IIEC 2012
The 8th International Industrial Engineering Conference
هشتمین کنفرانس بین‌المللی مهندسی صنایع
دانشگاه صنعتی امیرکبیر، تهران
۲۶ و ۲۷ بهمن ۱۳۹۰

One-by-one or altogether: The advantages of pursuing alternative innovation activities

Mohammad Ranjbar
Department of Industrial Engineering, Faculty of Engineering, Ferdowsi University of Mashhad
Mashhad, Iran
m_ranjbar@um.ac.ir

***Abstract*—A fundamental challenge associated with research or new product development projects is identifying that innovative activity that will deliver success. In such projects, it is typically the case that innovative breakthroughs can be achieved by any of several possible alternative technologies, some of which may fail due to the technological risks involved. In some cases, the project payoff is obtained as soon as any single technology is completed successfully. We refer to such a project as an *alternative-technologies* project and in this paper we consider the *alternative-technologies project scheduling problem*. We examine how to schedule alternative R&D activities in order to maximize the expected net present value, when each technology has a cost and a probability of failure. We formulate the problem, develop a branch-and-bound algorithm to obtain optimal solutions and evaluate the efficiency of the algorithm using extensive computational results.**

Keywords: Project scheduling; Technology; Research and Development; Innovation.

I. INTRODUCTION

The development of complex and innovative products is characterized by much uncertainty. In order to deal with this uncertainty, it has been suggested that research and development (R&D) projects should pursue multiple alternative solutions for developing the new products (see [1] and [2]). The selection and scheduling of these attempts, hereafter referred to as *alternatives*, is crucial for increasing the likelihood of successfully developing a product, minimizing development time and obtaining revenues as early as possible. Consider, for instance, a software development firm that has the option to develop their web services using either a traditional Java SPRING framework or the pioneering Ruby-on-Rails framework. While both might achieve a similar functionality, the traditional Java SPRING framework will take longer to develop, but is more likely to handle the expected volume of users.

In this paper, we focus on a single firm engaged in a single R&D or new product development (NPD) project. The project can be achieved by any one of several alternatives. Each alternative is characterized by a cost, a duration and a probability of technical success (PTS). The successful completion of an alternative corresponds to the completion of the project and obtaining the project payoff. A serial schedule, in which alternatives are not attempted simultaneously, is conservative in terms of costs and minimizing the downside risk, but might result in the maximum project duration. On the other extreme, simultaneously developing all the potential alternative technologies, which could lead to the minimum project duration and an earlier launch date, carries a large downside risk and higher upfront costs ([3] and [4]). Our goal is to analyze such trade-offs and to solve the underlying optimization problem, which will be referred as the *Alternative-Technologies Project Scheduling Problem* (*ATPSP*). An optimal solution boils down to the determination of the optimal timing of the alternatives in order to maximize the project's expected net present value (*eNPV*).

The remainder of this paper is organized as follows. We survey earlier related work in Section 2. The problem formulation and modeling are presented in Section 3. In each section 4, we present a branch-and-bound (B&B) algorithm. Computational results are discussed in Section 5. Finally, outlook on further research are given in Section 6.

II. RELATED WORKS

The challenges of planning and scheduling NPD activities have been the subject of a rich body of literature. Weitzman

[6] examines a serial, or sequential, execution of alternatives, each characterized by a cost, duration, and a reward probability distribution. Weitzman [6] identifies the optimal sequential strategy that maximizes the expected present value. While Weitzman [6] considers alternative activities sequentially, Nelson [7] and Abernathy and Rosenbloom [8] demonstrate the value of scheduling activities in parallel when time is a prime concern. Dahan [9] examines the trade-off between parallel and sequential scheduling in alternative prototype development. His analysis demonstrates that extremely high or low PTS reduces the variability of outcomes and should lead to fewer parallel prototypes being built. This topic has also been addressed, among others, by Eppinger et al. [10], Krishnan et al. [11] and Roemer and Ahmadi [12]. Teunter and Flapper [13] investigate issues similar to ours but in a flow production environment. While closely related settings have been studied, the ATPSP has, to our knowledge, not been considered as a separate problem and no solution approach has been developed for it. The earlier work surveyed supra has always imposed assumptions on the project's structure as consisting of a number of sequential stages, and/or considered the project's activities as being identical. De Reyck and Leus [14] consider the R&D project scheduling problem (RDPSP), in which project activities are interrelated by finish-to-start precedence relations. In their model, however, the project is successful only if all individual activities succeed. In this article, we allow for the alternatives' durations and other characteristics to differ, thus representing a more realistic challenge. Parallel as well as sequential execution of alternatives is possible, and the time value of money is taken into account by correcting the cash flows with an appropriate discount factor.

III. PROBLEM FORMULATION AND MODELING

We focus on a firm that is engaged in an NPD project and tries to maximize its project's *eNPV* by determining which alternative technologies to pursue and when. The development will end as soon as an alternative is successful, at which time the project terminates and the firm obtains the project payoff. For each alternative, we define two events: a start event and a finish event. The alternatives are executed without preemption from start to finish, and the alternative's outcome, which can be either success or failure, is revealed at the finish. Consequently, each start event will only occur if all alternatives with earlier finish events have had a negative outcome. We do not consider resource constraints and duration uncertainty and view the PTS of each of the alternatives as independent. The model's parameters are defined in Table 1.

Table 1. Parameters

Parameter	*Definition*
N	set of alternatives,
c_i	cost of alternative $i \in N$, a non-positive integer; incurred at the start of the alternative
C	end-of-project positive payoff, a positive integer
d_i	duration of alternative $i \in N$, a positive integer
p_i	PTS of alternative $i \in N$
r	continuous discount rate
A	(strict) partial order on *N*, an irreflexive and transitive relation imposing the constraints $s_i + d_i \le s_j$ for all $\forall (i,j) \in A$
δ	The project deadline

Precedence constraints imposed by A might represent repetitive testing or trials, and also allow to model fallback options: alternative plans devised by management in the event the primary option falters ([15]).

A schedule **s** is *feasible* if it respects the constraints imposed by *A*. The objective is to maximize the project's *eNPV*, and so each alternative's cost is weighted by the probability of joint failure of the already finished alternatives. We represent each schedule by a vector of start times $\mathbf{s} = (s_1, s_2, ..., s_n)$ where s_i is a non-negative integer and indicates the start time of alternative *i*. We also consider $\mathbf{f} = (f_1, f_2, ..., f_n)$ as the vector of finish times where $f_i = s_i + d_i$ indicates the finish time of alternative *i*.

The problem ATPSP can now be formulated as the following non-linear integer programming model.

$$\max \; g(\mathbf{s}) = \sum_{i=1} \left(\prod_{j \in FBA(s_i)} \right)$$

$$+C \sum_{t \in NUF} \left(\prod \prod \right)$$

$$+C \sum_{i: f_i \notin NUF} \left(\prod_{\in FB(f_i)} \right)$$

subject to

$$s_i + d_i \le \quad \forall (i,j) \in A$$

$$s_i + d_i \le \quad \forall \in N$$

$$s_i \in \mathbb{Z}^+ \quad \forall \in N$$

In the objective function, $FBA(s_i)$ indicates set of alternatives which are finished before or at start of alternative *i*. Also, $FB(f_i)$ shows set of alternatives which are finished before finish of alternative *i*. In addition, *NUF* shows set of non-unique finish times such that every $t \in NUF$ is identical to finish time of at least two alternatives. The set of alternatives which are finished at $t \in NUF$ are shown by $FA(t)$ and $FB(t)$ shows set of alternatives which are finished before time instant *t*.

Objective function includes three main terms in which *exp*(.) shows exponential function. The first term indicates the expected costs in which $\prod_{j \in FBA(s_i)} (1-p_j)$ indicates the probability of joint failure of all alternatives finished before or at the start of alternative *i*. The Summation of two other terms indicates the expected revenue. The second term displays the expected revenue for the alternatives whose finish times belong to *NUF*. In this formula, $\left(\prod \right)$ shows the probability of succeeding at least one of the alternatives finished at time instant *t*. Moreover, $\prod_{j \in FB(t)} (1-p_j)$ indicates the probability of joint failure of all alternatives finished prior to the time instant *t*. Finally, the last term indicates the expected income for the alternatives whose finish times are unique. In this formula, $\prod_{j \in FBA(f_i)} (1-p_j)$ represents the probability of joint failure of all alternatives finished prior to or at the completion of alternative *i*.

The first constraint of the model indicates the precedence relations among alternatives imposed by set *A*. The second constraint implies that all alternatives must be finished before or at the given deadline. If this constraint is not considered, some alternatives may be finished at infinite. The last constraint shows that all start times are non-negative integers (Z^+ indicates the set of non-negative integers).

The ATPSP can be shown to be NP-hard (Theorem 1 in [5]), even with unit durations and $r = 0$. Consequently, an exact algorithm with better than exponential time complexity is unlikely to exist. A number of sub-problems with $r = 0$ can be distinguished, however, that can be solved in polynomial time. One example is the case $A = \varnothing$, for which Price [16] shows that a serial schedule executing the activities in non-decreasing order of c_i / p_i is optimal. Other special cases (generalizing the result of Price) are surveyed by De Reyck and Leus [14].

IV. The Branch-and-Bound Algorithm

A. Branching strategy

In B&B, each node of the search tree corresponds to a partial schedule **s** where set $E(\mathbf{s})$ indicates the set of events of this schedule. Each event $e \in E(\mathbf{s})$ corresponds to start time or finish time of at least one alternative. We construct our B&B algorithm based upon the *property* 1 which reduces the search space noticeably.

Property 1: for each instance of the ATPSP, there is an optimal solution in which start time or finish time of each alternative is simultaneous with at least another event.

Now, in the root node, set $E(\mathbf{s})$ is initialized as $E(\mathbf{s}) = \{0, \delta\}$. For each node v which corresponds to a partial schedule **s**, we consider a set of eligible alternatives EL_v including all alternatives whose all predecessors whose all predecessors or all successors are scheduled are scheduled. For each alternative $i \in EL_v$, we consider a set of possible starting times based upon events of set *E* and precedence constraints. In other words, branches of node v are created by considering start time and finish time of all alternatives $i \in EL_v$ equal to the each event $e \in E(\mathbf{s})$ if the precedence constraints are not violated. For this purpose, the possible start times and finish times of each alternative i are calculated as $\{t \in \mathbf{s}, \ldots \leq \ldots \leq ls_i\}$ and $\{t \in \mathbf{s}, \ldots \leq \ldots \leq lf_i\}$, respectively. Since each alternative is added to a partial schedule when all of its predecessors are added before, in each node, the earliest start time and consequently the earliest finish time of each alternative may be updated while the latest start times and the latest finish times will not be updated. In each node, both earliest start time and latest start time of each alternative and consequently, the earliest finish times and the latest finish times may be updated.

B. Bounding strategy

Based upon *property* 1, for adding an alternative *i* to a partial schedule **s**, s_i or f_i should be concurrent with at least another event $e \in E(\mathbf{s})$. After adding alternative *i* to the partial schedule **s**, a new event e' may be added to *E*. If it happens, event *e* is the parent of event e' and event e' is the child of event *e*.

Dominance rule 1: consider a schedule S and event $e \in E(\mathbf{s})$. If C_e indicates set of children of e, all members of C_e should be added to it in increasing order of index number of their corresponding alternatives.

Dominance rule 2: In partial schedule S, if event e has been created before event e' (which is not necessarily the child of e), all members of C_e must be added to the S before any member of $C_{e'}$.

V. Computational performance

We have performed a series of computational experiments using randomly generated test sets in order to examine the

performance of F-B&B and FB-B&B and analyze the effects of different parameter choices.

A. Experimental setup

All coding was done in the Visual C++ 6.0 environment; all experiments were run on a PC Pentium IV 3 GHz processor with 1024 MB of internal memory.

We have generated a test set, called base set and includes 60 test instances, using the random network generator RanGen (Demeulemeester et al. [17]). We choose n=6,8,10,12 and OS=0.4,0.6 and 0.8 where OS is the order strength, the number of precedence-related activity pairs divided by the theoretically maximum number of such pairs in the network. For each combination of n and OS we generate five instances, varying the alternatives' duration, cost and PTS: durations and costs are realizations of (independent) discrete uniform random variables on the intervals [1;10] and [-100;-10], respectively, and the PTS-values are, unless stated otherwise, with uniform probability chosen from [50%;100%]. We refer to the ratio of project payoff to sum of the alternatives cost, $C \Big/ \sum_{i \in N} C_i$ as the payoff-to-cost (PTC) index, which is a characteristic of a given project. Unless mentioned otherwise, we set PTC=5 and r=0.05.

B. Detailed computational results

In order to compare the performance of B&B, we take the average T_{Total} as the criterion. Table 2 shows the average T_{Total} (in seconds) spent by B&B for different numbers of alternatives and different values of OS.

Table 2. Average T_{Total} (in seconds)

		n			
		6	8	10	12
OS	0.4	0.028	2.384	143.909	6267.659
	0.6	0.028	2.384	143.909	6267.659
	0.8	0.003	0.013	0.153	3.053

VI. SUMMARY AND OUTLOOK ON FURTHER RESEARCH

In this paper, we reformulated the ATPSP and developed a branch-and-bound algorithm which outperforms the only available branch-and-bound algorithm in the literature. We proved that there is an optimal solution for each instance of the ATSP in which each event is simultaneous with at least another event. Based on this property, we constructed our brand-and-bound algorithm.

Since exact algorithm are not able to solve large scale problems, as a future research opportunity, we recommend to develop heuristic or metaheuristic algorithms for the problem. We also suggest to interesting researches to develop the setting of this problem to more practical assumptions like stochastic durations for alternatives or constraint of resources.

REFERENCES

[1] M.T. Pich, C.H. Loch, and A. de Meyer, "On uncertainty, ambiguity, and complexity in project management," Management Science, vol. 48(8), pp. 1008-1023, 2002.

[2] S.C. Sommer, and C.H. Loch, "Selectionism and learning in projects with complexity and unforeseeable uncertainty," Management Science, vol 50(10), pp. 1334-1347, 2004.

[3] M.A. Cohen, J. Eliashberg, and T.H. Ho, "New product development: The performance and time-to-market tradeoff," Management Science, vol. 42(2), pp.17–186, 1996.

[4] K.B. Hendricks, and V.R. Singhal, "The effect of product introduction delays on operating performance," Management Science, vol. 54 (5), pp. 878–892, 2008.

[5] B. De Reyck, Y. Grushka-Cockayne, and R. Leus, "A new challenge in project scheduling: The incorporation of alternative failures," Tijdschrift voor Economie en Management, vol. LII (3), pp. 411-434, 2007.

[6] M.L. Weitzman, "Optimal search for the best alternative," Econometrica, vol.47, pp. 641-654, 1979.

[7] R.R. Nelson, "Uncertainty, learning, and the economics of parallel research and development efforts, " The Review of Economics and Statistics, vol. 43(4), pp. 351-364, 1961.

[8] W.J. Abernathy, and R.S. R.S. Rosenbloom, "Parallel strategies in development projects," Management Science, vol. 15(10), pp. B486-B505, 1969.

[9] E. Dahan, "Reducing technical uncertainty in product and process development through parallel design of prototypes," Working Paper, Graduate School of Business, Stanford University, 1998.

[10] S.D. Eppinger, D.E. Whitney, R.P. Smith, and D.A. Gebala, "A model-based technology for organizing tasks in product development," Research in Engineering Design, vol. 6, pp. 1-13, 1994.

[11] V. Krishnan, S.D. Eppinger, and D.E. Whitney, "A model-based framework to overlap product development activities," Management Science, vol. 43(4), pp. 437-451, 1997.

[12] T.A. Roemer, and R. Ahmadi, "Concurrent crashing and overlapping in product development," Operations Research, vol. 52(4), pp. 606-622, 2004.

[13] R.H. Teunter, S.D.P. Flapper, "A comparison of bottling alternatives in the pharmaceutical industry," Journal of Operations Management, vol. 24, pp. 215-234, 2006.

[14] B. De Reyck, and R. Leus, "R&D-project scheduling when activities may fail," IIE Transactions, vol. 40(4), pp. 367-384, 2008.

[15] D.N. Ford, and D.K. Sobek, "Adapting real options to new product development by modeling the second Toyota paradox," IEEE Transactions on Engineering Management, vol. 52(2), pp. 175-185, 2005.

[16] H.W. Price, "Least-cost testing sequence," Journal of Industrial Engineering, vol. July-August, pp. 278-279, 1959.

[17] E. Demeulemeester, M. Vanhoucke, and W. Herroelen, "A random generator for activity-on-the-node networks," Journal of Scheduling, vol. 6, pp. 13-34, 2003.