Optimal Scheduling of Work-Content-Constrained Projects

Philipp Baumann, Norbert Trautmann
Department of Business Administration, University of Bern, Schuetzenmattstrasse 14, 3012 Bern, Switzerland
e-mail: philipp.baumann@pqm.unibe.ch, norbert.trautmann@pqm.unibe.ch

Abstract - The execution of a project requires resources that are generally scarce. Classical approaches to resource allocation assume that the usage of these resources by an individual project activity is constant during the execution of that activity; in practice, however, the project manager may vary resource usage over time within prescribed bounds. This variation gives rise to the project scheduling problem which consists in allocating the scarce resources to the project activities over time such that the project duration is minimized, the total number of resource units allocated equals the prescribed work content of each activity, and various work-content-related constraints are met.

We formulate this problem for the first time as a mixed-integer linear program. Our computational results for a standard test set from the literature indicate that this model outperforms the state-of-the-art solution methods for this problem.

Keywords - Project scheduling, work-content constraints, mixed-integer linear programming

I. INTRODUCTION

A dynamic environment requires firms to execute many projects such as the development of new products and services. A project is a unique endeavor that can be divided into activities each of which requires time and scarce resources for its completion. Due to technological or organizational restrictions, these activities usually must be performed according to precedence constraints. The planning of such projects includes *temporal scheduling*, which consists of determining the early start schedule, the late start schedule and the slack times for each of the activities, and *resource allocation*, which consists in allocating scarce resources to the execution of the activities over time. The efficient allocation of scarce resources to the execution of the project activities is of particular importance for companies that wish to remain competitive. In classical project scheduling formulations, it is assumed that activities have fixed durations and constant resource requirements. These assumptions are often too restrictive for practical applications, in which the decision maker may change the resource usage of an activity over time. Examples of such applications can be found in the pharmaceutical and software development industries. Typically, the availability of a single renewable work-content resource (e.g., labor) is constant throughout the project, and an individual amount of work content (e.g., number of person-days) is specified for each activity. Activities can be executed in various ways, i.e., the number of units of the work-content resource allocated to an activity may vary over time, as long as the requirement for the total work content is met. This flexibility allows for a more efficient usage of the work-content resource. In addition to the work-content resource an activity often requires additional resources, such as tools or supplies. In this paper, we assume that the amount of the work-content resource used by an activity determines the activity’s requirement for further resources. Currently, most project planning software packages allow the specification of individual work contents for activities, which emphasizes the relevance of this concept to project managers.

In this paper, we consider the problem of scheduling the activities of a project to minimize the project duration; thereby, the activities (a) are subject to finish-start precedence relationships, (b) require a prescribed number of units of the work-content resource, (c) require units of multiple resources in addition to the work-content resource and (d) may require variable amounts of the work-content resource during their execution, subject to some organizational restrictions. These restrictions include a lower and an upper bound on the usage of the work-content resource and a minimum time lag between consecutive changes of the usage of the work-content resource. We refer to this time lag as the minimum block length.

For this project scheduling problem, Fündeling [1] presents an exact branch-and-bound method. Although this approach is tailored to the specific problem setting, only few and small-sized problem instances can be solved to optimality. To tackle large-scale instances, Fündeling and Trautmann [2] develop a priority-rule method that schedules activities iteratively using a serial schedule-generation scheme. In each iteration, a start time and a feasible resource profile are determined for an activity. To construct the resource profile, the work-content resource is allocated iteratively period by period. A consistency test is thereby used to exclude allocations that would violate the minimum block length constraint. The quality of the generated schedules cannot be evaluated thoroughly because of the lack of a sufficiently large number of optimal schedules. Both approaches address the problem in which, in a feasible schedule, the total number of resource units allocated to each activity must coincide with its prescribed work content. Alongside, the discrete time/resource trade-off problem has been discussed in the literature. In this problem, only the work-content resource is considered, and an execution mode must be selected for each activity; each mode corresponds to a combination of a duration and a constant resource usage such that the total number of resource units allocated to the activity is equal to or greater than its prescribed work content. Respective literature surveys are provided in Fündeling and Trautmann [2] and Weglarz et al. [3].

In this paper, we formulate the project scheduling problem with work-content constraints as a mixed-integer linear program (MILP). The MILP model is then used to devise optimal solutions for small instances to evaluate existing methods. Due to its general structure, the proposed formulation can easily
be adapted to different project scheduling problems, including
the well-known RCPSP; in this sense, we also contribute
to the recent development of novel MILP formulations for
various types of project scheduling problems (cf., e.g., Rieck
et al. [4] or Bianco and Caramia [5]). We have applied
the proposed formulation to 480 problem instances with 10
activities that were introduced in Fündeling and Trautmann [2].
The proposed model solves 400 out of the 480 instances to
optimality within short CPU times.

The remainder of the paper is organized as follows. In
Section II, we describe the planning problem. In Section III,
we present the MILP formulation. In Section IV, we report
our computational results. In Section V, we provide some
concluding remarks.

II. PLANNING PROBLEM

Given are a set of activities \( V \), a set of discrete resources
\( R \) and a set of finish-start precedence relationships between
the activities. We assume that all resources are renewable
and that their capacities are constant over time. The set of
resources \( R \) consists of the work-content resource \( k^* \) and the
non-work-content resources \( k \in R \setminus \{ k^* \} \). For each activity \( i \),
a work content \( w_i \in \mathbb{Z}_{\geq 0} \) is given that represents the total
number of units of the work-content resource \( k^* \) required
for its execution. The allocation of the work-content resource
per period can vary over integer values between a prescribed
lower bound \( \underline{q}_{ik} \) and a prescribed upper bound \( \overline{r}_{ik} \). The
requirements of the non-work-content resources depend on the
usage of the work-content resource. Here, a linear relation
between the requirement of the non-work-content resources
and the usage of the work-content resource is assumed, i.e., an
increase (decrease) in the usage of the work-content resource
results in a proportional increase (decrease) in the requirement
for further resources. However, because we assume that all
resources are discrete in nature, fractional requirements are
rounded up. Let \( R_{ikt} \) denote the requirement of resource
\( k \in R \) by activity \( i \) in period \( t \). The requirement \( R_{ikt} \) of
resource \( k \in R \setminus \{ k^* \} \) is computed as follows.

\[
R_{ikt} = \lceil \underline{q}_{ik} + s_{ik}(R_{ikt-1} - \overline{r}_{ik}) \rceil
\]

where \( \underline{q}_{ik} \) denotes the minimal requirement of resource \( k \) by
activity \( i \), and \( s_{ik} \) represents the increment per additional unit
of the work-content resource \( k^* \). The parameter \( s_{ik} \) equals

\[
s_{ik} = \frac{r_{ik} - \underline{q}_{ik}}{\overline{r}_{ik} - \underline{q}_{ik}}
\]

where \( r_{ik} \) denotes the maximal requirement of resource \( k \) by
activity \( i \); we set \( s_{ik} = 0 \) if \( \underline{q}_{ik} = \overline{r}_{ik} \). Figure 1 illustrates the
requirement of a non-work-content resource \( k \) by activity \( i \),
given the usage of the work-content resource \( k^* = 1 \). The
dotted lines represent the respective fractional values of the
resource requirement, which must be rounded up to integer
values. The minimum block length is denoted by \( m \), i.e., a
time lag of at least \( m \) periods is required between consecutive
changes of the usage of the work-content resource. Such
changes include the start and the end of an activity.

\[
\begin{array}{c}
R_{i1t} \\
\tau_{i1} = 6 \\
L_{i1} = 1 \\
1 2 3 4 5 6 \\
1 2 3 4 5 6 \\
\end{array}
\]

\[
\begin{array}{c}
R_{i2t} \\
\tau_{i2} = 5 \\
L_{i2} = 2 \\
1 2 3 4 5 6 \\
1 2 3 4 5 6 \\
\end{array}
\]

A feasible allocation of the work-content resource to the
execution of the activities is sought such that the precedence
relationships are satisfied, all activities are scheduled without
interruption, the capacities of the resources are never exceeded,
and the duration of the project is minimized.

III. MILP SCHEDULING MODEL

In this section, we formulate the project scheduling problem
described in Section II as an MILP. Several formulations for
general project scheduling problems in the literature are based
on a discrete representation of time. These formulations can
be divided into three categories based on the type of decision
variables used. The first category is based on binary variables
\( X_{it} \) that equal 1 if the activity \( i \) starts or ends in period \( t \) (cf.,
e.g., Pritsker et al. [6]). In the second category of formulations,
binary variables \( X_{it} \) equal 1 if activity \( i \) is processed in period
\( t \) (cf., e.g., Kaplan [7]). In the third category, binary variables
\( X_{it} \) equal 1 if activity \( i \) is in progress or has been processed
before period \( t \) (cf., e.g., Klein [8]). In addition, there exist
flow-based and event-based continuous-time formulations that
comprise sequencing variables or event variables, respectively
(cf., e.g., Koné et al. [9]).

In this paper, we present a formulation that is based on a
discrete representation of time and on binary variables \( X_{it} \) that
are equal to 1 if the activity \( i \) is processed in period \( t \) and are
equal to 0 otherwise. In Subsection III-A, we introduce the
notation. In Subsections III-B and III-C, we discuss the time-
related and resource-related constraints of the formulation,
respectively. In Subsection III-D, we introduce the objective
function and summarize the optimization problem.
A. Symbols

Indices

\( i \) Activity

\( t \) Time period

\( k \) Resource

\( k^* \) Work-content resource

Sets

\( V \) Activities (\( V = \{ 1, \ldots, n + 1 \} \))

\( T \) Periods (\( T = \{ 1, \ldots, LFT_{n+1} + 1 \} \))

\( R \) Resources

\( V^r \) Real activities

\( V^r_{kt} \) Real activities that can be processed in period \( t \in T \) and require resource \( k \in R \)

\( T_i \) Relevant periods for activity \( i \)

\( R_i \) Resources required by activity \( i \)

\( P_i \) Immediate predecessors of activity \( i \)

Parameters

\( w_i \) Work content of activity \( i \)

\( r_{ik} \) Upper bound on the amount of resource \( k \in R \) used by activity \( i \)

\( \sum_{ik} \) Lower bound on the amount of resource \( k \in R \) used by activity \( i \)

\( m \) Minimum block length

\( s_{ik} \) Increment of requirement for resource \( k \in R \setminus \{ k^* \} \) by activity \( i \) per unit of work content

\( R_{ik} \) Capacity of resource \( k \in R \)

Integer decision variable (non-negative)

\( R_{ikt} \) Amount of resource \( k \in R \) used by activity \( i \in V^r_{kt} \) in period \( t \in T_i \)

Binary decision variables

\( X_{it} \) \[
\begin{cases} 
= 1, & \text{if activity } i \text{ is processed in period } t \in T_i \\
= 0, & \text{otherwise}
\end{cases}
\]

\( D_{it} \) \[
\begin{cases} 
= 1, & \text{if the amount of resource } k^* \text{ used by activity } i \in V^r \text{ in period } t \in T_i \text{ differs from period } t-1 \\
= 0, & \text{otherwise}
\end{cases}
\]

B. Time-related constraints

For each activity \( i \in V \), we determine the set of available periods \( T_i \) based on the earliest start and the latest finish times \( (EST_i, LFT_i) \). The decision variables related to activity \( i \) are defined only for these periods. The earliest start times are computed by forward recursion. The earliest start time of an activity is thereby set to the latest of the earliest finish times of all of its immediate predecessors. The earliest finish time of an activity is computed by adding the lower bound on its duration \( [w_i/r_{ik^*}] \) to its earliest start time. The latest finish times are computed similarly using backward recursion. Thereby \( LFT_{n+1} \) must be sufficiently large to guarantee feasibility. Constraint (1) ensures that the project completion, i.e., the dummy activity \( n+1 \), is scheduled within the planning horizon.

\[ \sum_{t \in T_{n+1}} X_{n+1,t} = 1 \]  

Constraints (2) prevent activities from being interrupted. If activity \( i \) is executed in period \( t-1 \) and its work content has not been entirely processed by the end of this period, then activity \( i \) must also be processed in period \( t \).

\[ X_{i,t-1} = \sum_{v \in V^r; t' < t} R_{ik^*t'} \leq X_{it} \]  

The finish-to-start precedence relationships are formulated by Constraints (3). Activity \( i \) can start only if the total work content of all of its predecessors has been processed.

\[ X_{it} \leq \sum_{v \in V^r; t' \leq t' < t} R_{ik^*t'} \]  

C. Resource-related constraints

Constraints (4) guarantee that the work content of each activity is processed exactly during its execution.

\[ \sum_{t \in T_i} R_{ik^*t} = w_i \quad (i \in V^r) \]  

Constraints (5) and (6) impose lower and upper bounds on the usage of the work-content resource, respectively.

\[ \sum_{ik} X_{it} \leq R_{ik^*t} \quad (i \in V^r; \ t \in T_i) \]  

\[ R_{ik^*t} \geq X_{it} \quad (i \in V^r; \ t \in T_i) \]  

Constraints (7), (8) and (9) force the variable \( D_{it} \) to be valued 1 if the usage of the work-content resource by activity \( i \) in period \( t-1 \) differs from its usage in period \( t \). When an activity is completed in period \( t = LFT_i \), the variable \( D_{it} \) in the period \( t = LFT_i + 1 \) will capture the last change in the usage of the work-content resource.

\[ R_{ik^*t} \leq r_{ik^*} D_{it} \quad (i \in V^r; \ t = EST_i + 1) \]  

\[ R_{ik^*t} - R_{ik^*t-1} \leq r_{ik^*} D_{it} \quad (i \in V^r; \ t \in T_i : \ t > EST_i + 1) \]  

\[ R_{ik^*t-1} - R_{ik^*t} \leq r_{ik^*} D_{it} \quad (i \in V^r; \ t \in T_i : \ t > EST_i + 1) \]  

Constraints (10) prevent an activity \( i \) from being processed after its latest finish time.

\[ X_{it} = 0 \quad (i \in V^r; \ t = LFT_i + 1) \]  

The minimum block length is ensured through constraints (11), which allow only one change in the usage of the work-content resource during \( m \) consecutive periods \( t \in T_i \).

\[ \sum_{t' = 0}^{m-1} D_{i,t+t'} \leq 1 \quad (i \in V^r; \ t \in T_i : \ t \leq LFT_i - (m - 2)) \]
The requirement for resource \( k \in \mathcal{R} \setminus \{k^*\} \) by activity \( i \) in period \( t \) is computed by constraints (12). As this requirement must be an integer value, we use integer variables \( R_{i,k,t} \) to round up fractional values.

\[
\sum_{t \in T_i} X_{n+1,i,t} - 1 \leq R_{i,k,t} \tag{12}
\]

Constraints (13) ensure that the total requirement for each resource \( k \in \mathcal{R} \) does not exceed its capacity \( R_k \).

\[
\sum_{i \in V_k} R_{i,k,t} \leq R_k \quad (k \in \mathcal{R}; \ t \in T) \tag{13}
\]

**D. Objective function**

The objective is to minimize the duration of the project. The dummy activity \( n+1 \) represents the end of the project, as it can be scheduled only after the completion of all real activities \( V^r \). The optimization problem reads as follows.

\[
\begin{aligned}
& \text{Min.} \quad \sum_{t \in T_{n+1}} tX_{n+1,i,t} - 1 \\
\text{s.t.} \quad & R_{i,k,t} \in \mathbb{Z}_{\geq 0} \quad (i \in V^r ; \ k \in \mathcal{R}; \ t \in T_i) \\
& D_{i,r} \in \{0,1\} \quad (i \in V^r ; \ t \in T) \\
& X_{n+1,t} \in \{0,1\} \quad (i \in V ; \ t \in T) \\
\end{aligned}
\]

**IV. COMPUTATIONAL RESULTS**

In this section, we apply the proposed MILP model to an illustrative example (cf. Subsection IV-A) and to a set of 480 problem instances introduced in Fündeling and Trautmann [2] (cf. Subsection IV-B). We have used the method of Fündeling and Trautmann [2] to compute upper bounds on the makespan of these instances. To facilitate the computation of a feasible solution, we have set the latest finish time of the dummy activity \( n+1 \) to this upper bound plus two periods. We have implemented the proposed MILP formulation in AMPL and used Gurobi 5.5 as a solver on a standard workstation with two Intel Xeon 3.1GHz CPUs and 128GB RAM.

**A. Illustrative Example**

We consider a project with three real activities \( |V^r| = 3 \) and two resources \( |\mathcal{R}| = 2 \). Both the work-content resource \( k^* = 1 \) and the non-work-content resource \( k = 2 \) have capacities of 4. There is a finish-start precedence relationship between activities 1 and 2, i.e., activity 2 can only start when activity 1 has been completed. The minimum block length \( m \) corresponds to three periods. The remaining data for the illustrative example are given in Table I. With the priority-rule method of Fündeling and Trautmann, we computed a schedule for this project with makespan 14. Based on this upper bound, we set the latest finish time of activity \( n+1 \) to 16 and the number of periods \( |P| \) for the MILP model to 17. Figure 2 shows the schedule of an optimal solution and the corresponding values of all of the decision variables; the dots indicate that the decision variable has not been defined for the respective period. The solution shown in Figure 2 was found in less than 1 second of CPU time.

**B. Numerical results**

To evaluate the performance of the MILP model, we applied the model to the 480 problem instances introduced in Fündeling [1]. Each instance represents a project with 10 activities, one work-content resource and up to three non-work-content resources. The instances were generated by systematically varying three complexity parameters. The parameter order strength \( OS \) defines the density of the project network and was chosen from the set \( \{0, 0.25, 0.5, 0.75\} \); the higher the value, the more dense the project network. The parameter resource factor \( RF \) defines the mean percentage of resources used by an activity and was chosen from the set \( \{0, 0.25, 0.5, 0.75\} \). The parameter resource strength \( RS \) defines the degree of resource scarcity and was chosen from the set \( \{0.025, 0.5, 0.75\} \); the lower the value, the scarcer the resources. The test set contains 10 instances for each parameter combination. The minimum block length was randomly chosen from the set \( \{2, 3, 4\} \). The parameters \( w_1 \), \( \sum_{i,k} \) and \( \Gamma_{ik} \) were selected randomly such that a feasible solution exists for each instance. For further details, we refer to Fündeling and Trautmann [2].

Due to the complexity of problem (P), some instances require large CPU times. Therefore, we prescribed a time limit of 600 seconds per instance. A feasible solution was found for 426 instances, and solution optimality was proven for 400 instances. Table II lists the results for different combinations...
of the parameters RS and \( m \). These two parameters appear to drive the CPU time requirement of this test set. In columns 3 to 5 of Table II, we report the number of existing instances with the respective combination of RS and \( m \), the number of instances for which a feasible solution was found within the time limit, and the number of instances for which optimality could be proven within the time limit, respectively. In columns 6 and 7, we report the average and the maximum MIP gap of the instances for which a feasible solution was found but for which optimality could not be proven within the time limit. In column 8, we state the average relative deviation of our solutions to the lower bounds computed in Fündeling and Trautmann [2]. In columns 9 and 10, we list the average and the maximum relative deviations of our solutions from the solutions found by the multi-start priority-rule method of Fündeling and Trautmann [2].

We conclude that the longer the minimum block length or the scarcer the resources, the more CPU time is required to find a feasible solution and to prove optimality. The latter observation may be explained by the fact that the number of feasible resource profiles for an activity decreases with increasing minimum block length. If a feasible solution can be found within the time limit, the MIP gap is generally rather small. Our schedules can be up to 16.3% shorter than the schedules obtained by the priority-rule method of Fündeling and Trautmann [2]. However, the method of Fündeling and Trautmann [2] provides a feasible solution to all 480 instances.

V. CONCLUSIONS

For the work-content-constrained project scheduling problem, heuristic solution methods and a specific branch-and-bound procedure have been proposed in the literature. In this paper, this problem has been formulated for the first time as an MILP. In particular, the model covers the minimum block-length constraint and considers the dependent requirement for non-work-content resources. We have performed an experimental performance analysis with a set of 480 test instances previously introduced by Fündeling [1]. For 426 instances, the model found an optimal or near-optimal solution within a short CPU time; in these 426 instances, the proposed model outperforms the state-of-the-art method.

The model presented in this paper contributes to the development of efficient MILP formulations of resource-allocation problems that can then be solved with standard software. Our model will help to formulate exact models for related applications such as those discussed, e.g., in Kolisch et al. [10]. The possible extensions of this model include work-content-based precedence relationships, i.e., only a prescribed percentage of the work content of an activity must be completed before the succeeding activity can be started, as well as time-varying resource capacities. The model could easily be modified to consider these extensions by adapting constraints (3) and (13). Moreover, to further evaluate the performance of the proposed model, we plan to develop MILP formulations that are based on different types of decision variables.

REFERENCES