A methodology for scheduling overlapped design activities based on dependency information


Engineering Management, American University of Beirut, Lebanon

ARTICLE INFO

Article history:
Accepted 28 August 2012
Available online xxxx

Keywords:
Dependency structure matrix
Design
Fast-tracking
Information exchange
Overlapping
Scheduling

ABSTRACT

The practice of overlapping activities is becoming a requirement for fast-tracking complex construction projects. The amount, timing, and nature of the information exchanged between pairs of activities determine the degree to which pairs of activities may be overlapped. This paper presents a four-step process for scheduling the design phase of fast-tracked construction projects while taking into consideration information exchange among project activities. The process starts with capturing and quantifying this exchange of dependency information. A contemporary scheduling tool, the dependency structure matrix (DSM), aids in generating the shortest (overlapped) schedule based on dependencies among the different design disciplines. An algorithm is designed to calculate the shortest possible schedule for the design phase of a construction project. The developed scheduling algorithm is unique as it includes information exchange alongside task durations. The algorithm is validated in the context of a real-world case study, a fast-tracked multi-billion dollar educational facility project in the Arabian Peninsula.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Generally, there are two main contracting methods for delivering a construction project: the traditional (sequential) method and the fast-track (phased) method. In the traditional method, an activity starts once its predecessors are completed. For example, the construction phase of a project starts after the design phase is fully completed. Alternatively, in the fast-track method some amount of overlap occurs between pairs of activities such as design and construction. This method has become more popular in recent years in response to growing industry demands [1].

A key requirement for the success of the fast-track method is the timely release of information to the construction group by the project designers. As such, design managers must schedule design work in the shortest possible manner while taking into account construction requirements. This is typically achieved by overlapping, and possibly compressing the duration of, design disciplines. For example, the design of the architecture discipline proceeds in parallel with the design of other dependent disciplines such as structures and finishes. This requires a two-way exchange of information among dependent design disciplines [2]. Design of architectural features, for instance, typically kicks off the design process of a project. At some point in this process, preliminary architectural information (e.g., general layouts and sections) is shared with other design groups (e.g., structures, finishes) so that they can start the design of their disciplines. At a later point in this process, information is shared in the opposite direction (i.e., bi-directional information dependency and exchange). For example, interior design and finishing plans are shared with the architectural group, who awaits this information before concluding its part of the design. Currently, there are no formal procedures or computerized tools to model this exchange of information and guide design managers in making such overlapping decisions. Practitioners rely on experience and standard scheduling techniques (e.g., critical path method) to schedule design activities. However, such techniques do not allow for overlapping, or for a two-way exchange of information, between a pair of dependent activities. This leaves practitioners with a set of unanswered questions. How much overlapping is acceptable among design disciplines? Specifically, how much design work must be completed in a discipline to proceed with the design of the following one? How can this information be presented in a simple format and be fed into the project schedule?

By providing answers to these questions, this paper delivers a set of guidelines to help design companies create a compressed design schedule. The goal of this study is two-fold. First, it provides a methodology and a computerized tool to automatically generate a fast-track design schedule without violating dependency information. Second, it extends the basic dependency structure matrix (DSM) method, which is mainly used in product development projects, to accommodate the special features of construction projects. The novelty lies in...
the use of the DSM not only to represent two-way exchange of dependency information among design activities, but also to schedule the design phase of construction projects with the highest possible amount of overlap. The paper proceeds with a review of the relevant literature. This is followed by a presentation of the proposed four-step approach to schedule the design phase of construction projects while taking into account information exchange among design disciplines. The proposed approach is then illustrated in the context of a real-world case study project.

2. Background

Overlapping of sequential activities occurs on most construction projects, fast-track and non-fast-track alike. For example, even in projects executed under the traditional contracting method, construction activities often start with incomplete design drawings or missing detail in the shop drawings. Nonetheless, the amount and likelihood of overlapping intensifies on fast-track projects, which are bound by strict time constraints.

Fast-tracking falls under the umbrella of concurrent engineering and is described as overlapping successive activities to reduce project duration [3]. A more specific definition of fast-tracking is the compression of design and production schedules by overlapping activities [4]. Blackburn [5] documented the success of concurrent engineering in reducing product delivery times in the manufacturing industry by as much as 20 to 50%.

Considering downstream concerns in upstream phases of project design, as opposed to a strictly sequential stage-gate workflow, is the core philosophy of concurrent engineering. Yassine and Braha [6] report that, in the manufacturing industry, the primary benefits of concurrent engineering are achieved by overlapping product and manufacturing design phases. The basic overlapping model deals with overlapping two sequential activities, referred to as an upstream feeding activity (A) and a downstream dependent activity (B) (see Fig. 1a). The main objective of a basic overlapping problem is to find the greatest possible overlapping magnitude between A and B that minimizes total lead-time (see Fig. 1b). The trade-off encountered with this objective is: starting activity B earlier (based on preliminary upstream information) gives the downstream stage a head-start, but runs the risk of wasting valuable development resources if excessive downstream rework must occur as a consequence of using early and premature upstream information [7]. Due to this tradeoff, overlapping requires the accurate and active management of interactions and communication [8,9]. Potential candidates for overlapping and the appropriate intensity of overlap must be carefully selected [10]. Evaluating the effects of overlapping in a comprehensive industry-specific study, Loch and Terwiesch [11] confirmed the time savings of overlapping in general while arguing that its specific impact depends on a firm’s capability to resolve uncertainty early in the process.

Construction researchers did not take long to implement concurrent engineering principles in their industry. Similar to manufacturing, construction activities can be classified as upstream or downstream. The downstream category includes construction, operation, maintenance, and decommissioning, while upstream includes project conception, specification, and design [12]. Implementing concurrent engineering on construction projects involves starting a downstream activity (e.g., construction) before the completion of an upstream activity (e.g., project design). This, in turn, comes with the risk of costly rework in the event of major late design changes [13,14]. Scholars such as Smith [15] and Shang et al. [16] found that the improvement in project delivery time offered by fast-tracking or concurrent engineering does not necessarily come at the expense of life-cycle cost or quality. According to Williams [17], fast-tracking design and construction has been successful in reducing the project schedule on multiple projects with little cost increase.

Jaafari [18] studied fast-tracking in the construction industry in terms of total life-cycle management of capital projects. He presented a framework for implementing concurrent engineering in projects from planning through commissioning, thereby incorporating information from downstream activities into the execution of upstream activities. The level to which upstream and downstream activities may be overlapped depends on the type of information exchanged and the nature of dependency between those activities [19]. The nature of dependency, in turn, is a function of upstream task evolution and downstream task sensitivity [20]. The former is an indication of how fast upstream tasks need information to be completed, whereas the latter is an indication of how sensitive downstream tasks are to changes in an upstream activity [21].

Based on information gathered from sector-based case studies, Bogus et al. [19] offered strategies to speed up the evolution of upstream information or reduce the sensitivity of downstream activities. The strategies identified include early freezing of design criteria, overdesign, and early release of preliminary information. Other factors that affect the amount of sensitivity in construction activities include the level of transformation (e.g., irreversible processes), lead time, modularity, and the interaction of built components [1].

Eldin [22] presented four case studies and concluded that applying concurrent engineering standards could reduce the project schedule by up to 25% in comparison with the schedule under the traditional delivery process. Similarly, Attar et al. [23] found that concurrent engineering reduces the schedule of underground construction projects by 18%. Both studies also concluded that the construction industry was missing a definite model and computer tool for making overlapping decisions.

Several studies attempted to qualify and quantify the overlap of traditionally sequential activities through analytical approaches. These include virtual design team [24], system dynamics [20], Monte Carlo models [25], dynamic project simulation [9], sequencing algorithms [26] and genetic algorithms [27]. While they illustrate how tools can be used to overlap sequential activities, these studies fall short of qualifying and quantifying the information exchanged between pairs of project activities, especially in cases of two-way exchange of information. As indicated by Choo et al. [28] and Maheshwari and Varghese [29], analyzing information dependency relationships among design activities is difficult especially for new projects. Currently, there is no comprehensive framework that specifies how, and how much, to overlap design activities based on information dependencies in real-world construction projects.

This paper builds on earlier work (e.g., [30–34]), which advocate the use of the DSM tool to build design schedules while taking into consideration dependency information. In particular, Baldwin et al. [30] and Austin et al. [31] recognized the need for using innovative techniques (e.g., DSM) to accurately represent information requirements while scheduling dependent activities. Subsequent work by Senthilkumar et al. [32], Senthilkumar and Varghese [33], and Jeevan and Varghese [34] demonstrated the use of DSM to represent the exchange of information among pairs of dependent activities. Maheshwari et al. [35] recommended further studies to reduce the efforts entailed in estimating information dependency ratings.

This paper uses the DSM tool not only to represent bi-directional information dependency and exchange, but also to devise a scheduling

![Fig. 1. A sequential product development process (a) can be shortened via overlapping (b).](image-url)
algorithm which allows for maximum overlapping between activities in the design phase of construction projects. The selection of the DSM technique, as opposed to other traditional scheduling techniques, is due to several reasons. First, unlike traditional techniques (e.g., network analysis, bar charts) which are based on the completion of work elements, a DSM allows for scheduling a process on the basis of production of information [35]. Hence, it allows for partial exchange of information, addressing the case where work on a downstream activity requires the completion of only a fraction of work on an upstream activity. Second, DSMs allow for, and incorporate, iteration in the design process, an activity that is not permitted in traditional scheduling techniques. Third, DSMs provide a compact representation and improved visibility over other scheduling tools. For example, as the number of activities and relationships grows, DSMs provide advantages over digraphs in the visualization of feedback loops, which are immediately obvious as upper-diagonal marks. As such, the use of DSMs allows for an accurate representation of information exchanged between pairs of design activities.

The use of a matrix format to represent complex project information dates back to the mid-1960s. Steward [36] proposed the use of design structure matrix analysis to represent the interrelationships between design activities. More recent studies [37–41] extended the work to include other project phases such as manufacturing or construction. Given the wider application, the technique is commonly referred to as a dependency structure matrix.

3. Methodology

This paper introduces a four-step DSM-based framework for generating a fast-track design schedule. The first step is to quantify the types, and level, of information exchanged among design activities. The second step is to incorporate this information into a DSM. The third step consists of formulating an algorithm to derive the shortest possible design duration using the built DSM. The fourth and last step is to generate a bar chart schedule showing the duration of design activities. Fig. 2 illustrates an overview of the four-step framework. Each of the four steps is carefully detailed in the following sections.

4. DSM-based framework for scheduling design activities

4.1. Step 1: quantify design dependencies (Fig. 2a)

The first step in the proposed framework is to quantify the types, and level, of information exchange among design activities. This requires the breakdown of the project design into major activities and studying the exchange of information among these activities.

For example, design is typically organized into a set of disciplines: Substructure (SBS), Superstructure (SPS), Façade (FAC), Architecture (ARC), Mechanical/Electrical/Plumbing (MEP), and Finishes (FIN). These design disciplines might be dependent and therefore they cannot be executed in parallel. Some of these disciplines may require information from others to start. The design of SBS, for instance, relies on information provided by ARC (e.g., layout plans, sections, and elevations) which are needed to set the locations of the main structural elements (e.g., determine the height of columns, complete structural design of the foundation).

Collecting dependency information among design disciplines requires a thorough data collection effort such as the one shown in Fig. 3. The first stage of the depicted process consists of holding interviews with group leaders of major design disciplines and seeking their feedback on the type, and amount, of information delivered to the other design disciplines. The interviewees can use their previous experience to estimate the amount of work done in a certain design deliverable before information is shared with other disciplines. For example, information collected from ARC includes the various deliverables (e.g., design of doors and windows) that are typically shared with other design disciplines (e.g., FAC), and an estimate of the effort spent on these deliverables relative to the overall amount of work spent on all ARC deliverables (e.g., walls, doors, windows, room tags, axes, dimensions, references, etc.). Table 1 presents a typical template for representing the collected dependency information. Work for each design discipline is represented by a set of deliverables. For example, ARC has deliverables AR1 through ARn. Each of these deliverables has an associated weight (e.g., w1 through wn) representing the effort spent to complete it relative to the effort spent to complete the entire design discipline. Table 1 also shows an estimate of the amount of work (dARC-FAC) that a dependent discipline (e.g., FAC) requires from each of the design deliverables (e.g., ARi).

The second stage of the data collection process is to validate the gathered information. This is done by cross-checking the obtained quantities of work with other sources of information. These include double-checking the figures with the total number of design drawings exchanged between pairs of design disciplines or with another design group leader working on another project. Should there be major discrepancies among the sources of data (e.g., 20% or more), the process should be re-initiated. In other cases, the collected data from the original source (i.e. design discipline group leader) is used.

4.2. Step 2: convert dependency information into a DSM (Fig. 2b)

Step 2 of the proposed framework consists of converting the information collected in Step 1 into a DSM format. As shown in Fig. 4, a
DSM provides a matrix representation for the underlying project schedule network [42]. The n activities of the project correspond to the n column and row headings in the matrix. The matrix shows activities along its diagonal, activity outputs in its columns, and activity inputs in its rows. The marks inside the matrix correspond to the predecessor relationships between the activities. Pairs of activities can be generally classified as dependent (one way exchange of information), independent (no exchange of information), or interdependent (two way exchange of information). For example, in Fig. 4, activities A and C are dependent, E and F are independent, and B and D are interdependent. The feed-forward information in the process is shown below the diagonal, and non-zero upper-diagonal cells indicate the presence of feedback.¹ A mark above the diagonal is an indication of information flowing from a downstream activity to an upstream activity.

To reduce the feedback marks — thereby design process complexity — a simple sorting procedure is performed [43]. The procedure, also known as partitioning, consists of rearranging rows and columns (i.e. activities) in a way to force all marks below the diagonal without augmenting any of the existing relationships. An ideal sequence is defined as one with no feedback marks, represented by a lower-triangular DSM. In real and complex product development processes, this ideal sequence is unlikely to exist [43]. Then, design process optimization becomes a problem of finding the sequence of design activities that minimizes feedback.

4.3. Step 3: build an algorithm to convert dependency information into a schedule (Fig. 2c)

The objective of the proposed DSM-based algorithm is to find the shortest possible (overlapped) design schedule by processing the dependency information gathered in Step 1 and represented by the DSM built in Step 2. As illustrated in Fig. 5, after initialization, the algorithm proceeds in two main stages. In the first stage, feed-forward information (i.e., information flowing from upstream activities to downstream activities) is analyzed to find the earliest start time for each dependent (i.e. downstream) discipline. This will constitute the preliminary fast-tracked project schedule. In the second stage, the preliminary fast-track schedule is confirmed, or adjusted, by examining the impact of feedback information (i.e., information flowing from downstream activities to upstream activities). This second phase involves checking whether any requisite feedback information would have enough time to be incorporated by the upstream activity. We require that the upstream activity be ongoing at the time when feedback arrives. If this requirement is not met, the upstream activity’s earliest end time is adjusted (i.e. delayed) to accommodate this requirement. An adjustment to at least one of the earliest end times requires that all calculations in phase 1 of the algorithm be repeated to ensure information flow consistency. The algorithm terminates when this requirement is ensured for all activities. The main output of the algorithm is a fast-track schedule with the earliest possible start time for each design activity. Figs. A1 and A2 in the appendix summarize the formulated algorithm and its input variables. To ease computations, the algorithm was coded in Microsoft Excel Visual Basic.

The following assumptions were made to ensure proper implementation of the algorithm:

- To meet the project’s schedule, work on an activity starts as soon as some information is received from the other dependent activities.
- To make sure that the additional information is received while the activity is still ongoing, we built into the activity duration computation a condition ensuring that the duration is long enough to allow for all information to be received prior to releasing the deliverable.
- The ARC design discipline starts first reflecting the fact that ARC is typically the lead discipline.
- Downstream interdependent activities give information to the upstream design discipline as a whole and the specific activities receiving the information are not identified. Identifying the specific receivers of such information would require daily observation of information exchange, which was beyond the scope of this study.
- All design disciplines give information to the first activity in the dependent design disciplines.
- The time needed to complete each deliverable is determined by the weight or percentage allocated to this deliverable in the corresponding design discipline. This information can be obtained from estimates of man-hours spent, or number of drawings generated, per design deliverable.
- Activities within a design discipline are strictly sequential.

The algorithm is generic by nature and can be applied to the design phase of any construction project. To ensure that it is error free, the algorithm was tested and verified on several small-scale examples illustrating possible scheduling scenarios. The examples

---

¹ Some DSM literature uses the opposite convention (the transpose of the matrix), with inputs in columns and outputs in rows, and thus feedback below the diagonal; the two conventions convey equivalent information.

| Table 1 | Information exchange between pairs of design activities (feed forward). |
| --- | --- | --- | --- | --- | --- |
| **ARC** | **Weight** | **FAC** | **SBS** | **SPS** | **MEP** | **FIN** |
| AR₁, Architecture layout plans, sections, and elevations | w₁ | d<sub>AR₁-FAC</sub> | - | - | - | - |
| AR₂, - | w₆ | - | - | - | - | - |
| Total percentage | 100 | - | - | - | - | - |
| FAC | **Weight** | **ARC** | **SBS** | **SPS** | **MEP** | **FIN** |
| FA₁ | - | - | - | - | - | - |
included small DSMs (e.g., $2 \times 2$, $3 \times 3$, $4 \times 4$), scenarios with fully sequential activities, and scenarios with full overlap among activities. For each of these examples, the solution provided by the algorithm was compared with the solution obtained manually.

4.4. Step 4: derive a bar chart schedule (Fig. 2d)

The next step after running the algorithm is to use its output to derive a bar chart schedule showing the earliest start and completion times of each of the activities in the design period of the design phase taking into account dependency information. This information is useful to the project manager and to the group leaders who are in charge of staffing each of the major design disciplines.

5. Validation of the framework through a case study

As mentioned earlier, this paper proposes a four-step DSM-based framework to quantify the dependencies among design activities. The DSM, in turn, serves as the basis for scheduling project design activities in the shortest possible manner, thereby ensuring the execution of a fast-track project. In this section, data obtained from a real-world case study project is used to illustrate the implementation of each of the steps of the proposed framework. The collected data was the result of several interviews held with senior (or lead) engineers and project managers representing key engineering disciplines working on the case study project. The interviews were semi-structured. In other words, there were no fixed questions. Instead the interviewer used a general list of topics to guide the interview. The questions revolved around the nature and timing of information exchanged by each pair of design activities or disciplines.

The purpose of the case study project is to design and build a multi-billion dollar educational facility that will provide educational services and accommodation for about 40,000 full-time graduate and undergraduate students and about 30,000 teaching and administrative staff. The project is located in the Arabian Peninsula and has a planned built-up area of 2.8 million m² and a total site area of 800 ha.

The case study project is fast-tracked by nature. Given the project’s large scale and significant impact on the country’s economy and image, the client expressed a clear objective to have the educational facilities operational by the beginning of 2011. As such, the selected design and contracting firms, which started working on the project early 2009, had only two years to complete design and construction. Therefore, they planned to execute as much work in parallel as possible. Construction started prior to the full completion of project design.

The information gathered pertaining to the case study was essential in illustrating the previously described four-step framework for automatically generating a fast-track design schedule. The following sections elaborate on the details of each step of the proposed framework.

5.1. Step 1: quantifying design dependencies

As previously mentioned, the first step in the proposed framework is to quantify the types, and level, of information exchanged among design activities. For this purpose, interviews were held with the group leaders of all disciplines working on the case study project. Each interviewee gave his or her vision of the type, and amount, of information delivered to the other disciplines during the construction documentation phase of the project design. The selection of the construction documentation phase in this study for quantifying dependency was due to its relative importance. This phase lasted 6.5 months during which a significant amount of information was exchanged among design disciplines. The two previous phases, schematic design and design development, lasted three months.

Overall, the interviewees provided the following information:

- A breakdown of deliverables in each design discipline along with the corresponding percentage (or weight) reflecting the relative amount of work needed to complete the deliverable. For example, as shown in Table 2, the ARC discipline includes three deliverables: $AR_1$ (layout, plans, sections, and elevations) with a weight of 70%, $AR_2$ (stairs details and circulation cores) with a weight of 20%, and $AR_3$ (fire zoning) with a weight of 10%. In other words, according
to the architecture group leader, AR1 consumes approximately 70% of the time allocated for ARC, whereas AR2 and AR3 consume 20% and 10%, respectively. Each deliverable is represented by two letters of the design discipline name: AR for ARC, ME for MEP, SB for SBS, SP for SPS, FA for FAC, and FN for FIN. The deliverables are listed according to the sequence of design in each design discipline, i.e. AR1 is typically designed first, followed by AR2 and AR3.

- The percentage of work done on each design discipline deliverable before information is released to the other dependent design discipline. For example, FIN needs approximately 65% of AR1, 50% of AR2, and 80% of AR3 before it can start.

To come up with these percentages, the interviewer relied on the process depicted in Fig. 3 and the interviewees relied on their previous experience with regard to the amount of work done in a certain deliverable before information is shared with other disciplines. For example, the layout plans in ARC consist of walls, doors, windows, room tags, axes, dimensions, references, etc., whereas the information shared with the MEP group during the construction documentation phase is limited to the doors and windows only, which constitute about 15% of the work.

The information presented in Table 2 represents the percentages required from each design discipline in order for the other design disciplines to start the construction documentation. During this phase, another level of coordination takes place where information is shared in the opposite direction, i.e. from a typically “receiver” activity to a typically “sender” activity. For example, the FAC discipline

Table 2
Information exchange between pairs of design activities (feed forward).

<table>
<thead>
<tr>
<th>Design Discipline</th>
<th>ARC</th>
<th>FAC</th>
<th>SBS</th>
<th>SPS</th>
<th>MEP</th>
<th>FIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR1 (Architecture)</td>
<td>70%</td>
<td>15%</td>
<td>20%</td>
<td>20%</td>
<td>15%</td>
<td>65%</td>
</tr>
<tr>
<td>AR2 (Stairs details)</td>
<td>20%</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>AR3 (Fire zoning)</td>
<td>10%</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>80%</td>
<td></td>
</tr>
<tr>
<td>Total Frequency</td>
<td>100%</td>
<td>10%</td>
<td>15%</td>
<td>15%</td>
<td>10%</td>
<td>65%</td>
</tr>
</tbody>
</table>

Table 3
DSM.

<table>
<thead>
<tr>
<th>Design Discipline</th>
<th>ARC</th>
<th>FAC</th>
<th>SBS</th>
<th>SPS</th>
<th>MEP</th>
<th>FIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SBS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4
Simplified DSM.

<table>
<thead>
<tr>
<th>Design Discipline</th>
<th>ARC</th>
<th>FAC</th>
<th>FIN</th>
<th>MEP</th>
<th>SPS</th>
<th>SBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARC</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SBS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MEP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FIN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

...
Table 5
DSM for the case-study project.

<table>
<thead>
<tr>
<th></th>
<th>AR1</th>
<th>AR2</th>
<th>AR3</th>
<th>FA1</th>
<th>FA2</th>
<th>FA3</th>
<th>FA4</th>
<th>FA5</th>
<th>FN1</th>
<th>FN2</th>
<th>FN3</th>
<th>FN4</th>
<th>FN5</th>
<th>FN6</th>
<th>FN7</th>
<th>FN8</th>
<th>FN9</th>
<th>ME</th>
<th>ME</th>
<th>ME</th>
<th>ME</th>
<th>SP1</th>
<th>SP2</th>
<th>SP3</th>
<th>SP4</th>
<th>SP5</th>
<th>SB1</th>
<th>SB2</th>
<th>SB3</th>
<th>SB4</th>
<th>SB5</th>
<th>SB6</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR1</td>
<td>4.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.1</td>
<td>0.2</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AR2</td>
<td>1</td>
<td>1.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AR3</td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA1</td>
<td>0.15</td>
<td></td>
<td></td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA2</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN1</td>
<td></td>
<td></td>
<td></td>
<td>0.15</td>
<td>0.5</td>
<td>0.8</td>
<td>0.5</td>
<td>0.4</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FN7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ME</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.1</td>
<td>0.35</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SP3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB1</td>
<td>0.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SB6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I.M. Srour et al. / Automation in Construction 28 (2013) 1–11
5.2. Step 2: converting dependency information into a DSM

The next step in the process consists of converting the dependency information illustrated, for the case study, in Table 2 to a DSM format. Table 3 shows basic dependency information for the case study at the level of design disciplines.

The DSM shown in Table 3 includes seven (x) marks above the diagonal representing feedback information. To reduce these feedback marks — thereby reducing design process complexity — a simple partitioning or sorting procedure is performed [42]. The procedure consists of rearranging rows and columns (i.e. activities) in a way to force all marks below the diagonal without augmenting any of the existing relationships. Table 4 illustrates a simplified DSM after rearranging some of the activities; the marks above the diagonal indicate interdependent activities.

Taking the analysis a step further, the percentages presented in Table 2 were added to the simplified DSM. The idea is to represent the percentages and durations of activities, and to quantify the exchange of information between pairs of activities. Table 5 shows the updated DSM including the percentages of dependencies as well as activity durations, shown in the diagonal. For example, the duration of the FAC discipline is 2 months which is the sum of the durations needed to complete each of the deliverables. The DSM representation is such that tasks give information to the tasks in the rows and requires information from the tasks in the columns. For example, the first column reads that the ARC design discipline gives information to all other design disciplines (e.g., FAC needs 15% of AR1 work before it can start). The first row reads that ARC requires information from three design disciplines: FAC, FIN, and MEP. Examples of independent disciplines include SBS and FIN; examples of dependent disciplines include SPS and ARC; and examples of interdependent disciplines include ARC and FAC.

5.3. Steps 3 and 4: apply algorithm and derive design schedule

The next step after generating the DSM is to apply the algorithm which converts the information into a schedule of overlapped design activities for the case study project. The algorithm run time is in the order of a few seconds. Fig. 6 (blue activities) shows the bar chart schedule derived from the DSM for the case study project. Each of the arrows within a design discipline indicates a deliverable. For example, MEP has four arrows referring to the time needed to complete each of the four deliverables: ME1, ME2, ME3, and ME4.

To validate our proposed DSM-based schedule, we overlaid the actual schedule used on the case study project (i.e. red activities in Fig. 6) on top of our proposed schedule. Discussions with the design group leaders indicated that the actual schedule was obtained using an informal process, i.e. through trial and error. Start and end dates for design activities were calculated using several pieces of information including: final design delivery dates, engineering production rates, and information exchange requirements among design activities.

To compare the two schedules, the following observations were made at the level of design discipline:

- The ARC design discipline has the same start and finish dates in both schedules due to the assumption that ARC starts at \( t = 0 \).
- There is a minimal difference between the used and the DSM-based schedules for SPS, MEP, and FIN.
- The actual schedule indicates that the SBS design discipline starts two months earlier than in the DSM-based schedule. When asked for the reason for starting the SBS discipline early in the design process, the design group leader in charge highlighted the criticality of this discipline since it is the first activity in construction.
- Finally, the actual schedule indicates that the FAC discipline starts four months after the beginning of the ARC design discipline, whereas in the DSM-based schedule, FAC can start at 0.75 months. Again, as confirmed by the design group leader in charge, the reason for not starting the FAC design discipline earlier was due to construction priorities. Ideally, this type of information should be incorporated in the dependency, and therefore DSM, analysis.

---

2 The interviewees could not point out to the actual deliverable (e.g., AR1, AR2, or AR3) that receives the feedback information. As such, the DSM representation indicates feedback information flowing to the entire discipline (ARC).
6. Concluding remarks and recommendations

Given the pressure from the business world to execute projects in the fastest possible manner, fast-tracking is gaining popularity in the industry. This, in turn, requires not only overlapping design and construction activities but also a two-way exchange of information among design disciplines. Standard scheduling techniques such as the critical path method allow for only a one-way exchange of information and therefore cannot be used to schedule this type of activities. This paper addressed this problem by providing a methodology for using bi-directional exchange of information to schedule a fast-track design phase. Previous efforts [30–34] focused on using the DSM tool to sequence design activities. The novelty in this paper lies in the proposed methodology for not only sequencing but also quantifying the amount of overlap between pairs of design disciplines at a detailed level. The methodology consists of a multi-step framework where the first step is to break down design activities into deliverables within each discipline and quantify the type, amount, and timing of information exchanged between pairs of design activities. The second step is to translate the information into a DSM. The third step is to use an algorithm to convert the information into a schedule. The fourth and last step is to derive a bar chart schedule showing the earliest possible start for each design deliverable, taking into account dependency information.

In addition to allowing for a two-way exchange of information, the proposed framework allows for the exchange of partial information. In other words, unlike traditional scheduling techniques which allow for only a complete exchange of information between pairs of activities, the proposed DSM-based approach allows for modeling situations where incomplete information is exchanged among design activities, which is the norm in a fast-track environment.

The proposed framework provides a formalized approach for capturing, and using the experience of practitioners, to schedule the design phase of construction projects. As illustrated by the case study, the proposed framework can serve as a basis for scheduling the design phase of fast-track projects, especially megaprojects which are complex by nature [44].

The proposed framework has several limitations. First, it is limited to the design phase of construction projects. Future work can expand the idea of representing dependency information in a matrix format to include both design and construction activities. Second, the framework does not take into account resource considerations. Starting an activity at the earliest possible time and maintaining its “on-going” status until all information is exchanged might result in resource inefficiencies. An improved DSM-based framework should not only provide the shortest possible schedule but also evaluate this schedule vis-à-vis resource availability.

Future work in this area should examine the feasibility of integrating the proposed DSM-based framework into commercially available scheduling programs (e.g., Primavera, MS Project) or advanced simulation tools (e.g., MicroCyclone, EZStrobe [45]). Future work can also elaborate on the factors ensuring a successful fast-track strategy. An example would be the role of proper interface management in ensuring an effective execution process — whether it is the interface between the design firm and contractors or interface among contractors. Finally, we recommend incorporating sensitivity information into the proposed DSM-based scheduling framework, thereby expanding Eppinger’s [21], Bogus et al.’s [19], Pena-Mora and Li’s [20], and Blacud et al.’s [1] work. How sensitive is the information needed by downstream activities to changes in the upstream activities? How costly will these changes be? What strategies can be used to minimize the risk of costly rework?

Acknowledgments

The authors would like to thank the American University of Beirut’s University Review Board for funding this study.

**Appendix A. Input variables and pseudo code for the proposed DSM-based overlapping algorithm**

| $K$: Index of discipline; $1 \leq k \leq n$ |
| $i$: Index of activity in discipline $k; 1 \leq i \leq m_k$ |
| $v$: Index of previous discipline |
| $j$: Index of activity in previous discipline |
| $n$: Number of disciplines |
| $m_k$: Number of activities in discipline $k$ |
| $AD_{ak}$: A matrix of $(k \times i)$ containing the activities’ duration |
| $PD_r$: Discipline duration vector, a vector of length $n$ and its variables are calculated from activity duration inputs ($PD_r = AD_{ria} + AD_{rj} + \cdots + AD_{rj} + \cdots + AD_{rkn}$) |
| $PW_{k,ij}$: Input matrix of the percent work needed for a certain activity to give information to a discipline. |
| $PW_{k,ij}$: Input matrix of the percent work needed for a certain downstream activity to give information to an upstream discipline. |
| $RSTP_{k,v}$: Calculated earliest start time of a discipline relative to an upstream dependent activity. |
| $RETP_{k,v}$: Calculated earliest end time of a discipline relative to an upstream dependent activity. |
| $STP_v$: Calculated earliest start time of a discipline: A vector of length $n$ |
| $ETP_v$: Calculated earliest end time of a discipline: A vector of length $n$ |

**Fig. A1. Summary of model input variables.**
1. Initialize, schedule discipline 1:
   a. \( STP_1 = 0 \)
   b. \( ETP_1 = STP_1 + PD_1 \)

2. Start with discipline \( k = 2 \)

3. Calculate earliest schedule for discipline \( k \) relative to its predecessor disciplines:
   a. For each upstream activity \( j \) in previous discipline \( v \):
      i. Calculate: \( RSTP_{kj,v} = STP_v + (AD_{v,k} + AD_{v,k+1} + \ldots + AD_{v,k+j}) + PW_{v,k} \times AD_{v,k} \)
         Calculate: \( RETP_{kj,v} = RSTP_{kj,v} + PD_k \)
      ii. Record Maximum Start Time: \( MST = \max (MST, RSTP_{kj,v}) \)

   b. Choose earliest start time for discipline \( k \):
      Choose earliest start time of discipline \( k \) from \( RSTP_{kj,v} \) for minimum positive difference between its end time and \( MST \):
      \( STP_k = RSTP_{kj,v} \) given \( V - v \) and \( J = j \) for minimum positive \( \Delta_j \) \( (RSTP_{kj,v} - MST) \)
      \( ETP_k = RETP_{kj,v} \)

4. Check validity of schedule discipline \( k \) relative to feedback:
   For each activity \( i \) in discipline \( k \):
   i. Calculate start time as temporary value: \( tmpST = STP_i + (AD_{i,k} + AD_{i,k+1} + \ldots + AD_{i,k+i}) \)
   ii. Check for each previous discipline \( v \):
      i. Calculate difference in time between end time of previous discipline and arrival of feedback: \( tmpDiff = STP_v - (tmpST + PW_{i,k} \times AD_{i,k}) \)
      2. If \( tmpDiff > 0 \) then:
         a. Schedule is OK
         b. \( k = k + 1 \)
      3. If \( tmpDiff \leq 0 \) then:
         a. Set \( MST = tmpST + PW_{i,k} \times AD_{i,k} \)
         b. Choose optimal start time for discipline \( v \):
            Choose optimal start time of discipline \( v \) from \( RSTP_{kj,v} \) (discipline \( i \) is a previous discipline of \( v \)) for minimum positive difference between its end time and \( MST \):
            \( STP_v = RSTP_{kj,v} \) given \( U = u \) and \( J = j \) for minimum positive \( \Delta_j \) \( (RSTP_{kj,v} - MST) \)
            \( ETP_v = RETP_{kj,v} \)
         c. Set \( k = k + 1 \)

5. If \( k > n \) then finish Algorithm else return to step 3

---

References


