Computers & Industrial Engineering 107 (2017) 39–56

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Scheduling multiple, resource-constrained, iterative, product development projects with genetic algorithms

Ali A. Yassine [a,*], Omar Mostafa [a], Tyson R. Browning [b]

[a] Department of Industrial Engineering & Management, American University of Beirut, Beirut, Lebanon
[b] Neeley School of Business, Texas Christian University, TCU Box 298530, Fort Worth, TX 76129, United States

ARTICLE INFO

Article history:
Received 15 August 2016
Received in revised form 22 January 2017
Accepted 1 March 2017
Available online 3 March 2017

Keywords:
Resource-Constrained Project Scheduling Problem (RCPSP)
Resource-Constrained Multi-Project Scheduling Problem (RCMPSP)
RCMPSP with Feedback (RCMPSPwF)
Iteration
Rework
Design Structure Matrix (DSM)
Genetic Algorithm (GA)

ABSTRACT

Many product development (PD) projects rely on a common pool of scarce resources. In addition to resource constraints, there are precedence constraints among activities within each project. Beyond the feed-forward dependencies among activities, in PD projects it is common for feedback dependencies to exist that can result in activity rework or iteration.

In such a multi-project, resource-constrained, iterative environment, this paper proposes two new genetic algorithm (GA) approaches for scheduling project activities. The objective is to minimize the overall duration of the portfolio of PD projects. These proposed GAs are tested on sample scheduling problems with and without stochastic feedback. We show that these algorithms provide quick convergence to a globally optimal solution.

Furthermore, we conducted a comparative analysis of the proposed GAs with 31 published priority rules (PRs), using test problems generated to the specifications of project, activity, and resource-related characteristics such as network density (complexity), resource distribution, resource contention, and rework probability (amount of iteration). The GAs performed better than the PRs as each of these factors increased. We close the paper by providing managers with a decision matrix showing when it is best to use the published PRs and when it is best to use the GAs.

1. Introduction

The timely delivery of new products and services is critical for the success and survival of organizations. Due to an increase in market competition, fast technological advancements, as well as evolving customers' needs and impatience, it has become very important to improve the efficiency with which projects are completed and new products are brought to market (Hendricks & Singhal, 1997; Herm, 2013; Hum & Sim, 1996; Majava, Haapasalo, Belt, & Mottonen, 2013). Moreover, many organizations are faced with the challenge of managing the simultaneous execution and management of a portfolio of development projects under tight time and resource constraints (Beaujon, Marin, & McDonald, 2001; Pennypacker & Dye, 2002; Rad & Levin, 2006). In such an environment, project management and scheduling skills become very critical to the organization. Herroelen (2005) mentioned that multi-project environments are quite common in project scheduling practice and offer many future research opportunities. He added that a large number of projects are carried out in a multi-project environment, and thus even a small improvement in their management will yield a large benefit to the project management field.

Project cost and schedule overruns have persisted for decades, in spite of numerous advances in the field of project management (Anderson & Tucker, 1994; Kerzner, 2013; PMI, 2013). Starting with the Project Evaluation and Review Technique (PERT) and the critical path method (CPM), network techniques have continued to evolve and advance (Mantel, Meredith, Shafer, & Sutton, 2007; Spinner, 1989). Advances include resource-constrained scheduling, resource leveling, and probabilistic risk assessments (Demeulemeester, 2002; Schwindt & Zimmermann, 2014). Additionally, alternative approaches to product and software development management such as the waterfall and spiral methods have been proposed (Unger & Eppinger, 2009).

The resource-constrained project scheduling problem (RCPSP) presents an extension to the standard CPM and PERT techniques by including the availability of resources during scheduling. However, an organization may often have several concurrent projects. While the projects may otherwise be unrelated, they depend on a common pool of resources. An overwhelming theme in the

* Corresponding author.
E-mail addresses: ali.yassine@aub.edu.lb (A.A. Yassine), omarmm19@gmail.com (O. Mostafa), t.browning@tcu.edu (T.R. Browning).

http://dx.doi.org/10.1016/j.cie.2017.03.001
0360-8352/

multi-project management literature is the issue of resource allocation between simultaneous projects (Engwall & Jerbrant, 2003). This extension is known as the resource-constrained multi-project scheduling problem (RCMPSP) (Kolisch and Padman, 2001).

RCPSPs and RCMPSPs are both strongly NP-hard, so there are no known algorithms for finding optimal solutions in polynomial time (Lenstra & Kan, 1978). Hence, most researchers have sought efficient heuristic and meta-heuristic techniques. Priority rule (PR) heuristics are the most common heuristics considered for very large problems, and are known for their speed, simplicity and ability to construct initial solutions (Browning & Yassine, 2010b). Several meta-heuristics have also been used, including simulated annealing, Tabu search, genetic algorithms, and ant colony optimization (Hartmann & Kolisch, 2000; Kolisch & Hartmann, 1998, 2006).

To complicate things further, in product development (PD) projects, scheduling activity iteration or rework has always been a challenge. Kang and Hong (2009) noted that the delays caused by activity iteration in a multi-project environment are as significant as those resulting from resource constraints. Even though the occurrence of iterations may not be known with certainty prior to project execution, a skilled manger can identify many potentially iterative activities and plan accordingly. Understanding the web of information flow in a project can help identify potential iterations (Eppinger, 2001). Unfortunately, many managers fail to plan for iterations in advance and integrate them into project schedule and cost estimation (Browning & Eppinger, 2002).

The main objectives of this paper are (1) to find an optimal or near-optimal duration distribution for a RCMPSP in the presence of stochastic activity iterations and (2) to compare this result to the performance of PRs. To achieve the first objective, this paper introduces two new genetic algorithm (GA) -based approaches. The procedure is based on a modified genetic encoding of a standard simple GA and the tailoring of its operators to suit this problem. To achieve the second objective, we utilized a full factorial experiment with randomly generated problem instances to demonstrate the superiority of the proposed GA-based approach by comparing our results to published PRs. We found significant differences in the performance of the GAs relative to the PRs in cases of high levels of iteration, network density (complexity), and resource constraints. Moreover, the GAs showed better convergence towards optimal solutions (shortest duration), especially in high-complexity and iterative projects. Finally, we organize these results for managers, distinguishing between the project and portfolio management perspectives.

The rest of the paper is organized as follows. In Section 2 we describe the relevant literature in traditional project management, design structure matrix (DSM), and GAs. Section 3 describes the two proposed GA approaches (Sampling and Variable-length GAs), discussing all parameters and methodologies and giving managers a guideline for scheduling decisions. Section 4 discusses the implementation of the problem, and Section 5 introduces the data used in the analyses and calibration of the model via sensitivity analyses of GA parameters and network characteristics. It also discusses model validation according to published benchmark studies of Kolisch and Sprecher (1997) and Browning and Yassine (2016). Furthermore, Section 5 describes the setup of and computational results from the comparison of the GAs with PRs. Section 6 summarizes the paper and draws final conclusions.

2. Literature review

The literature review covers both the DSM method and the RCMPSP. However, within this vast literature, we focus on meta-heuristic approaches, genetic algorithms (GAs) in particular. From this review, it will be noticed that DSM-based simulation techniques have not yet considered schedule optimization, nor have GA-based scheduling problems considered iteration (i.e. cyclic project networks such as PD), when dealing with RCPSPs. So, our paper fills this particular gap in the project scheduling literature.

2.1. Design structure matrix (DSM)

A DSM is an efficient and commonly used method of showing the relationships among the activities in a project (Yassine & Braha, 2003). Given a set of n activities in a project, the corresponding DSM is an $n \times n$ matrix where the project activities are the row and column headings listed in the same order. The precedence relationships among activities appear in the off-diagonal elements of the matrix. If activity j depends on activity i (that is, i feeds j), then the value of element ij (column i, row j) is one (or flagged with a mark such as "X") in a binary DSM. Otherwise, the value of the element is zero (or left empty) (Yassine, 2004). If the activities are executed in the same order as they appear in the DSM, then marks below the diagonal represent forward information from activity i to j, while those above the diagonal represent feedback information from activity j to i.

Feed-forward dependencies are typical to deal with, but feedback dependencies are more challenging from a scheduling point of view. The latter dependencies exist due to the uncertainty in performing some activities with a lack of information. As new information emerges throughout the development process, this uncertainty is resolved and may either validate how activities were performed or reveal some mistakes that require (first-order) rework, repetition, or iteration. Rework in an upstream activity, caused by downstream activity, can also cause *second-order rework* (Browning & Eppinger, 2002) due to the cascading of changes through interim activities. Thus, the super-diagonal marks represent the probability of iteration (returning to previous activities), while the sub-diagonal marks note the probability of second-order rework (following any first-order rework). The DSM that contains such rework probabilities is called a Probability DSM. Browning and Eppinger (2002) also introduced an Impact DSM, which includes impact values (between 0 and 1) representing the rework percentages of an activity's initial duration.

Several DSM-based simulation models (and tools) exist in the literature (e.g., Abdelsalam & Bao, 2006; Browning & Eppinger, 2002; Zhuang & Yassine, 2004). Although all of these models are mainly aimed at determining the process completion time and cost for a given task network structure (process architecture), some also have an added component for determining the optimal architecture (e.g., Abdelsalam & Bao, 2006; Kang & Hong, 2009; Zhuang & Yassine, 2004) by testing (i.e., calculating the time and cost of) various architectural arrangements. Abdelsalam and Bao (2006) presented a simulation-based optimization framework that determines the sequence of activity execution in a PD project with minimal total duration based on an "iteration factor" representing coupling strength and the number of iterations required for convergence. Zhuang and Yassine (2004) sampled several potential iterative scenarios from an existing DSM model, where each scenario represents a project without feedback. Then, these projects were scheduled using a simple GA scheduling algorithm. However, none of these DSM-based simulation/optimization models have accounted for resource constraints across multiple projects.

2.2. Genetic algorithms (GAs)

Metaheuristic algorithms became an important part of modern optimization (Goldberg, 1989; Holland, 1975). A wide range of metaheuristic algorithms have emerged over the last two decades, and many metaheuristics such as genetic algorithms (GA) and