test k . For the most common case where durations are fixed the parameters δ_{rk} are set to zero. Constraint (6) ensures that if the new resource unit $q \in QN$ is assigned to test k (i.e. $x_{kq}=1$), then the starting time of test k is larger than the installation time b_q of resource unit q :

$$s_k + d_k \leq T_j \quad \forall j, \forall k \in K(j) \quad (1)$$

$$y_{kk'} = 1, \quad y_{k'k} = 0 \quad \forall (k, k') \in A \quad (2)$$

$$s_k + d_k \leq s_{k'} \quad \forall (k, k') \in A \quad (3)$$

$$s_k + d_k \leq s_{k'} + U(1 - y_{kk'}) \quad \forall k, \forall k' \in KK(k) \mid (k, k') \notin A \quad (4)$$

$$d_k = d_k^{MAX} - \sum_r \delta_{rk} N_{rk} \quad \forall k \quad (5)$$

$$s_k \geq b_q - U(1 - x_{kq}) \quad \forall q \in QN, \forall k \in K(q) \quad (6)$$

4.2. Resource Constraints

The number of resource units of category r allocated to test k , N_{kr} , has to lie between a lower (N_{kr}^{MIN}) and upper (N_{kr}^{MAX}) bound [constraint (7)], where $R(k)$ is the set of resource categories needed for test k . Constraint (8) enforces that exactly N_{kr} discrete resource units are allocated to test k , including outsourcing. Constraint (9) ensures that there is a precedence between tests k and k' if are both assigned to resource unit q (see Maravelias and Grossmann, 2001). Constraint (9) is expressed for existing ($q \in QE$) and potentially new ($q \in QN$) resource units, but not for the dummy resource units ($q \in QD$) that represent the outsourcing option. Constraint (4*) enforces a precedence among tests of different products that can be scheduled on the same resource unit. The logical condition that a potentially new unit $q \in QN$ cannot be assigned to any test ($x_{kq}=1$) if it is not installed ($w_q=1$) is expressed by constraint (10).

$$N_{kr}^{MIN} \leq N_{kr} \leq N_{kr}^{MAX} \quad \forall k, \forall r \in R(k) \quad (7)$$

$$\sum_{q \in Q(r)} x_{kq} = N_{kr} \quad \forall k, \forall r \in R(k) \quad (8)$$

$$x_{kq} + x_{k'q} - y_{kk'} - y_{k'k} \leq 1 \quad \forall q \in (QE \cup QN), \forall k \in K(q), \forall k' \in K(q) \quad (9)$$

$$s_k + d_k \leq s_{k'} + U(1 - y_{kk'}) \quad \forall q \in (QE \cup QN), \forall k \in K(q), \forall k' \in K(q) \setminus KK(k) \quad (4^*)$$

$$x_{kq} \leq w_q \quad \forall q \in QN, \forall k \quad (10)$$

4.3. Testing Cost Constraints

The cost of testing consists of three elements: (a) a fixed cost for each task, (b) a cost associated with the use of particular resource units, and (c) the cost of resource installation. Since the scheduling horizon is commonly 2-10 years all costs should be discounted and a discounting rate ρ has been used. The discounting introduces nonlinearities and in order to avoid solving a large scale MINLP, we use a piece-wise linearization scheme to approximate the exponential discounting function. The cost elements for (a), (b) and (c) are approximated through grid points α_n , and weights λ^1_{kn} , λ^2_{kqn} and λ^3_{qn} , respectively. The linearization factors are activated through equations (11), (12) and (13). For each test k , the fixed cost is always paid [constraint (11)], while the cost associated with the use of a particular resource unit q is paid only if unit q is assigned to test k [constraint (12)]; similarly, the installation cost for a resource unit $q \in QN$ is only paid if the unit is installed [constraint (13)]. Linearization weights are calculated through equations (14), (15) and (17), where the cost of each test is discounted to reflect both the time-value of money (term $-\rho \cdot s_k$ or $-rb_q$), and the probability of conducting this test (term $\sum_{k' \in KK(k)} \ln(p_k) y_{k'k}$). Since the RHS of constraint (15) is always non-zero and the summation of $\lambda^2_{kqn} \alpha_n$ over n in the LHS may be zero [due to (12)], the slack variable sl_{kq} has been added. This variable is non-zero only when $x_{kq} = 0$ (constraint 16). The details of the linearization scheme can be found in Schmidt and Grossmann (1996).

$$\sum_n \lambda^1_{kn} = 1 \quad \forall k \quad (11)$$

$$\sum_n \lambda^2_{kqn} = x_{kq} \quad \forall q, \forall k \in K(q) \quad (12)$$

$$\sum_n \lambda^3_{qn} = w_q \quad \forall q \in QN \quad (13)$$

$$\sum_n \lambda^1_{kn} \alpha_n = -\rho \cdot s_k + \sum_{k' \in KK(k)} \ln(p_k) y_{k'k} \quad \forall k \quad (14)$$

$$\sum_n \lambda^2_{kqn} \alpha_n - sl_{kq} = -\rho \cdot s_k + \sum_{k' \in KK(k)} \ln(p_k) y_{k'k} \quad \forall q, \forall k \in K(q) \quad (15)$$

$$sl_{kq} \leq \alpha_{|N|} (1 - x_{kq}) \quad \forall q, \forall k \in K(q) \quad (16)$$

$$\sum_n \lambda^3_{qn} \alpha_n = -\rho \cdot b_q \quad \forall q \in QN \quad (17)$$

4.4. Objective Function Calculations

The total fixed (*CT1*), resource-dependent (*CT2*) and installation (*CT3*) costs are calculated through constraints (18)-(20), where cf_k , cq_{kq} and cb_q are the non-discounted costs. The income is calculated through constraints (21) and (22), where parameters g_{jm} and f_{jm} are used to represent income as a piecewise linear decreasing function of time. Alternative objective functions that can easily be handled are minimization of the testing completion time and the minimization of testing cost. The objective in (23) is to maximize the NPV of multiple projects:

$$CT1 = \sum_j \sum_{k \in K(j)} cf_k \sum_n \lambda_{kn}^1 e^{a_n} \quad (18)$$

$$CT2 = \sum_j \sum_{k \in K(j)} \sum_q cq_{kq} \sum_n \lambda_{kqn}^2 e^{a_n} \quad (19)$$

$$CT3 = \sum_{q \in QN} cb_q \sum_n \lambda_{qn}^3 e^{a_n} \quad (20)$$

$$u_{jm} \geq T_j - g_{jm} \quad \forall j, \forall m \quad (21)$$

$$INC_j = INC_j^{MAX} - \sum_m f_{jm} u_{jm} \quad \forall j \quad (22)$$

$$\max NPV = \sum_j INC_j - CT1 - CT2 - CT3 \quad (23)$$

$$y_{kk'}, x_{kq}, w_q \in \{0, 1\}, N_{rk} \in \{0, 1, 2, \dots\}, \lambda_{kn}^1, \lambda_{kqn}^2, \lambda_{qn}^3 \in [0, 1], s_k, T_j, N_{kr}, u_{jm}, CT1, CT2, CT3, INC_j, sl_{kq} \geq 0 \quad (24)$$

The proposed MILP model (M) for optimizing the scheduling of tests and investment of resources for new product development consists of optimizing the objective in (23) subject to constraints (1) – (22) and (24).

4.5. Remarks

The possibility of temporarily “hiring” resources can also be modeled. For a resource unit $q \in QH$ (where QH is the set of resources that can be temporarily hired, $Q = QE \cup QN \cup QD \cup QD$) we introduce a new variable b_q^* that denotes the end of the hiring period (in addition to variable b_q that denotes the beginning of the hiring period). The cost of hiring can thus be modeled either as an one-time cost discounted at b_q or $(b_q + b_q^*)/2$ (similar to the cost for permanently installing a resource $q \in QN$), or as a variable cost proportional to the length of the temporary hiring period: $[(b_q^* - b_q) - \varepsilon] \overline{cb}_q$, where \overline{cb}_q is the cost per time unit of hiring resource unit $q \in QH$ and ε is a linear approximation of the correction for the discounting.

Thus, if the temporary hiring of resources is allowed we should:

- (1) Express constraints (6) and (10) $\forall q \in (QN \cup QH)$, instead of $\forall q \in QN$.
- (2) Express constraints (9) and (4*) $\forall q \in (QE \cup QN \cup QH)$, instead of $\forall q \in (QE \cup QN)$.
- (3) Add constraint (6*) that enforces that no task can be processed by $q \in QH$ after the end of the hiring period:

$$s_k + d_k \leq b_q^* + U(1 - x_{kq}) \quad \forall q \in QH, \forall k \in K(q) \quad (6^*)$$

(4A) If temporary hiring incurs a constant cost we treat resources in QH similarly to resources in QN and express constraints (13) and (17) $\forall q \in QH$ and in constraint (20) sum over $q \in (QN \cup QH)$ instead of $q \in QN$

(4B) If temporary hiring incurs a cost proportional to the duration of hiring, we modify constraint (20) as follows:

$$CT3 = \sum_{q \in QN} cb_q \sum_n \lambda_{qn}^3 e^{a_n} + \sum_{q \in QH} \overline{cb}_q [(b_q^* - b_q) - \varepsilon] \quad (20)$$

where ε is the correction factor: if α is the annual discount rate, H is the time horizon and the time is expressed in years, the correction factor is equal to $(b_q + b_q^*)/2 \alpha^H/H$.

5. Example

An example with two new products P1 and P2, ten tests, 1 to 10, and two resource categories, A (scientists) and B (experimental equipment), is presented. Each test requires both scientists and equipment. We assume that one unit of each resource category is currently available (scientific group A1 and equipment units B1). The company can hire more scientists or buy equipment, and thus we assume that one additional unit for each category can be acquired: group A2 of type A at a cost of \$200,000 and unit B2 of type B at a cost of \$300,000. Outsourcing is available and the dummy resource units A3 and B3 are used to represent outsourcing. In this example we assume that the duration of all tests cannot be reduced by allocating more resources. Thus, the duration of all tests is fixed ($d_k^{MAX} = D_k^F$) and parameters δ_{kr} are equal to zero, and the level of resources that can be allocated to a test is also fixed ($N_{rk}^{MIN} = N_{rk}^{MAX} = 1 \ \forall k, \ \forall r$). Testing data are given in Table 3, and income data in Table 4. As shown in Table 3, the resource-utilization cost of the dummy out-sourcing units A3 and B3 is significantly higher. The duration D_k^F is in months, the costs cf_k and cq_{kq} are in $\$10^4$ and the decrease f_{jm} in the income is in $\$10^4$ /month.

Table 3: Testing data

Product	Test	D_k^F	p_k	cf_k	cq_{kq}						Precedences
					A1	A2	A3	B1	B2	B3	
P1	1	12	1	10	5	5	20	6	5	20	
	2	13	0.7	15	4	5	16	5	5	22	1
	3	12	0.95	20	6	6	18	3	3	15	2
	4	20	1	40	10	8	30	2	2	12	
	5	18	1	60	3	3	15	5	5	30	
	6	15	0.6	20	8	8	22	10	10	35	3, 5
P2	7	8	1	20	5	5	22	2	2	16	
	8	15	0.8	30	6	9	24	6	5	20	7
	9	21	1	60	5	2	18	4	4	18	
	10	17	0.7	40	4	4	12	6	6	24	8, 9

Table 4: Income data

m	g_{jm}	f_{jm}
1	0	0
2	24	8
3	48	5

$r = 0.0075$
 $INC^{MAX} = \$5,000,000$

The example is first solved assuming that additional resources cannot be acquired during the course of testing (case 1). The optimal solution yields an NPV \$1,465,000 and the Gantt chart of resources is shown in Figure 2a. The example is next solved allowing installation of new resources (case 2). The optimal solution yields an NPV of \$1,750,700 and involves the hiring of new scientists and the installation of new experimental equipment. The

Gantt chart is shown in Figure 2b and shows that some of the tests that were previously outsourced (e.g. P1-1, P1-2 and P1-4) are now performed in-house. Thus, although the duration of the testing and the income from sales is the same in the two cases for both products, a better solution is obtained in the second case because the cost of testing is lower. Specifically, the resource utilization cost CT2 in case 2 is \$1,486,900 compared to \$2,272,700 of case 1. This decrease of \$785,800 is due to the installation of the additional resource units (at a cost of \$500,000) that reduces the cost of outsourcing. The completion time, the income from sales, the testing costs and the objective value for the three cases are reported in Table 5.

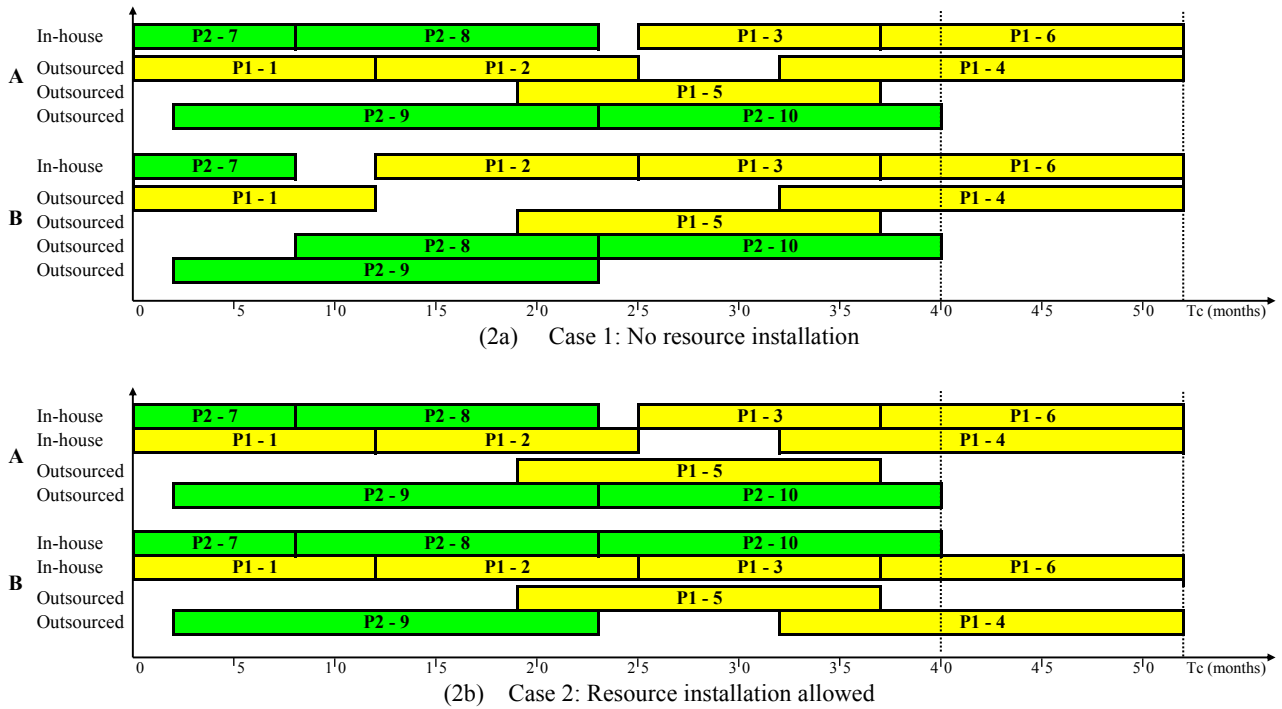


Figure 2: Resource Gantt charts of Example

Table 5: Completion time, income and testing costs of Example 1.

	Case 1		Case 2	
	P1	P2	P1	P2
Product				
Completion Time	52	40	52	40
Income (\$10 ³)	2,560	3,720	2,560	3,720
CT1 (\$10 ³)	1,196.8	1,345.6	1,196.8	1,345.6
CT2 (\$10 ³)	1,358.6	914.1	826.8	660.1
CT3 (\$10 ³)	-		500	
NPV (\$10 ³)	1,464.9		1,750.7	

The MILP model of this example consists of 979 constraints, 150 discrete and 864 continuous variables. Case 1 was solved in 2.2 CPU seconds exploring 227 nodes, while case 2 was solved in 143.0 CPU seconds exploring 22,354 modes on a PIII PC at 900 MHz using GAMS 20.7 / CPLEX 7.5.

6. Computational Improvements

As explained earlier in the paper, due to the uncertainty in many of the problem parameters, model (M) may have to be solved as a stochastic optimization problem, or it may be just a subproblem of a more general model that accounts for detailed capacity planning and/or real options. Therefore, it is crucial to be able to solve model (M) with reasonable computational effort.

Agrochemical products, for example, have to undergo a large number of testing tasks that are usually short and have small variability, i.e. they require specific type and amount of resources and their cost and duration are fixed and well known. While the technological precedence graph is usually dense, there are precedences that are undetermined and therefore subject to optimization. Thus, there are many free sequencing and resource assignment binaries ($y_{kk'}$ and x_{kq} respectively), and since the expected cost of testing depends heavily on these binaries, the difficulty in solving these problems is due to the combinatorics of the problem.

Pharmaceutical products, on the other hand, have to undergo few but very long testing tasks (Phase I, II, III clinical trials). Moreover, the precedences between clinical trials are fixed and the only unspecified precedences are those between clinical tests and investment or marketing tasks. The combinatorics of the problem, therefore, is not a complicating factor. Due to the length of the scheduling horizon, however, the planning for resources and the decisions regarding the allocation of resources are very important. As will be shown this leads to very loosely constrained problems with large linear programming (LP) relaxation gaps. To enhance the solution of the proposed model we have to both reduce the combinatorics and tighten the LP relaxation. This is accomplished by the following methods.

6.1. Fixing of Additional Precedences in the Technological Precedence Graph

Consider the example where a potential product X has to undergo the five tests of Figure 3, where the duration of each test is 5 months. The probability of success of tests 1, 4 and 5 is 1, while the probability of success of tests 2 and 3 is 0.75 and 0.6, respectively.

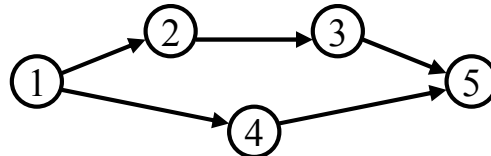


Figure 3: Graph of technological precedences.

Binaries y_{12} , y_{23} , y_{35} , y_{14} and y_{45} are fixed through constraint (2) to be equal to 1, binaries y_{21} , y_{32} , y_{53} , y_{41} and y_{54} are fixed through (2) to be equal to 0 and the remaining sequencing binaries are free. Constraint (4) for pair (3,1), thus, reads:

$$s_3 + 5 \leq s_1 + 20(1 - y_{31}) \quad (25)$$

where we have assumed that the big-M parameter U is equal to 20. Constraint (25) is satisfied by $s_3=10$, $s_1=0$ and $y_{31}=0.25$. In fact, this solution will be the optimal solution of the LP relaxation due to the fact that the objective function pushes binary y_{31} to be as large as possible in order to lower the expected cost of test 1 (because of the low probability of success of test 3). Note also that constraint (4) for pair (1,3) is satisfied by $s_1=0$, $s_3=10$ and any value of y_{13} .

As shown, some of the free sequencing binaries that we intuitively expect to take integer values due to the existing technological precedences (i.e. $y_{13}=1$ and $y_{31}=0$) obtain fractional values in the optimal LP solution. The reason for this is that, unlike other scheduling problems, the probabilities of success enter the objective function and push sequencing variables $y_{kk'}$ to take large values if the probability of success of test k is low. Fortunately, this can be avoided by preprocessing and, specifically, by fixing all the free sequencing variables that are implied by the given technological precedences, i.e. the precedence $k \rightarrow k''$ (k must finish before k'' starts) is implied when there is no technological precedence for pair (k, k'') , but there are technological precedences between pairs (k, k') and (k', k'') , specifically $k \rightarrow k'$ and $k' \rightarrow k''$.

To derive all implied precedences we construct the acyclic activity-on-node directed graph, D , of the originally given technological precedences. Note that the nodes of an acyclic digraph can always be topologically labeled, i.e. the heads of the arcs are numbered higher than the tails of the arcs (Ahuja et al., 1993). Let $n(v)$ be the label of node v , and assume that a topological labeling is available for digraph D . Then, for any implied precedence $k \rightarrow k''$ we will have $n(k) < n(k'')$, because the existing technological precedences $k \rightarrow k'$ and $k' \rightarrow k''$ (i.e arcs (k, k') and (k', k'')) imply $n(k) < n(k')$ and $n(k') < n(k'')$. Thus, if a topological labeling is available, we need only to examine ordered triplets (k, k', k'') with $n(k) < n(k') < n(k'')$ and the preprocessing algorithm derives all implied precedences in one iteration. If A is the set of technological precedences (i.e. $k \rightarrow k'' \Leftrightarrow (k, k'') \in A$), and $|K_j|$ is the cardinality of the set of tests of product j , the preprocessing algorithm (FIS) for the fixing of implied sequences is as follows:

For all products $j \in J$,
 For $k \in K(j) | n(k) = 1 \dots |K_j| - 2$
 For $k' \in K(j) | n(k') = n(k) + 1 \dots |K_j| - 1$
 For $k'' \in K(j) | n(k'') = n(k') + 1 \dots |K_j|$
 If $((k, k') \in A \text{ AND } (k', k'') \in A)$, then $(k, k'') \in A$

For the precedence graph of Figure 3, the FIS procedure identifies three additional precedences and the new precedence graph is the one shown in the right side of Figure 4.

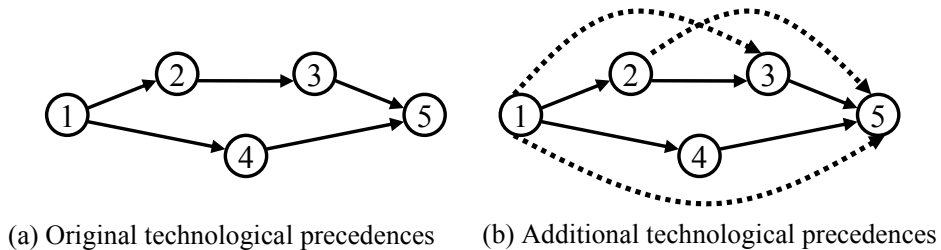


Figure 4: Application of FIS procedure on Example of Figure 4.

6.2. Tightening of Time Horizon

Constraints (4), (5) and (4*) of the proposed model are big-M constraints that yield poor LP relaxations. The original big-M factor U is an upper bound on the completion time of all products. It is usually calculated by solving the LP relaxation of the problem, choosing the greatest completion time among all products and increasing this value by a safety factor to ensure that no feasible solutions are cut-off. Since the various candidate products are in different stages of the R&D pipeline, their completion time may vary significantly, and thus we can calculate a tighter, product-specific upper bound U_j by adding a safety factor SF to the value of T_j of the LP solution.

$$U_j = T_j^{LP} + SF$$

where T_j^{LP} is the completion time of product j at the optimal LP relaxed solution. Moreover, the big-M parameters of constraints (4), (5) and (4*) can be further reduced by calculating the Earliest Start Time (EST), Latest Start Time (LST) and Latest Finish Time (LFT) of all tests and using these values to calculate tighter big-M parameters.

In the general case with variable test duration, the EST, LST and LFT of the tasks depend on the graph of the technological precedences and the minimum duration d_k^{MIN} of tests, which is given by,

$$d_k^{MIN} = d_k^{MAX} - \sum_r \delta_{rk} N_{rk}^{MAX} \quad \forall k$$

If a topological labeling of the nodes of the graph of the technological precedences is available, we can calculate the EST of tests in an increasing order, and thus calculate the EST of all tests in one iteration. Similarly, we can calculate the LFT of all tests in decreasing order in one iteration, and given the LFT we can easily calculate the LST. The procedure for the calculation of EST, LST and LFT of tasks is the following:

Time Windows Calculations (TWC)

For all $j \in J$
 For all $k \in K(j)$ [in increasing order]
 $EST_k = \max \{0, \max_{k' | (k',k) \in A} \{EST_{k'} + d_{k'}^{MIN}\}\}$
 For all $j \in J$
 For all $k \in K(j)$ [in decreasing order]
 $LFT_k = \min \{U_j, \min_{k' | (k,k') \in A} \{LFT_{k'} - d_{k'}^{MIN}\}\}$
 For all k
 $LST_k = LFT_k - d_k^{MIN}$

Thus, we can now replace parameter U in constraints (4) and (4*) by the difference $(LFT_k - EST_k)$ which is always smaller:

$$s_k + d_k \leq s_{k'} + (LFT_k - EST_{k'})(1 - y_{kk'}) \quad \forall k, \forall k' \in KK(k) \mid (k, k') \notin A \quad (4)$$

$$s_k + d_k \leq s_{k'} + (LFT_k - EST_{k'})(1 - y_{kk'}) \quad \forall q \in (QE \cup QN), \forall k \in K(q), \forall k' \in K(q) \setminus KK(k) \quad (4^*)$$

We can also calculate the Latest Installation Time (LIT) of a resource unit $q \in QN$ as the maximum LST among the tests that can be assigned in resource unit q :

$$LIT_q = \max_k \{LST_k | k \in K(q)\} \quad \forall q \in QN$$

Therefore, constraint (5) can be tightened as follows:

$$s_k \geq b_q - (LIT_q - EST_k)(1 - x_{kq}) \quad \forall q \in QN, \forall k \in K(q) \quad (5)$$

Moreover, although satisfied in every integer solution, the addition of constraint (26) tightens the LP relaxation:

$$b_q \leq LIT_q w_q \quad \forall q \in QN \quad (26)$$

To illustrate the effect of the tightening of constraint (4), consider the pair of tests 3 and 4 of the example of Figure 3. When the original upper bound $U=20$ is used, constraint (4) reads:

$$s_3 + 5 \leq s_4 + 20(1 - y_{34})$$

In the LP relaxation we get a solution where $s_3=10$, $s_4=10$ and $y_{34}=0.75$, because the objective pushes variable y_{34} to be as large as possible in order to reduce the expected cost of test 4 (due to the small probability of success of test 3). Note that binary y_{34} cannot be fixed to 0 by the FIS procedure. Assuming $U=20$ we can easily determine $EST_4 = 5$ and $LFT_3 = 15$ which means that constraint (4) can be written as:

$$s_3 + 5 \leq s_4 + (15-5)(1 - y_{34})$$

which yields a solution $s_3=10$, $s_4=10$ and $y_{34}=0.5$ which is tighter than the previously found solution ($s_3=10$, $s_4=10$ and $y_{34}=0.75$).

6.3. Fixing of Sequencing Binaries based on EST, LST and LFT

In the previous example we saw that by reducing the big-M parameters we tighten the LP relaxation but we still obtain an LP solution with a fractional value for binary y_{34} ($y_{34}=0.5$). Based on the time-windows calculations and using logic inference, however, we can fix binary y_{34} to 0. As seen in the previous section, $EST_3=10$ and $LFT_4=15$, which means that task 4 cannot be scheduled after task 3 because the sum of the durations of the two tasks (i.e. $d_3+d_4=10$) is larger than the time window (i.e. $LFT_4-EST_3=5$) within which the two tasks should take place. Similarly, since the LFT of test 1 is smaller than the EST of test 3 we can conclude that test 3 will always be scheduled after test 1, i.e. $y_{13}=1$. Hence, for all pairs of tests we can perform the two checks of Figure 5, which are similar to “edge finding” techniques of Constraint Programming (Hentenryck, 1989; Hooker, 2000):

$$\text{If } (d_k + d_{k'}) > (LFT_{k'} - EST_k) \text{ then } y_{kk'} = 0$$

$$\text{If } LFT_k \leq EST_{k'} \text{ then } y_{kk'} = 1 \text{ AND } y_{k'k} = 0$$

The second inference is very useful for tests of different products that can be scheduled in the same resource unit, because the FIP procedure cannot fix any of those sequencing binaries. Since the time required for this check is very short, we perform the two tests for all pairs of tests:

Fixing of Binaries from Time Windows (FB-TW)

For all $k \in K$,
 For $k' \neq k$
 If $(d_k^{\text{MIN}} + d_{k'}^{\text{MIN}}) > (LFT_{k'} - EST_k)$ then $y_{kk'} = 0$
 If $LFT_k \leq EST_{k'}$ then $(k, k') \in A$

Note that the tightening of big-M parameters based on the (TWC) procedure and the fixing of sequencing binaries with the (FB-TW) procedure are complementary. If $U = 20$, for example, the tightening of the big-M parameters reduces binary y_{34} to 0.5, while (FB-TW) procedure fixes binary y_{34} to 0. If the upper bound U is equal to 25, however, the tightening of big-M parameters still reduces the maximum value of binary y_{34} to 0.66 (from 0.8), while (FB-TW) does nothing.

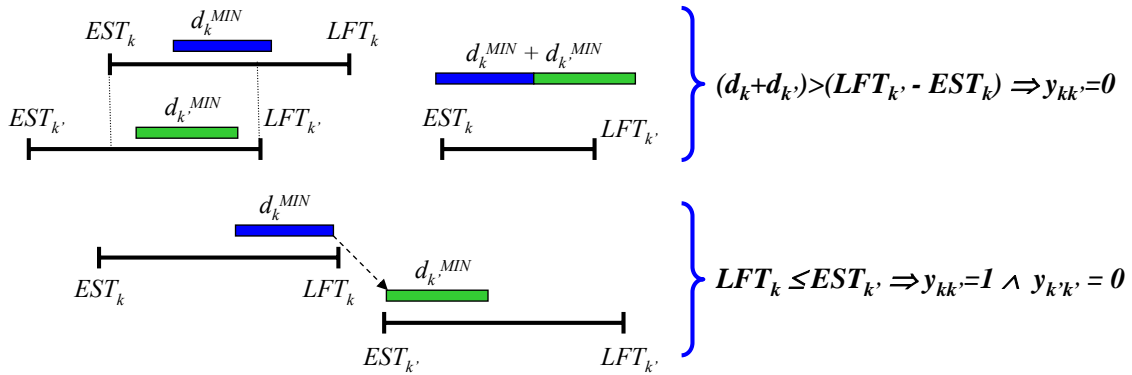


Figure 5: Logic implication based on EST and LFT of tests.

6.4. Reformulation of Constraints (12), (15) and (16)

Since the RHS of (15) is always non zero and we want to discount the resource-utilization cost cq_{kq} only when unit q is assigned to test k , we need to have two terms in the LHS of (15). In the current formulation the two terms are the summation of $\lambda_{kqn}^2 \alpha_{kn}$ (which is non-zero when $x_{kq} = 1$) and the slack variable sl_{kq} (which is non-zero when $x_{kq} = 0$). The formulation of constraints (15) and (16) and the fact that assignment variables x_{kq} have fractional solutions close to 0.5 in the optimal LP solution leads to large LP relaxation gap. The reason is that the summation in the LHS of constraint (12) is less than 1 and the slack variable sl_{kq} can take non-zero values [constraint (16)], which means that the discounting factor of the RHS of constraint (15) is equal to the sum of the slack variable sl_{kq} and a linear combination of the lowest grid points of the linearization scheme. An alternative way of formulating the LHS of constraint (15) is to define a new set of linear weights that are activated when $x_{kq} = 0$. Thus, constraint (15) is re-written as:

$$\sum_n \lambda_{kqn}^2 \alpha_n + \sum_n \bar{\lambda}_{kqn}^2 \alpha_n = -r \cdot s_k + \sum_{k' \in KK(k)} \ln(p_k) y_{k'k} \quad \forall q, \forall k \in K(q) \quad (27)$$

where:

$$\sum_n \lambda_{kqn}^2 = x_{kq} \quad \forall q, \forall k \in K(q) \quad (12)$$

$$\sum_n \bar{\lambda}_{kqn}^2 = 1 - x_{kq} \quad \forall q, \forall k \in K(q) \quad (28)$$

Since the RHS of (26) is the same as the RHS of (14), we can add the following $|N|$ constraints that give much tighter LP relaxations:

$$\lambda_{kn}^1 = \lambda_{kqn}^2 + \bar{\lambda}_{kqn}^2 \quad \forall q, \forall k \in K(q), \forall n \quad (29)$$

Any LP feasible solution of (28) and (29) is also feasible for (16); we only need to calculate sl_{kq} from $sl_{kq} = \sum_n \bar{\lambda}_{kqn}^2 \alpha_n$ and replace it into (16). The opposite, however is not true because constraint (29) is in general not satisfied by the LP solutions of (M). Replacing equations (14) and (15) with equations (27)-(29) we get a tighter and much faster model. Note also that variables w_q and N_{rk} automatically take integer values in a feasible solution:

$$y_{kk'}, x_{kq} \in \{0, 1\}, w_q, \lambda_{kn}^1, \lambda_{kqn}^2, \bar{\lambda}_{kqn}^2, \lambda_{qn}^3 \in [0, 1], s_k, T_j, N_{kr}, u_{jm}, N_{rk}, CT1, CT2, CT3, INC_j \geq 0 \quad (30)$$

The reformulated MILP model (M*) consists of equations (1) – (14), (17) – (24) and (26) – (30). The tightness of the proposed reformulation is illustrated in Appendix A through an example.

6.5. Fixing of Linearization Variables

As shown in Appendix A, because the discounted cost of a test is lower when non-adjacent approximation points are used (and preferably the two extreme ones), the discounting factor of a test in an LP solution is often approximated by two non-adjacent points. While the reformulation of equations (27) – (29) tightens the LP relaxation, it does not completely eliminate the non-adjacency phenomenon (see Appendix A). In order to further tighten the LP relaxation we can fix or restrict some of the linearization weights. To do so, we first calculate the Greatest Discounting Factor (GDF) and the Lowest Discounting Factor (LDF) of every test, i.e. the greatest and lowest values of the RHS of equations (14) and (26):

$$GDF_k = -rEST_k + \sum_{k'|(k',k) \in A} \ln(p_{k'})$$

$$LDF_k = -rLST_k + \sum_{k' \in P(k)} \ln(p_{k'})$$

where $P(k)$ is the set of tests that can be scheduled before test k :

$$P(k) = \{k' \in KK(k) \mid (k, k') \notin A \text{ AND } (d_k^{MIN} + d_{k'}^{MIN}) < (EST_k + LFT_k)\}$$

Let $v(k) \in N$ be the smallest grid point that is greater than the GDF of test k . This implies that GDF_k can be expressed as a linear combination of α_v and α_{v+1} :

$$GDF_k = \lambda_{kv(k)}^1 \alpha_{v(k)} + \lambda_{kv(k)+1}^1 \alpha_{v(k)+1} = \left(1 - \frac{\alpha_{v(k)} - GDF_k}{\alpha_{v(k)} - \alpha_{v(k)+1}}\right) \alpha_{v(k)} + \left(\frac{\alpha_{v(k)} - GDF_k}{\alpha_{v(k)} - \alpha_{v(k)+1}}\right) \alpha_{v(k)+1}$$

Since the discounting factor of test k will always be smaller than GDF_k , the linearization weights for all $n < v(k)$ will be zero, and furthermore, the maximum value of the weight of grid point $v(k)$ will be equal to the one above.

$$\lambda_{kn}^1 = 0 \quad \forall k, \forall n < v(k) \quad (31.1)$$

$$\lambda_{kv(k)}^1 \leq \left(1 - \frac{\alpha_{v(k)} - GDF_k}{\alpha_{v(k)} - \alpha_{v(k)+1}} \right) \quad \forall k \quad (31.2)$$

Similarly, if $v'(k) \in N$ is the largest grid point that is smaller than the LDF_k , LDF_k can be expressed as a linear combination of $\alpha_{v'(k)-1}$ and $\alpha_{v'(k)}$:

$$LDF_k = \lambda_{kv'(k)-1}^1 \alpha_{v'(k)-1} + \lambda_{kv'(k)}^1 \alpha_{v'(k)} = \left(1 - \frac{\alpha_{v'(k)-1} - LDF_k}{\alpha_{v'(k)-1} - \alpha_{v'(k)}} \right) \alpha_{v'(k)-1} + \left(\frac{\alpha_{v'(k)-1} - LDF_k}{\alpha_{v'(k)-1} - \alpha_{v'(k)}} \right) \alpha_{v'(k)}$$

Since the discounting factor of test k will always be greater than LDF_k , the linearization weights for all $n > v'(k)$ will be zero and the maximum value of the weight of point $v'(k)$ will be given by the weight of $\alpha_{v'(k)}$ above.

$$\lambda_{kv'(k)}^1 \leq \left(\frac{\alpha_{v'(k)-1} - LDF_k}{\alpha_{v'(k)-1} - \alpha_{v'(k)}} \right) \quad \forall k \quad (31.3)$$

$$\lambda_{kn}^1 = 0 \quad \forall k, \forall n > v'(k) \quad (31.4)$$

Since linearization weights λ_{kqn}^2 and $\bar{\lambda}_{kqn}^2$ are linked to linearization weights λ_{kn}^1 through constraint (29), constraints (31.1) – (31.4) imply that similar constraints hold for the sum $(\lambda_{kqn}^2 + \bar{\lambda}_{kqn}^2)$. A graphic representation of constraints (31.1) – (31.4) is shown in Figure 6. The effect of the addition of constraints (31.1) – (31.4) is illustrated in Appendix B through an example.

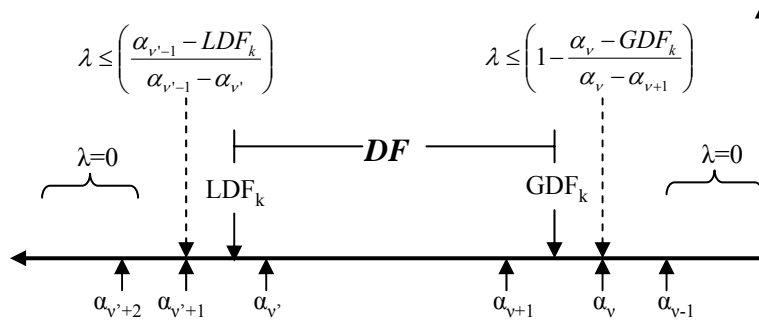


Figure 6: Constraints on linearization factors.

6.6. Logic Cuts

To enhance the solution of the MILP, we consider two major types of logic cuts (Hooker et al., 1994). These cuts are redundant in the sense that they are not necessary for the formulation and they do not cut-off any integer feasible solution. However, they serve to tighten the LP relaxation and reduce the size of the branch and bound tree. First we consider the cuts proposed by Schmidt and Grossmann (1996) that forbid cycles of two ($k \rightarrow k' \rightarrow k$) and three tests ($k \rightarrow k' \rightarrow k'' \rightarrow k$).

$$y_{kk'} + y_{k'k} \leq 1 \quad \forall k, \forall k' \in KK(k) | k < k' \quad (32.1)$$

$$y_{kk'} + y_{k'k''} + y_{k''k} \leq 2 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (32.2)$$

$$y_{k'k} + y_{kk''} + y_{k''k'} \leq 2 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (32.3)$$

Schmidt and Grossmann proposed a model in which all these cuts are added a priori, but the number of these cuts in large problem becomes prohibitively large (e.g. for a problem with two products, 90 total tests and ten resource units the number of cuts is 78,765, whereas the original equations in the model are only 10,156). Since only few of these cuts are violated, we can use cuts in (32.1) – (32.3) in a branch and cut scheme (Johnson et al., 2000), where in each node we add only the cuts that are violated.

A second class of logic cuts enforces sequences that cannot be fixed by procedure FIS. Specifically, if there are no technological precedences or sequences fixed by procedures (FIS) and (TW-FS) among tests k, k', k'' , binaries $y_{kk'}, y_{k'k''}, y_{kk''}, y_{k'k}, y_{k''k'}$, and $y_{k''k}$ will be free. However, when binaries $y_{kk'}$ and $y_{k'k''}$, for example, are fixed to 1 at some node of the branch and bound tree, binary $y_{kk''}$ should also be equal to 1. In the existing formulation, however, there are no constraints preventing binary $y_{kk''}$ from having values smaller than 1. Nevertheless, we can add a new class of cuts that force binary $y_{kk''}$ to be equal to 1 whenever $y_{kk'}$ and $y_{k'k''}$ are equal to 1 at some node of the branch and bound tree:

$$y_{kk''} \geq y_{kk'} + y_{k'k''} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.1)$$

For any triplet of tests we should also add the following cuts:

$$y_{k''k} \geq y_{k''k'} + y_{k'k} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.2)$$

$$y_{kk'} \geq y_{kk''} + y_{k''k'} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.3)$$

$$y_{k'k} \geq y_{k'k''} + y_{k''k} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.4)$$

$$y_{k'k''} \geq y_{k'k} + y_{kk''} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.5)$$

$$y_{k''k} \geq y_{k''k'} + y_{kk'} - 1 \quad \forall k, \forall k', k'' \in KK(k) | k < k' < k'' \quad (33.6)$$

The number of these cuts is very large ($O(n^3)$), but since only a few of them are violated we can again use them in a branch and cut scheme where we add only the cuts that are violated. Alternatively, at each node of the branch-and-bound tree we can perform symbolic fixing of the variables whose values are implied by the cuts in (33.1) – (33.6).

6.7. Complete Preprocessing

In summary, the complete preprocessing procedure (PRE_PR) that is applied in the improved model (M*) and includes all the methods described in the previous sections is as follows:

1. Apply FIS procedure: Fix sequencing binaries
2. Solve (M*) as LP: Calculate U_j . Repeat until there is no improvement in the objective.
3. Apply TWC procedure: Calculate big-M parameters
4. Apply FB-TW procedure: Fix additional sequencing binaries
5. Repeat 1-4 with the new information
6. Calculate GDF and LDF for all tests: Add constraints (31.1) – (31.4)
7. Solve (M*) as LP: Add violated logic cuts in (32) and (33); repeat until no improvement in LP relaxation

7. Decomposition Heuristic

The improvements described in the previous paragraph reduce the computational requirements by a factor of 3 or 4, but this is not enough for some large real-world problems. Thus, in this section we present a heuristic decomposition algorithm that yields considerable computational improvements over the full space method. The main idea is to solve one scheduling problem for each product, and then use the individual schedules to solve a reduced MILP problem for the simultaneous testing of all products. Consider the example shown in Figure 7, where products 1 and 2 are tested. When tested separately, the completion times for products 1 and 2 is 17 and 9 months, respectively. When tested simultaneously the optimal schedules are longer due to the resource constraints; the completion times for products 1 and 2 are 22 and 10 months, respectively. In terms of the graph of the sequences, this means that the graph that depicts the solution of a subproblem is “stretched” to accommodate more tests scheduled in the same resources. The “stretching” is done by the inclusion of some new sequences (thick lines of Figure 7). In the individual schedule of product 1, for instance, there is no precedence between tests 4 and 5. In the simultaneous schedule, however, test 5 is delayed (because resource 1 is used for tests 15 and 11 of product 2 as well) and thus there is a new precedence between tests 4 and 5. Note, also, that all the sequences defined in the subproblems are preserved in the simultaneous solution.

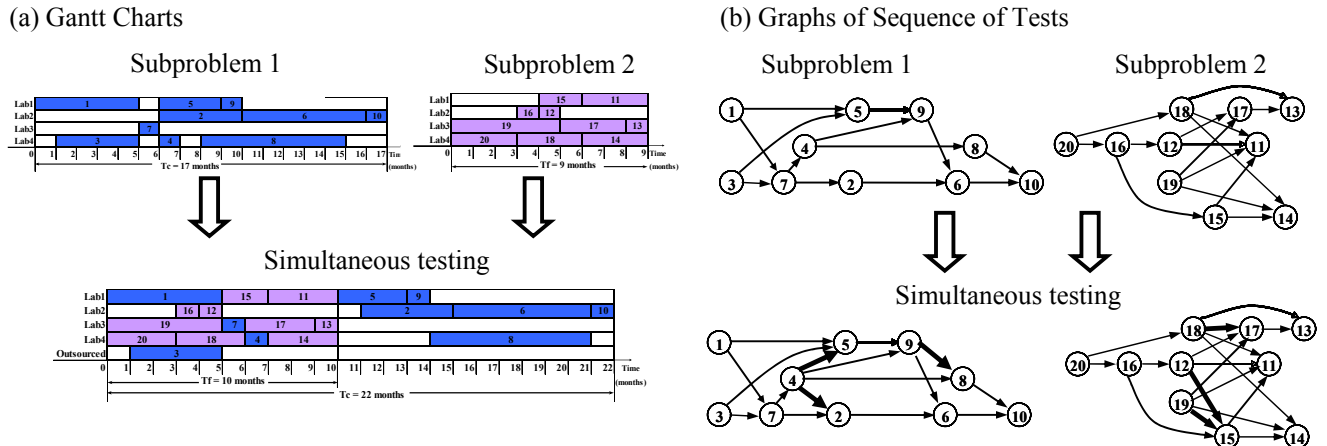


Figure 7: Decomposition Heuristic

In general, we expect that in the simultaneous testing the optimal schedules of the subproblems will be “stretched” to accommodate tests of other products in the same resource units. In other words, we expect that any precedence present in the optimal solution of a subproblem will be preserved in the optimal solution of the entire problem, and, in addition, some new precedences will be determined. Thus, in the proposed heuristic we first solve all subproblems separately, we fix the sequencing variables $y_{kk'}$ for which $y_{kk'} = 1$ in the optimal solution of a subproblem, we leave free the remaining binaries, and then we solve the simultaneous testing problem with some fixed precedences. If (M^j) is the MILP model for the scheduling of tests of product j , the proposed decomposition scheme is as follows:

1. Apply Preprocessing algorithm (PRE_PR).
2. For each product $j \in J$, solve the corresponding subproblem (M^j).
3. Add $(k, k') \in A$ if $y_{kk} = 1$ in the optimal solution of (M^j).
4. Apply (PRE_PR) in the amended precedence graph.
5. Solve (M^*) with fixed precedences from steps 3 and 4.

8. Computational Results

In this section we solve three examples using (a) model (M), (b) model (M^*), (c) preprocessing algorithm (PRE_PR) combined with model (M^*), and (d) the proposed decomposition heuristic. The size of the problem ranges from 1 to 3 products and from 10 to 28 tests. The data for the three examples are given in Appendix C. The number of products, tests and resources are given in Table 6. In order to test our models and algorithm in the more difficult instances, we have assumed that all tests have variable resource requirements (i.e. variable costs and processing time). The computational effort for the solution of more realistic examples, where only few of the tests have variable resource requirements, costs and processing time will be often significantly lower.

Table 6: Basic features of Examples 1, 2 and 3.

Example	1	2	3
Products	1	2	3
Total Tests	10	20	28
Tests per product	P1:10	P1:10, P2: 10	P1:10, P2:10, P3:8
Resource Categories	2	6	6
Existing units per category	A:2, B:2	A:2, B:2, C:1, D:1, E:2, F:2	A:2, B:2, C:2, D:2, E:2, F:2
New units per category	A:1, B:1	A:1, B:1, C:1, D:1, E:1	A:1, B:1, C:1, D:1, E:1, F:1
Outsourcing allowed	A, B	A, B, C, D, F	A, B, C, D, E, F

In Example 2, in order to develop two new products P1 and P2 we can use 10 existing resource units and we can install 5 new units. Each new product has to undergo ten tests. The resource unit Gantt chart of the optimal solution is shown in Figure 8. As shown, the testing of product P1 lasts 86 months, while the testing of product P2 lasts 134 months. A resource unit of type D (unit D2) is installed after 61 months. The optimal net present value is \$44,650,000.

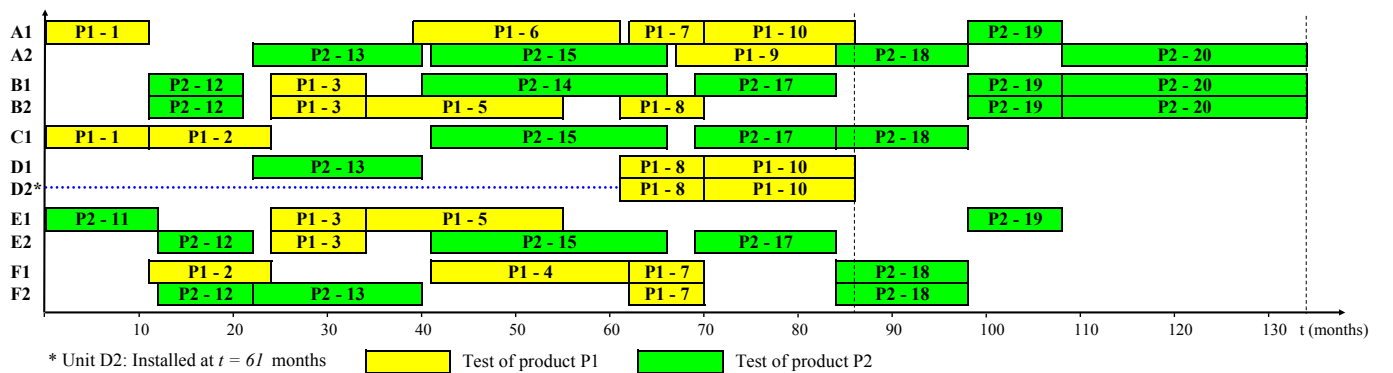


Figure 8: Resource unit Gantt chart of optimal solution of Example 2.

The model and solution statistics for examples 1, 2 and 3 are given in Tables 7, 8 and 9, respectively. For the decomposition heuristic the model statistics (number of constraints, continuous and binary variables) refer to the final MILP model and the CPU time includes the total time needed for preprocessing, the solution of the subproblems and the solution of the reduced model for the simultaneous testing. All runs were performed on a Pentium III 1000 MHz PC using GAMS 20.7 / CPLEX 7.5. The default CPLEX options and a relative optimality of 1% as termination criterion have been used for all models. For the largest problem solved, 3.3% of the total time needed by the decomposition algorithm was spent in the preprocessing stage, 1.4% in solving the individual subproblems and 95.3% in solving the final MIP.

Table 7: Model and Solution statistics of Example 1.

Model	(M)	(M*)	(M*) & Pre.
Constraints	1,368	2,088	2,116
Continuous variables	1,082	1,802	1,802
Binary variables	170	170	131
LP Relaxation (\$10 ⁶)	46.50	40.98	36.75
Objective (\$10 ⁶)	32.18	32.18	32.18
CPU seconds	47.0	14.8	3.1
Nodes	6,528	535	287

Table 8: Model and Solution statistics of Example 2.

Model	(M)	(M*)	(M*) & Pre.	Decomposition
Constraints	2,765	4,322	4,398	4,428
Continuous variables	2,328	3,885	3,885	3,885
Binary variables	507	507	310	195
LP Relaxation (\$10 ⁶)	80.41	70.56	56.13	52.74
Objective (\$10 ⁶)	36.15	44.65	44.52	44.53
CPU seconds	18,000 *	8,336.4	29.1	9.4
Nodes	1,014,083	336,873	1,520	135

* Terminated after 18,000 CPU secs.

Table 9: Model and Solution statistics of Example 3.

Model	(M)	(M*)	(M*) & Pre.	Decomposition
Constraints	5,130	7,614	7,710	7,750
Continuous variables	3,611	6,095	6,095	6,095
Binary variables	930	930	522	473
LP Relaxation (\$10 ⁶)	241.18	223.02	199.78	197.69
Objective (\$10 ⁶)	90.28 *	146.03 **	167.79	167.79
CPU seconds	18,000	18,000	5,029.4	184.0
Nodes	194,909	72,995	116,163	5,683

* Terminated after 18,000 CPU secs (best bound = 192.73)

** Terminated after 18,000 CPU secs (best bound = 192.46)

Comparing models (M) and (M*), we see that model (M*) is larger in terms of number of constraints (due to the inclusion of constraints in (28)) and number of continuous variables (due to the replacement of sl_{kq} variables by $\bar{\lambda}_{kqn}^2$ variables) but has a tighter LP formulation. Thus, in Example 1 it is faster than model (M), in Example 2 it obtains the optimal solution while (M) does not, and in Example 3 it yields a better solution with the same computational effort. As shown in Tables 7, 8 and 9, the addition of the preprocessing algorithm (PRE_PR)

reduces computational requirements significantly. The LP relaxation gap is closed and the computational effort for all examples is significantly smaller compared to model (M) and model (M*) without the preprocessing algorithm. Finally, the proposed decomposition algorithm coupled with the preprocessing algorithm is faster than the best full space model. Note that the efficiency of the decomposition algorithm becomes greater as the problem size increases.

Conclusions

A new MILP optimization model for the scheduling of testing tasks in new product development has been proposed. The proposed model allows for installation of new resources to handle more effectively the testing of new potential products by eliminating bottlenecks and reducing the outsourcing. The model also considers the amount and type of resources allocated to tests as a decision variable. The proposed model represents real-world situations more accurately than the previously reported models, and it allows the construction of more flexible schedules. In order to provide a basis for stochastic optimization approaches or for at least effectively solve multiple scenarios of real-world problems, several computational enhancements have been proposed. A graph-theoretic procedure was used to fix some of the free sequencing binaries and to compute the earliest start time (EST), latest start time (LST) and latest finish time (LFT) of tests. Based on EST, LST and LFT of tests, we calculate tight big-M parameters and fix some additional sequencing binary variables. A reformulation that yields a smaller LP relaxation gap, the addition of constraints that restrict linearization weights and logic integer cuts were also proposed. Finally, a heuristic decomposition algorithm was proposed for the solution of large problems.

Acknowledgements

The authors would like to gratefully acknowledge financial support from the National Science Foundation under Grant ACI-0121497.

Nomenclature

Sets:

J	New products \rightarrow Index j
K	Tests \rightarrow Indices k, k', k''
Q	Resource units [$Q=QE \cup QN \cup QD$] \rightarrow Index q
R	Resource categories \rightarrow Index r
N	Grid points for linearization of discounted cost \rightarrow Index n
M	Points for piece-wise income linear function \rightarrow Index m

$K(j)$ Tests of product j
 $K(q)$ Tests that can be scheduled on resource unit q
 $KK(k)$ Test that belong to the product that test k belongs
 $Q(r)$ Resource units of category r
 $R(k)$ Resource categories needed for test k

Parameters:

A $|K|*|K|$ matrix; the $(n*m)$ th element is 1 if there is a technological precedence between tests n and m
 d_k^{MAX} Parameter for the calculation of the duration of test k
 δ_{kr} Parameter for the calculation of the duration of test k
 p_k Probability of success of test k
 cf_k Fixed cost of test k
 cq_{kq} Cost of using resource unit q for test k
 cb_q Cost of installing resource unit $q \in QN$
 N_{rk}^{MIN} Minimum amount of resource units of category r needed for test k
 N_{rk}^{MAX} Maximum amount of resource units of category r needed for test k
 a_n Grid points for linear approximation of testing cost
 g_{jm} Grid points for income function
 ρ Discounting exponent
 U Upper bound on the completion time of testing
 EST_k Earliest start time of test k
 LST_k Latest start time of test k
 LFT_k Latest finish time of test k
 LIT_q Latest installation time of resource unit q

Binary Variables:

$y_{kk'}$ 1 if test k must finish before k' starts
 x_{kq} 1 if resource unit q is assigned to test k

Continuous Variables:

N_{rk} Number of resource units of category r assigned to test k
 T_j Completion time of testing of potential product $j \in J$
 s_k Starting time of test k
 d_k Duration of test k
 b_q Installation time of resource $q \in QN$
 $w_q \in [0, 1]$: 1 if resource unit $q \in QN$ is installed
 $\lambda_{kn}^1 \in [0, 1]$: Linearization weight for fixed cost of test k
 $\lambda_{kqn}^2 \in [0, 1]$: Linearization weights for the resource-dependent cost of test k
 $\bar{\lambda}_{kqn}^2 \in [0, 1]$: Linearization weights for the resource-dependent cost of test k
 $\lambda_{qn}^3 \in [0, 1]$: Linearization weight for the cost of installation of resource unit $q \in QN$
 sl_{kq} Slack variable; it is non-zero when resource unit q is not assigned to test k

- u_{jm} Variable for the calculation of income of product j
- $CT1$ Total fixed cost of tests
- $CT2$ Total resource-dependent cost
- $CT3$ Total cost for installing resource units
- INC Total income

Appendix A: Effectiveness of Reformulation of (12), (15) and (16)

To illustrate the effectiveness of the proposed reformulation consider the example of Figure 4, where test 3 must start after the completion of test 2 whose probability of success is 0.75. Test 3 requires resource unit A or resource unit B and the cost of assigning unit A or unit B to test 3 is equal to \$1,000. Let further assume that the discounting factor is approximated by the three points of Figure A1: (0,1), (-0.5, $e^{-0.5}$) and (-1.0, $e^{-1.0}$).

If test 3 starts as soon as test 2 finishes, at $t=10$ months, and assuming a discounting exponent $r=0.0075$ (annual discounting of 9%) the real discounted, expected resource cost for test 3, regardless of which resource unit is used, is:

$$CT2 = 1,000 e^{-0.0075 \cdot 10 + \ln 0.75} = 1,000 e^{-0.36} = \mathbf{\$696} \quad (A1)$$

Using the linearization of Figure A1, the exponent -0.36 can be expressed as the following linear combination, where we have dropped the index q from the weights λ^2_{kqn} because both A and B resource units have the same cost:

$$-0.36 = 0.28(0.0) + 0.72(-0.5) + 0(-1.0) = \lambda^2_{31} a_1 + \lambda^2_{32} a_2 + \lambda^2_{33} a_3$$

and thus the approximated expected, discounted cost of test 3 in an integer solution should be:

$$CT2 = 1,000 (0.28e^0 + 0.72e^{-0.5} + 0e^{-1.0}) = 1,000 (0.28 + 0.437) = \mathbf{\$717} \quad (A2)$$

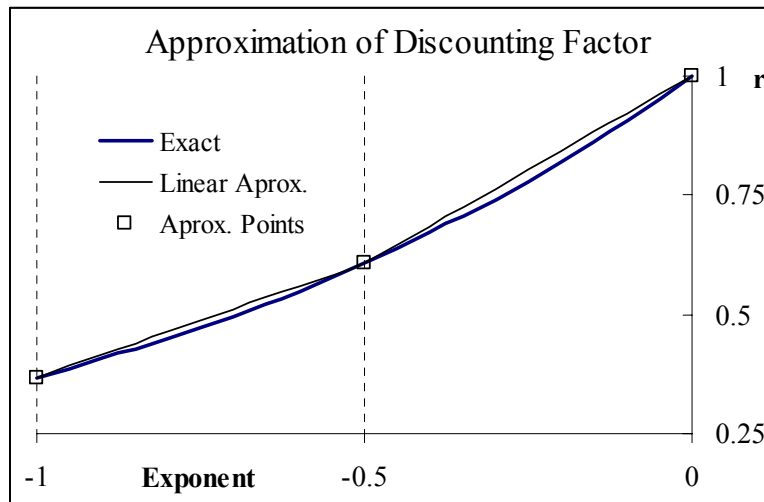


Figure A1: Linear approximation of discounting factor.

In the LP relaxation, however, we obtain a solution where binaries x_{YA} and x_{YB} are equal to 0.5, which means that the sum $\sum_n \lambda^2_{kqn}$ is also equal to 0.5. Thus, constraint (15) for test 3 and resource units A and B reads as follows:

$$\begin{aligned}\sum_n \lambda^2_{3An} \alpha_n + sl_{3A} &= -0.36 \Rightarrow \lambda^2_{3A2} \alpha_2 + \lambda^2_{3A3} \alpha_3 = -0.36 \Rightarrow 0.28(-0.5) + 0.22(-1.0) = -0.36 \\ \sum_n \lambda^2_{3Bn} \alpha_n + sl_{3B} &= -0.36 \Rightarrow \lambda^2_{3B2} \alpha_2 + \lambda^2_{3B3} \alpha_3 = -0.36 \Rightarrow 0.28(-0.5) + 0.22(-1.0) = -0.36\end{aligned}$$

Hence, the discounted cost of test 3 in an optimal LP solution using constraints (15) and (16) will be:

$$CT2 = cq_{3A} \sum_n \lambda^2_{3An} \exp(\alpha_n) + cq_{3B} \sum_n \lambda^2_{3Bn} \exp(\alpha_n) = 2 \cdot 1,000(0.28e^{-0.5} + 0.22e^{-1.0}) = \mathbf{\$501.5} \quad (A3)$$

which is considerably smaller than the actual (\$696) or approximated expected, discounted (\$717) cost.

Since the discounting exponent of test 3 in the LP optimal solution is -0.36 and the summation of λ^1_{kn} in constraint (12) is equal to 1, the summation of $\lambda^2_{kqn} + \bar{\lambda}^2_{kqn}$ in the LHS of (27) should also be equal to 1 due to constraint (29) and this implies that there are only two ways to express the discounting exponent of test 3.

Case 1: As a linear combination of α_1 and α_2

$$\begin{aligned}-0.36 &= \lambda^1_{31} \alpha_1 + \lambda^1_{32} \alpha_2 = 0.28(0.0) + 0.72(-0.5) \\ \Rightarrow \lambda^2_{3A1} &= 0, \quad \lambda^2_{3A2} = 0.5, \quad \lambda^2_{3A3} = 0, \quad \bar{\lambda}^2_{3A1} = 0.28, \quad \bar{\lambda}^2_{3A2} = 0.22, \quad \bar{\lambda}^2_{3A3} = 0 \\ \lambda^2_{3B1} &= 0, \quad \lambda^2_{3B2} = 0.5, \quad \lambda^2_{3B3} = 0, \quad \bar{\lambda}^2_{3B1} = 0.28, \quad \bar{\lambda}^2_{3B2} = 0.22, \quad \bar{\lambda}^2_{3B3} = 0 \\ \Rightarrow CT2 &= cq_{3A} \sum_n \lambda^2_{3An} \exp(\alpha_n) + cq_{3B} \sum_n \lambda^2_{3Bn} \exp(\alpha_n) = 2 \cdot 1,000(0.5e^{-0.5}) = \mathbf{\$606.5} \quad (A4)\end{aligned}$$

Case 2: As a linear combination of α_1 and α_3

$$\begin{aligned}-0.36 &= \lambda^1_{31} \alpha_1 + \lambda^1_{33} \alpha_3 = 0.64(0.0) + 0.36(-1.0) \\ \Rightarrow \lambda^2_{3A1} &= 0.14, \quad \lambda^2_{3A2} = 0, \quad \lambda^2_{3A3} = 0.36, \quad \bar{\lambda}^2_{3A1} = 0.5, \quad \bar{\lambda}^2_{3A2} = 0, \quad \bar{\lambda}^2_{3A3} = 0 \\ \lambda^2_{3B1} &= 0.14, \quad \lambda^2_{3B2} = 0, \quad \lambda^2_{3B3} = 0.36, \quad \bar{\lambda}^2_{3B1} = 0.5, \quad \bar{\lambda}^2_{3B2} = 0, \quad \bar{\lambda}^2_{3B3} = 0 \\ \Rightarrow CT2 &= cq_{3A} \sum_n \lambda^2_{3An} \exp(\alpha_n) + cq_{3B} \sum_n \lambda^2_{3Bn} \exp(\alpha_n) = 2 \cdot 1,000(0.14e^0 + 0.36e^{-1.0}) = \mathbf{\$544.9} \quad (A5)\end{aligned}$$

Thus, the discounted cost of test 3 in the LP solution using the reformulation of constraints (27)-(29) is higher than the discounted cost of the LP solution with constraints (15) and (16) (\$518) for both cases. A summary of the results is given in Table A1.

Table A1: Summary of results for $s_3=10$.

	Linear Combination	Discounted Cost
Exact (nonlinear)	-	696
Approx. in integer solution	$0.28\alpha_1 + 0.72\alpha_2$	717
Approx. in LP solution with (15) & (16)	$2(0.28\alpha_2 + 0.22\alpha_3)$	501.5
Approx. in LP solution with (27)-(29) (case 1)	$2(0.5\alpha_2)$	606.5
Approx. in LP solution with (27)-(29) (case 2)	$2(0.14\alpha_1 + 0.36\alpha_3)$	544.9

When the reformulation is used one would expect to always obtain an LP solution where the linear combination of points α_1 and α_3 (case 2) is used, because the objective is to minimize cost. This, however, does not always happen. More specifically, the linear combination of points α_1 and α_2 (case 1) may be the optimal LP solution if

the fixed cost cf_3 of test 3 is high compared to the resource-utilization cost (cq_{3A} or cq_{3B}). To illustrate, assume that the fixed cost of test 3 is \$5,000. This implies that if the discounting factor in the RHS of equation (13) is expressed as a linear combination of the two adjacent points α_1 and α_2 the discounted fixed cost will be:

$$CTI = 5,000(0.28e^0 + 0.72e^{-0.5}) = 5,000 \cdot 0.7167 = \$3,583.5 \quad (A6)$$

whereas if it is expressed as a linear combination of the two extreme points α_1 and α_3 the discounted fixed cost will be much higher:

$$CTI = 5,000(0.64e^0 + 0.36e^{-0.5}) = 5,000 \cdot 0.7724 = \$3,862 \quad (A7)$$

Thus, the total costs in the LP solution for test 3 for the two cases are:

$$\text{Case 1: } CTI + CT2 = 3,583.5 + 606.5 = \mathbf{\$4,189}$$

$$\text{Case 1: } CTI + CT2 = 3,862 + 544.9 = \mathbf{\$4,406.9}$$

Since the total testing cost for case 1 is smaller, points α_1 and α_2 will be used in both constraints (14) and (28). The optimal LP solutions for the discounted resource-utilization cost of test 3 are shown in Figure A2.

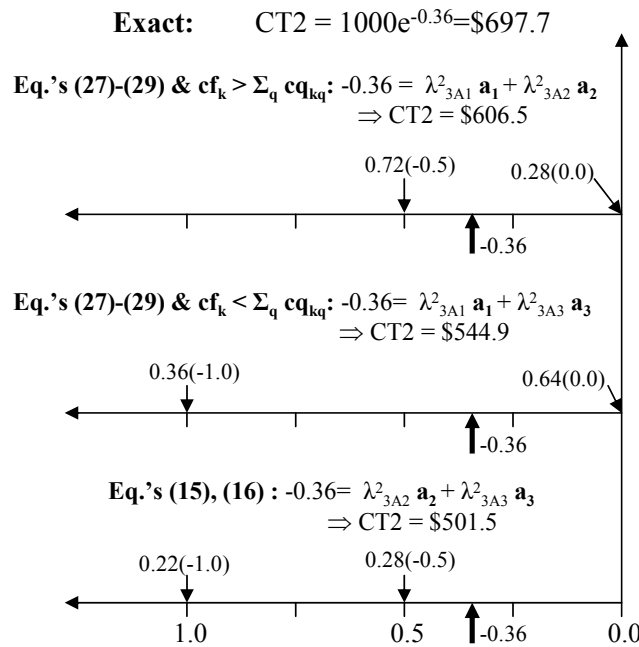


Figure A2: LP solutions for the two formulations.

Appendix B: Application of Fixing of linearization Weights

In Appendix A we showed that constraints (27)-(29) yield tighter LP solutions, especially when the fixed cost cf_k is larger than the resource-utilization cost $\sum_q cq_{kq}$ of test k . If the fixed cost of test 3 is \$1,000 instead of \$5,000, for instance, the fixed cost using points α_1 and α_2 is:

$$-0.36 = 0.28(0.0) + 0.72(-0.5) \Rightarrow CTI = 1,000(0.28e^0 + 0.72e^{-0.5}) = 1,000 \cdot 0.7167 = \$716.7$$

while using points α_1 and α_3 is:

$$-0.36 = 0.64(0.0) + 0.36(-1.0) \Rightarrow CT1 = 1,000(0.64e^0 + 0.36e^{-0.5}) = 1,000 \cdot 0.7724 = \$772.4$$

The total discounted cost for test 3 will be:

$$\text{Case 1: } CT1 + CT2 = 716.7 + 606.5 = \mathbf{\$1,323.2}$$

$$\text{Case 1: } CT1 + CT2 = 772.4 + 544.9 = \mathbf{\$1,317.3}$$

which means that in the optimal LP solution points α_1 and α_3 will be used.

This optimal LP solution, however, can be eliminated by constraints (31.1) – (31.4). Applying constraints (A) and (B) for test 3 we get:

$$GDF_3 = -rEST_3 + \sum_{k'|(k',3) \in A} \ln(p_{k'}) \Rightarrow GDF_3 = -0.0075 \cdot 10 + \ln(0.75) \Rightarrow GDF_3 = -0.36$$

$$SDF_3 = -rLST_3 + \sum_{k' \in P(3)} \ln(p_{k'}) \Rightarrow LDF_3 = -0.0075 \cdot 10 + \ln(0.75) \Rightarrow LDF_3 = -0.36$$

which means that we have $v(3)=1$ and $v'(3)=2$ and by plugging the corresponding values into constraints (31.1) – (31.4) we get:

$$\lambda_{31}^1 \leq 0.28 \quad \lambda_{32}^1 \leq 0.72 \quad \lambda_{33}^1 = 0$$

Hence, the addition of constraints (31.1) – (31.4) forces λ_{33}^1 to zero, and because of constraint (29) terms $(\lambda_{3A3}^2 + \bar{\lambda}_{3A3}^2)$ and $(\lambda_{3B3}^2 + \bar{\lambda}_{3B3}^2)$ are also forced to zero; i.e. -0.36 is expressed as a linear combination of points α_1 and α_2 .

For test 4 we can calculate $EST_4=5$ and $LST_4=10$ which give:

$$GDF_4 = -rEST_4 + \sum_{k'|(k',3) \in A} \ln(p_{k'}) \Rightarrow GDF_4 = -0.0075 \cdot 5 \Rightarrow GDF_3 = -0.038$$

$$SDF_4 = -rLST_4 + \sum_{k' \in P(3)} \ln(p_{k'}) \Rightarrow LDF_4 = -0.0075 \cdot 10 \Rightarrow LDF_3 = -0.075$$

Thus, $v(4)=1$ and $v'(4)=2$ and constraints (31.1) – (31.4) become:

$$\lambda_{41}^1 \leq 0.924 \quad \lambda_{32}^1 \leq 0.15 \quad \lambda_{33}^1 = 0$$

Appendix C: Example Data

In Example 1 only product P1 is tested; in example 2, products P1 and P2 are tested, and in Example 3, products P1, P2 and P3 are tested. The data for all tests are given in Tables C1, C2 and C3. A test k requires a resource category r if and only if the corresponding δ_{kr} entry is non-zero. For all pairs $(k,r)|\delta_{kr}>0$ we have $N_{rk}^{MIN}=1$. The cost of outsourcing is 80% higher.

Table C1: Income data for Examples 2 and 3.

m	g_m (months)	f_m ($\$10^5$ /month)	
1	0	10	$INC_{P1}^{MAX} = \$300,000,000$
2	24	10	$INC_{P2}^{MAX} = \$350,000,000$
3	48	10	$INC_{P3}^{MAX} = \$400,000,000$

Table C2: Resource classes and units for Examples 2 and 3.

Category	Existing Units		New Units / Cost ($\$10^5$)		Outsourcing	
	Example 2	Example 3	Example 2	Example 3	Example 2	Example 3
A	A1, A2	A1, A2	A3 / 80	A / 80	YES	YES
B	B1, B2	B1, B2	B3 / 60	B3 / 60	YES	YES
C	C1	C1, C2	C2 / 100	C3 / 100	YES	YES
D	D1	D1, D2	D2 / 120	D3 / 120	YES	YES
E	E1, E2	E1, E2	E3 / 90	E3 / 90	NO	YES
F	F1, F2	F1, F2		F3 / 80	YES	YES

Table C3: Testing data for Examples 2 and 3.

Test	k	d_k^{MAX}	p_k	cf_k	Precedences $k' (k',k) \in A$	$r \in R(k)$	cq_{kr}	δ_{kr}	N_{rk}^{MAX}
		(months)		($\$10^5$)			($\$10^5$)		
P1	1	16	1	20		A/C	7/8	1/2	1/2
	2	18	0.88	25	1	C/F	9/10	2/3	1/1
	3	20	0.95	30	2	B/D/F	3/5/11	2/4/1	2/1/2
	4	24	1	50	2, 3	A/F	5/4	2/1	2/2
	5	30	0.75	80		B/E	5/6	2/1	1/2
	6	24	0.9	60		A/C	7/3	0/2	2/2
	7	22	1	100		A/D/F	5/15/10	3/2/2	2/1/3
	8	20	1	50	3, 6	B/D/F	5/8/10	3/1/2	2/3/1
	9	18	0.85	20	4, 5	A/C	7/3	1/0	1/2
	10	22	1	40	7, 8	A/D	9/10	0/2	1/3
P2	11	15	0.6	40		C/E	17/13	1/2	2/1
	12	20	0.8	50	11	B/E/F	12/14/15	1/1/3	3/1/2
	13	25	1	60	12	A/D/F	18/12/19	2/1/3	1/2/1
	14	30	1	70	11, 12, 13	B/D	8/9	1/2	3/1
	15	28	0.95	80	13	A/C/E	12/11/10	1/0/2	2/3/1
	16	24	1	90	14	A/D/F	7/9/8	2/1/3	1/3/1
	17	20	1	20	14, 15	B/C/E	6/5/4	2/1/2	1/1/2
	18	22	0.98	40	16, 17	A/C/F	3/6/9	1/0/3	2/1/2
	19	16	1	10	17, 18	A/B/E	8/6/4	2/1/1	1/3/1
	20	35	1	12	18, 19	A/B/D	12/14/16	1/3/2	1/2/1
P3	21	20	0.9	25		A/B	8/6	1/2	2/2
	22	25	1	30		B/C	10/5	2/2	2/3
	23	30	1	30	22	C/D	8/10	3/2	1/3
	24	32	0.8	60	21, 23	D/E	4/7	1/2	2/3
	25	25	0.9	70	22, 23	E/F	9/6	2/2	2/1
	26	24	1	45	23, 24	A/E	6/2	3/2	2/2
	27	18	1	50	25, 26	B/C/D	8/6/5	1/2/2	1/2/3
	28	28	0.95	40	26, 27	A/E/F	4/10/8	3/2/1	2/2/1

References

- (1) Gardner, C.R.; Almarsson, O.; Chen, H.; Morissette, S.; Peterson, M.; Zhang, Z.; Wang, S.; Lemmo, A.; Gonzalez-Zugasti, J.; Monagle, J.; Marchionna, J.; Ellis, S.; McNulty, C.; Johnson, A.; Levinson, D.; Cima, M. Application of High Throughput Technologies to Drug Substance and Drug Product Development. In Proceedings of the Fourth International Conference on Foundations of Computer-Aided Process Operations (I. E. Grossmann and C. M. McDonald, editors), CACHE, **2003**, 3-11

- (2) Schmidt, C.W.; Grossmann, I.E. Optimization Models for the Scheduling of Testing Tasks in New Product Development. *Ind. Eng. Chem. Res.*, **1996**, 35, 3498-3510.
- (3) Jain, V; Grossmann, I. E. Resource-constrained Scheduling of Tests in New Product Development. *Ind. Eng. Chem. Res.*, **1999**, 38, 3013-3026.
- (4) Honkomp, S.J.; Reklaitis, G.V.; Pekny, G.F. Robust Planning and Scheduling of Process Development Projects under Stochastic Conditions. AIChE Annual Meeting, Los Angeles, CA, **1997**.
- (5) Subramanian, D.; Pekny, J.F.; Reklaitis, G.V. A Simulation-Optimization Framework for Research and Development Pipeline Management. *AIChE Journal*. **2001**, 47 (10), 2226-2242.
- (6) Subramanian, D.; Pekny, J. F.; Reklaitis, G.V.; Blau, G.E. Simulation-Optimization Framework for Stochastic Optimization of R&D Pipeline Management. *AIChE Journal*. **2003**, 49(1), 96-112.
- (7) Maravelias, C.; Grossmann, I.E. Simultaneous Planning for New Product Development and Batch Manufacturing Facilities. *Ind. Eng. Chem. Res.*, **2001**, 40, 6147-6164.
- (8) Maravelias and Grossmann, Logic Inference and a Decomposition Algorithm for the Resource-Constrained Scheduling of Testing Tasks in Development of New Pharmaceuticals and Agrochemicals. Submitted for publication (2002).
- (9) Schmidt, C. W.; Grossmann, I. E.; Blau, G. E. Optimization of Industrial Scale Scheduling Problems in New Product Development. *Computers Chem. Engng.* **1998**, 22, S1027-S1030.
- (10) Hooker, J.N.; Yan, H.; Grossmann, I.E.; Raman, R. Logic Cuts for Processing Networks with Fixed Charges, *Computers and Operations Research*, **1994**, 21 (3), 265-279.
- (11) Johnson E.L.; Nemhauser, G.L.; Savelsbergh, M.W.P.; Progress in Linear Programming Based Branch-and-Bound Algorithms: Exposition, *INFORMS Journal of Computing*, **2000** (12).
- (12) Ahuja, R.K.; Magnanti, T.L.; Orlin, J.B. Network Flows: Theory, Algorithms and Applications, Prentice Hall, **1993**.
- (13) Rogers M.J.; Gupta, A.; Maranas, C.D. Real options based analysis of optimal pharmaceutical research and development portfolios. *Ind. Eng. Chem. Res.*, **2002**, 41 (25), 6607-6620.
- (14) Papageorgiou L.G.; Rotstein G.E.; Shah, N. Strategic supply chain optimization for the pharmaceutical industries *Ind. Eng. Chem. Res.*, **2001**, 40 (1), 275-286.
- (15) Shah, N. Pharmaceutical Supply Chains: Key Issues and Strategies for Optimization. In Proceedings of the Fourth International Conference on Foundations of Computer-Aided Process Operations (I. E. Grossmann and C. M. McDonald, editors), CACHE, **2003**, 73-85.
- (16) Birge, J.R.; Louveaux, F. *Introduction to Stochastic Programming*; Springer: New York, **1997**.
- (17) Hentenryck, P.V. Constraint Satisfaction in Logic Programming. MIT Press, Cambridge, MA, **1989**.
- (18) Hooker, J. Logic Based Methods for Optimization: Combining Optimization and Constraint Satisfaction. John Willey and Sons, Inc. New York, **2000**.