Scientia Iranica E (2019) 26(4), 2506–2523

Sharif University of Technology
Scientia Iranica
Transactions E: Industrial Engineering
http://scientiairanica.sharif.edu

A new heuristic method for improved structuring of the Work Transformation Matrix (WTM)

M. Haji Jafari, A. Kosari*, and M. Fakoor

Aerospace group, Faculty of New Sciences and Technologies (FNST), University of Tehran, North Kargar Street, Tehran, Iran.

Received 23 September 2017; received in revised form 10 February 2018; accepted 9 June 2018

KEYWORDS
Coupled Block of Activities (CBA); Tearing; Partitioning; Work Transformation Matrix (WTM); Discrete-Time Simulation (DTS).

Abstract. The Design Structure Matrix (DSM) can be used as a potent tool in the management of product design processes. Although the compactness and ability to represent design cycles are the main advantages of DSMs over the existing traditional tools, the intact whole DSM is not always an understandable piece of information. To overcome this shortcoming, certain analyses have been proposed for a better understanding of the matrix in which partitioning and tearing are of significant importance. There are several algorithms for these two analyses that mainly focus on a few rules of thumb. Although partitioning and tearing were originally developed for binary DSMs, they can be extended to numerical variants in which the Work Transformation Matrix (WTM) is of the highest fame and application. In this paper, the authors have proposed an algorithm inspired by the formation of sugar crystals in saturated syrup for reordering the activities in a Coupled Block of Activities (CBA) based on their level of coupling. To implement this, a code was developed to achieve pseudo-optimum solutions. By using a discrete-time simulation, which was applied to an aerospace case study, it was demonstrated that the method produced restructured schemes of the WTM comparable/superior to the classical methods.

1. Introduction

With the emergence of the Design Structure Matrix (DSM) as a potent tool in design process modeling, a window opened for new opportunities. A plethora of various applications and hybridizations have been proposed since that point. While these methods are diverse in appearance, they all insist on a single pivotal goal: "to minimize the total completion time of the processes with respect to cost and quality constraints".

Although a proper and precise definition of the design activities is a trivial requirement, no design process is complete without defining the structure of information flow among its constituent activities, or in technical terms, its process architecture.

As complexity is an inherent property of modern engineering systems, their design processes are characterized as "highly coupled". In the matricial form, it is interpreted into Coupled Blocks of Activities (CBAs) in DSMs, or their numerical forms, e.g., Work Transformation Matrices (WTMs). As a result, converting a CBA into an understandable and manageable piece of information involves considerable effort.

In this paper, the authors aim to introduce a new heuristic method for restructuring the CBAs in a WTM. To do this, a heuristic algorithm (named Nabat, which means sugar crystals in Persian) has been proposed that mimics the gradual growth of crystals in saturated sugar syrup. This algorithm combines the acts of tearing/partitioning in a gradual down-up manner in order to reorder and manage activities. As an accom-

*. *Corresponding author. Tel./Fax: +98 21 61118495*
E-mail addresses: m_hajijafari@ut.ac.ir (M. Haji Jafari); kosari_a@ut.ac.ir (A. Kosari); mfakoor@ut.ac.ir (M. Fakoor)

doi: 10.24200/sci.2018.5248.1168

paniment, an execution plan has been produced while developing the whole WTM from smaller sub-blocks.

2. Literature review

New market opportunities come about in a very competitive environment. The time taken to introduce a new product in the market is one of the main factors besides design costs and risks during development [1]. Although several complex systems have been developed in response to increasingly sophisticated needs of operation, bringing them to reality is becoming more difficult.

Whereas most people try to understand the complexity of a system in terms of its physical features, Baccarini [2] proposed that complexity of a project should be defined as consisting of many varied interrelated parts and operationalized in terms of its differentiation and interdependency. Meanwhile, Suh defines complexity as a measure of the uncertainty involved in satisfying the functional requirements [3]. For systems with a higher number of elements, satisfying the functional requirements becomes more difficult, which is termed as a higher complexity. As a result, it can be said that complex systems tend to have a large number of entities accompanied by intermingled relations and repetitive transmission of information among them. In other words, the difficulties involved in designing complex engineering products are significantly affected by their managerial complexity [4]. In this respect, many aerospace products are regarded as complex and, from a design process point of view, they include cyclic processes [5].

The CBAs are the main sources of rework in design processes, which may lead to unacceptable levels of lead time, cost, and risk [1]. The main approach to controlling the process is to reorder and restructure the activities of a process in order to have blocks of activities that are smaller in dimension with reasonable orders of execution [6]. While the former is mainly realized in the partitioning of the WTM, the latter can be done by proper tearing or organizing the activities inside a coupled block to reduce the number and severity of rework feedbacks.

Although an exotic variety of tools have been devised and proposed for managing the design process of complex systems, each tool has certain liabilities in some aspects. The critical path method as the most prevalent digraph model has been used to represent the relationships between the activities in a design process [7]. Meanwhile, the PERT diagrams have been developed to overcome the deterministic nature of the CPM [8]. The digraph methods, generally, are ill-suited for showing the cyclic nature of the processes [9].

A rather recent and continually evolving method of modeling the process structure is the Design Structure Matrix (DSM), which encapsulates the relations between the components of a system in a square matrix with dimension n (the number of components). In this method, each X mark indicates the existence of relationships between two pertaining components, whereas no mark indicates that it is void [1]. This study follows a counterclockwise information flow perspective, which relates the lower diagonal marks to the feedforward flow and the upper diagonal marks to the feedbacks (Figure 1). The DSMs are understandable regardless of the complexity with respect to size [4]; therefore, they are useful in concurrent engineering management and implementation. DSM methods are becoming increasingly mainstream, especially in the areas of engineering design, engineering management, organization science, and systems engineering [5].

Although the design structure matrix introduces a new range of capabilities, its functionality is restricted to the binary domain. This means that DSMs are useful only in indicating the existence and void of relations; however, their strength is not represented. To overcome this shortcoming, an augmented version of the DSM has been introduced in which X marks are replaced by the probabilities of the transmission of information between each pair of activities. This matrix is called the Work Transformation Matrix (WTM), which describes the work produced and done for each component of the system [10].

The structural model of a DSM usually begins with partitioning, in which the matrix is reshaped into an upper diagonal form by reordering its components. However, due to coupling, or cyclic relations between the components, this is not always possible. In this case, the portioning aims to minimize the size of the coupled blocks and place them as downstream as possible.

There are various procedures and algorithms by which a set of activities can be reordered. Steward suggests swapping the rows and columns to find the CBAs [11]. When the matrix is in the lower triangular form to the maximum extent, the CBAs are situated along the main diagonal. Horowitz et al. [12] proposed a new algorithm known as topological sorting using

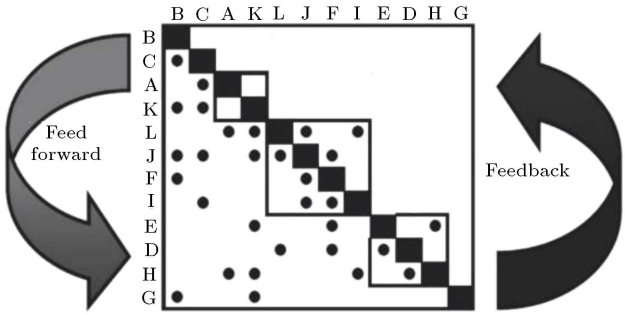


Figure 1. Information flow in a typical DSM [31].

adjacency lists. Lawler [13] developed a simple and efficient procedure to find a topological order of activities using the matrix form when no cycles exist. This algorithm can detect the cycles, but cannot predict when one will be found. Kingston's proposition is based on deleting the activities that have no inputs up to the unsolvable couplings [14].

Most of the recent studies are based on heuristic agents. For example, Meier et al. applied the genetic algorithm to the information flow model with various types of arrangement in order to find the best orders [15]. Sen et al. related uncertainty with the entropy of the system process and tried to reduce it in the processes where the solution was refined. They defined coupling as the reduction of the variable search space, which acts as a guide to the sequence of tasks for the fastest possible design convergence [16]. Meanwhile, Tsai et al. proposed an Improved Differential Evolution Algorithm (IDEA) based on the proposed cost and time models for optimizing task scheduling and resource allocation [17].

The majority of the partitioning algorithms developed up to now are in similar forms to each other. The main difference among them is the method of identification of the cycles and, once the cycle has been identified, the group of activities in the CBA is collapsed into one representative node. Some generic partitioning procedures are described in [18-20].

Partitioning is generally followed by tearing. The starting point of a coupled block, which is not a single well-defined point, involves a chicken or egg type of dilemma. In order to obtain component B, information from component A is needed, whereas A is determined from the information produced from B. For solving this problem, a starting point must be chosen, i.e., either A or B is better to be solved first. In such a case, an assumption must be made regarding the starting activity instead of the information that is to be provided by the other; this involves breaking an information cycle.

Contrary to partitioning, which is generally the same for binary and numerical matrices, there are several plans for tearing a numerical DSM (e.g., WTM) with respect to the information further provided. It means that a richer DSM allows the development of more practical and efficient tearing algorithms. As a result, the choice of the structural model and the tearing criterion affects the process significantly.

No optimal method exists for tearing [21]; therefore, any method that insists on some aspects may be weak in others. Kusiak and Wang proposed two main rules of thumb for the tearing procedures [22]:

- Minimal number of tears: the less number of connections is torn, lesser information is lost and, thereby, lesser guesses are made;
- Confining tears into smaller CBAs: The nearer the connections are to the main diagonal, the lower the cost of guesses done for them.

They used the frequency of occurrence for each edge-disjoint cycle in a block to tear the edges with the highest frequency and, then, repartitioned the matrix. Eppinger et al. addressed the strength of dependency among the activities involved in the tearing process for the first time [23]. They used the likelihood of repeating the activity if it proceeds without a particular required input.

Nearly all studies that are based on measuring the strength of a coupling between the activities were aimed at improving the tearing analysis. One of the most notable attempts in tearing the numerical DSMs was made by Yassine et al. [4]. They used two variables, sensitivity and variability, to develop an index for improving the tearing. Zhang et al. used the gross workload of iteration for different sequences of coupled activities to measure the strength of the coupling between tasks [24]. Su et al. used the Analytic Hierarchic Process (AHP) to measure the strengths of the coupled design tasks. They chose to ignore the weaker couplings and, finally, proposed an algorithm for the best processing sequence of the coupled activities [25]. In another work, Xu et al. used the AHP based on a triangular fuzzy number to analyze the relations between the sources and activities through a degree of cross-relations. This helped them to introduce a method for tearing coupled activities in a concurrent design [26].

Although tearing is usually considered as omitting a number of feedbacks in order to make the process as straightforward as possible, some researchers have used it to break a large CBA into some smaller and more manageable blocks. The idea of controlling the features in [10] helped SoltanMohammad and Malaek [27] break the main CBA within their WTMs into two smaller blocks, in which the upstream block plays the role of the most determining/controlling features. In another research, Bashir et al. proposed a quantitative approach and grouping index to break the large block of activities into smaller CBAs [28]. Finally, Xiao et al. [29] decided to exclude some less significant activities from the coupled block of activities and took certain measures to compensate for the lost data.

As stated earlier, the authors aimed to develop a novel method for the structural analysis of the WTM based on a gradual growth scheme, which means allowing the activities of a process to engage in a gradual manner. This method is a novelty due to its following features:

- The CBAs are grown gradually, wherein they are formed from the constituent activities in several steps. Here, the starting points of the CBAs (cores)

are the activities with higher mutual couplings. The activities are absorbed into these cores to grow the CBAs to see whether the possible CBAs merge to form larger ones;

- No bond (information flow) is permanently broken; the less important ones are neglected until they play a role as an input/output in the growing coupled block;
- The order of tasks is adjusted within each CBA. It can be regarded as an inter-CBA partitioning.

The remainder of the paper is structured below. Section 3 is dedicated to the introduction of the method. Here, the Nabat algorithm has been described using a flowchart and a simple example. In Section 4, the case study is introduced. Section 5 includes the results of applying the Nabat algorithm to the Fajr F.3 GA aircraft (the case study) and its pseudo-optimum solutions. Section 6 is devoted to the discussion on the results of the discrete-time simulation, showing the applicability of the method. The conclusion and scope for future works are presented in Section 7, and the final section is dedicated to references.

3. Method

In this paper, the authors aimed to introduce a method for improving the partitioning-tearing of a CBA. In order to tackle this problem, a heuristic method has been proposed. Nabat (meaning sugar crystal in Persian) has been introduced as an algorithm to solve the problem of coupled blocks. This algorithm is based on the gradual incarnation of the CBAs by mimicking the gradual growth observed in the formation of sugar crystals. However, for further understanding, some additional explanations seem to be required.

3.1. Eigenstructure analysis

As mentioned earlier, there are several methods and indices for measuring the strength of the coupling between the activities within a CBA. In the present study, the authors decided to use the Eigenstructure analysis as the measure [10]. In this regard, the largest element of the eigenvector of each CBA (presented as a WTM) was accepted to indicate the strength of its coupling. In a similar manner, the eigenvector of a WTM entails the contribution of each Eigenvalue to the body of the work.

In the proposed method, if the largest element of the eigenvector of the proposed activities was larger than a unit random value, the subsequent creation of the sub-block would be allowed. A more detailed description of the algorithm is presented in the following sections.

3.2. Time-based versus static DSMs

Static DSMs describe the constituent elements of a system, all of which are present at the beginning. According to Browning, physical and organizational DSMs are examples of this type of DSM [6]. On the other hand, parametric and activity-based DSMs evolve with time, which means that all of their components are not present at the beginning and are introduced as and when required. An exemplary matrix of such type is presented in Figure 2.

As explained above, it is possible to form a coupled block of activities gradually in order to develop a better plan of execution, e.g. how to introduce, do, and redo a set of activities. According to Browning et al. [30], it is important leverage to control the process in order to have a shorter time of execution for lesser costs.

3.3. Nabat algorithm

The Nabat algorithm employs the gradual formation of a CBA to find an improved execution scheme for the activities. To find a CBA, one can use an eigenstructure of the nominated groups of activities as a guide, i.e., a group of activities is selected, and their eigenstructure is calculated as a numerical matrix. The largest non-zero component of the corresponding eigenvector indicates a cyclic information flow, or technically speaking, a coupling. Kosari et al. demonstrated that, in a CBA, the upstream and downstream activities are mutually controllable/observable [31]. An introductory graphical view of the Nabat algorithm is illustrated in Figure 3.

Figure 3 entails the basic actions involved in the Nabat algorithm, including shuffling the columns, setting/revising the size of groups, checking the coupling, possible merging of the CBAs, checking and setting the internal order of activities within the CBAs, and performing a similarity check as the termination check. From here onwards, the words "column" and "crystal" indicate the same thing, i.e., a CBA. Whereas the former is used with respect to the physical origin of the algorithm, the latter indicates its mathematical resemblance. These actions are defined as below:

- Shuffling is the random reordering of columns (CBAs or singular activities) in order to find the possible couplings. Here, only (a group of) the adjacent columns are checked for couplings;

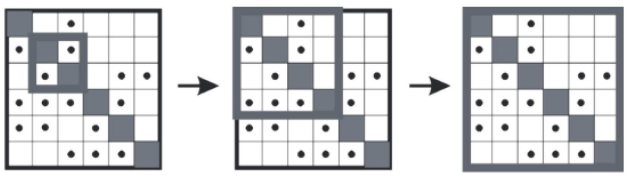


Figure 2. Gradual growth of a DSM.

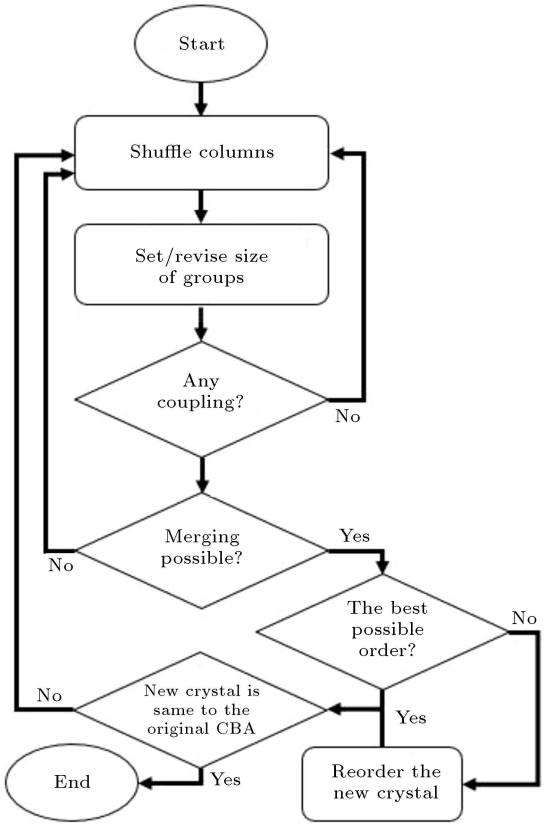


Figure 3. Nabat algorithm flowchart.

- Setting/revising the size of the groups indicates the number of adjacent columns according to which their coupling is measured ($2 \leq m \leq n$, where m is setting/revising the size of the groups and n is size of the whole DSM);
- As stated before, the eigenstructure analysis is used to check a coupling. For the largest non-zero component of the eigenvector, a group of columns is regarded as coupled and, then, merged when it surpasses a randomly set threshold;
- Two or more columns (i.e., CBAs and singular activities) may be merged when they are coupled according to the criterion mentioned above. This action is responsible for "growth";
- Whenever a group of activities is to be merged, an internal order must be decided, which means determining the order of precedence of the columns within the CBA. Here, due to the rather small number of columns (determined from the aforementioned size of groups m and controlled to be manageable), all possible permutations are checked;
- The similarity check terminates the algorithm when the CBA is grown to form the complete WTM, i.e., a column that contains all the activities.

The following example facilitates a better understanding:

- **Start.** A pool of all the activities in single columns (scalars) was created. For a set of five exemplary activities, we have [1][2][3][4][5];
- **Shuffle columns.** The columns (here, the singular activities at the beginning) were randomly shuffled to find the possible chances of coupling. Note the following example for the five activities: [1][2][3][4][5] may be shuffled to [4][1][2][5][3] or $\begin{bmatrix}1\\3\end{bmatrix}$ [2][5][4] to [2] $\begin{bmatrix}1\\3\end{bmatrix}$ [4][5];
- **Set/revise size of groups.** A number of columns must be combined to form new crystals. This number evidently determines the process of crystallization. The algorithm begins each cycle with a randomly (but not evenly) chosen number for the size of the groups and tries to increase it when it fails to find a coupling after several attempts. The recursive function that is used in this case and tuned to reach this aim is:

$$k_1 = 2,$$

$$k_{i+1} = \min\left(k_i + \lfloor 0.05 + \text{rand}^{k_i} \rfloor, L_a\right), \qquad (1)$$

where *rand* is a random unit number, and L_a is the number of the present columns in each step;

- **Check couplings.** As stated before, when the eigenvector of the matrix pertaining to the columns directed to form a new crystal has a non-zero value, then those columns are coupled in some form. When this occurs, there is a chance for the crystal to form. Otherwise, even though the group of columns is coupled in some manner, they are not allowed to form a larger crystal. This is helpful in finding stronger couplings and, therefore, finding better execution plans;
- **Merge columns whenever possible.** Although a number of columns have the chance to form/grow a crystal, not all will always be successful. To encourage the formation of stronger bonds, a threshold was defined based on the level of coupling and the size of the groups. If no new chances appear, the threshold is lowered gradually until one is found. For example, in case there is no new crystal formed from the groups of 4 columns after several attempts, the algorithm tries to find crystals from the groups of 3 columns; if it is unsuccessful again, it will try with groups of 2 columns;
- **Check the best possible order in the crystal.** The following example illuminates this problem. Let us assume that for the example with five activities mentioned above, the WTM has the following initial structure:

$$\begin{bmatrix} \#1 & 0.1 & 0.2 & 0 & 0 \\ 0.2 & \#2 & 0.1 & 0 & 0.1 \\ 0.3 & 0.4 & \#3 & 0.4 & 0 \\ 0 & 0.6 & 0.1 & \#4 & 0 \\ 0.8 & 0.2 & 0.3 & 0 & \#5 \end{bmatrix}.$$

While trying to form/grow crystals between $\begin{bmatrix}1\\3\end{bmatrix}$ and [2] in structure $\begin{bmatrix}1\\3\end{bmatrix}$ [2][5][4], there are two possibilities: 1) put $\begin{bmatrix}1 & 3\end{bmatrix}^T$ ahead and add [2] after it, or 2) do the reverse. For each case, the coupled sub-block containing elements [1], [2] and [3] is as follows:

$$\begin{bmatrix} \begin{bmatrix} \#1 & 0.2 \\ 0.3 & \#3 \end{bmatrix} & \begin{bmatrix} 0.1 \\ 0.4 \end{bmatrix} \\ \begin{bmatrix} 0.2 & 0.1 \end{bmatrix} & [\#2] \end{bmatrix}$$

for $\begin{bmatrix}1\\3\\2\end{bmatrix}$,

and:

$$\begin{bmatrix} [\#2] & \begin{bmatrix} 0.2 & 0.1 \end{bmatrix} \\ \begin{bmatrix} 0.1 \\ 0.4 \end{bmatrix} & \begin{bmatrix} \#1 & 0.2 \\ 0.3 & \#3 \end{bmatrix} \end{bmatrix}$$

for $\begin{bmatrix}2\\1\\3\end{bmatrix}$

(internal brackets are used for a better representation). If the cost of each structure is defined as the sum of components in upper diagonal brackets $\left(\begin{bmatrix}0.1\\0.4\end{bmatrix}$ in case I and $\begin{bmatrix}0.2 & 0.1\end{bmatrix}$ in case II$\right)$, it is obvious that case II $\left(\begin{bmatrix}2 & 1 & 3\end{bmatrix}^T\right)$ is better.

It this example, the size of groups is 2 (for columns $\begin{bmatrix}1 & 3\end{bmatrix}^T$ and [2]). For larger sizes, the number of all possible orders is $n!$ (the number of all possible permutations);

- **Reorder the new crystal.** All the possible orders are checked for their cost-effectiveness, and the one with the lowest cost is chosen for consolidation into the crystal. Therefore, its internal orders will not change in the forthcoming crystal growths. It means, for example, in order to combine $\begin{bmatrix}1\\3\end{bmatrix}$ and [2], activity [3] comes after [1] and, therefore, it is only possible to have $\begin{bmatrix}1\\3\\2\end{bmatrix}$ or $\begin{bmatrix}2\\1\\3\end{bmatrix}$ from 6 possible permutations for 3 tasks;
- **Check whether the original coupled block is obtained.** when this happens, a CBA containing all activities is formed; therefore, no more growth is possible. The order of all activities is consolidated; therefore, the algorithm is over.

4. Case study

A case study of the Fajr F.3 GA aircraft was chosen for investigation in order to demonstrate the applicability of this method. The Fajr F.3 GA aircraft is a lightweight full-composite small airplane; the maiden flight of its original model took place in 1997 (Figure 4). In order to develop a more advanced version of the vehicle, its design-process was studied for the possible improvements. Figure 5 represents the initial WTM of this project.

5. Results

This section has been divided into three subsections. In Subsection 5.1, the proposed method was applied to the case study. For demonstrating the applicability, an example was introduced that necessarily did not comply with the optimum solution. The execution plan was then readily extracted from the example and illustrated. Subsection 5.2 is dedicated to the pseudo-optimum solutions derived from the MATLAB® code. Meanwhile, Subsection 5.3 presents the results of the discrete-time simulation for pseudo-optimum solutions.

5.1. Applying Nabat algorithm to Fajr F.3 GA aircraft case study

To demonstrate the applicability of the method, the Nabat algorithm was applied to the WTM of the design process involved in the Fajr F.3 GA aircraft project. This sample is only an example and does not necessarily pertain to the optimum solution. Due to space limitations, only the determining (in term of the changes) steps are introduced in Table 1. For the detailed sample, Table A.I in Appendix 1 should be checked.

5.2. Finding an execution plan

The previously mentioned example has been introduced here for finding an execution plan using the

Figure 4. Fajr F.3 GA aircraft [33].

Select preliminary configuration alternative		0.0	0.0	0.0	0.3	0.0	0.3	0.0	0.2	0.0	0.0	0.0	0.0
Mathematical surface models	1.0		0.4	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Aerodynamic calculation	0.0	0.5		0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Preliminary structural arrangement	0.0	0.5	0.0		0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.3	0.1
Prepare for cabin and fuselage design	0.5	0.0	0.0	0.0		0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Develop structural design conditions	0.0	0.0	0.0	0.2	0.0		0.0	0.0	0.0	0.0	0.0	0.0	0.0
Integration propulsion	0.0	0.0	0.0	0.0	0.4	0.0		0.0	0.0	0.0	0.0	0.0	0.0
Perform preliminary weight and balance	0.0	0.0	0.0	0.0	0.3	0.0	0.4		0.0	0.0	0.0	0.5	0.0
Stability and control analysis	0.0	0.0	0.7	0.0	0.0	0.0	0.5	0.7		0.0	0.0	0.0	0.0
V-n diagram	0.0	0.0	0.1	0.0	0.0	0.2	0.0	0.1	0.0		0.4	0.0	0.0
Internal load distributions	0.0	0.0	0.0	0.0	0.0	0.5	0.0	0.5	0.0	0.3		0.0	0.0
Structural analysis	0.0	0.0	0.0	0.0	0.0	0.5	0.2	0.0	0.0	0.1	0.1		0.0
Preliminary production program	0.0	0.0	0.0	0.2	0.1	0.0	0.1	0.0	0.0	0.0	0.0	0.1	

Figure 5. Original WTM of Fajr F.3 GA aircraft [27].

Nabat algorithm. Accordingly, the exemplary structure was investigated. The following methodology helps find the execution plan:

- Execute the activities with respect to their order; ignore all feedbacks; assume that all the information is required;
- Find the first-order crystals; find all branches of the execution plan; execute all branches in parallel, wherever needed; ignore all feedbacks outside the crystals; continue the process until an acceptable level of convergence is attained;
- Find the ith order crystals ($i = 2, 3, \cdots$) until the whole CBA becomes a single crystal; for each order of the crystal, merge the parallel lines of execution into concurrent sub-blocks; ignore all feedbacks outside the crystals; continue the process until an acceptable level of convergence is attained.

Regarding the structure of the exemplary process for the F.3 aircraft design, at first, the activities were executed in order:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13 & 11 & 10 & 6 & 12\end{bmatrix}^T$$

as the zero iteration. All feedbacks were ignored, and all of the required initial information was assumed.

The first-order crystals that were identified are as follows:

$$\begin{bmatrix}7 & 5 & 1\end{bmatrix}^T \quad [8] \quad [9] \quad \begin{bmatrix}3 & 2\end{bmatrix}^T \quad \begin{bmatrix}4 & 13\end{bmatrix}^T$$

$$\begin{bmatrix}11 & 10\end{bmatrix}^T \quad [6] \quad [12].$$

Each crystal constituted a separate branch. The crystals were executed until they reached the acceptable levels of convergence. The second-order crystals were then identified.

While $\begin{bmatrix}7 & 5 & 1\end{bmatrix}^T$ is grown into $[7\ \ 5\ 1\ \ 8\ \ 9]^T$, the other former crystals remained intact. Therefore, we have:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9\end{bmatrix}^T \quad \begin{bmatrix}3 & 2\end{bmatrix}^T \quad \begin{bmatrix}4 & 13\end{bmatrix}^T$$

$$\begin{bmatrix}11 & 10\end{bmatrix}^T \quad [6] \quad [12].$$

Thus, the first two branches merged, and there remained six parallel branches. The third-order crystals are identified through these, wherein $\begin{bmatrix}3 & 2\end{bmatrix}^T$ is absorbed in $\begin{bmatrix}7 & 5 & 1 & 8 & 9\end{bmatrix}^T$ to form $\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2\end{bmatrix}^T$. Hence, we have:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2\end{bmatrix}^T \quad \begin{bmatrix}4 & 13\end{bmatrix}^T$$

$$\begin{bmatrix}11 & 10\end{bmatrix}^T \quad [6] \quad [12].$$

The first two branches merged, and only five parallel branches remained.

Next, the fourth-order crystals were identified. Again, the first two branches merged:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13\end{bmatrix}^T.$$

The remaining four branches were:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13\end{bmatrix}^T \quad \begin{bmatrix}11 & 10\end{bmatrix}^T$$

$$[6] \quad [12].$$

Then, the fifth-order crystals were identified. The first two initial branches merged again to form $\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13\end{bmatrix}^T$ and, thus, the structure becomes:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13 & 11 & 10\end{bmatrix}^T$$

$$[6] \quad [12].$$

The sixth-order crystal contained the whole column of activities, which is represented as:

$$\begin{bmatrix}7 & 5 & 1 & 8 & 9 & 3 & 2 & 4 & 13 & 11 & 10 & 6 & 12\end{bmatrix}^T.$$

Figure 6 shows the process in which the crystals are grown. In addition, Figure 7 illustrates the diagram for the execution plan.

5.3. Pseudo-optimum solutions

The Nabat algorithm was implemented into a MATLAB® code for finding the best possible orders

Table 1. An exemplary application of Nabat algorithm to Fajr F.3 GA aircraft case study (determining steps).

Structure	Description
$[1][2][3][4][5][6][7][8][9][10][11][12][13]$	Initial structure; activities are ordered numerically
$[13][3][4][2][6][9][7][12][5][8][1]\begin{bmatrix}[11]\\[10]\end{bmatrix}$	The structure is randomly shuffled; activities [10] and [11] are interdependent and able to make a crystal with $[11 \quad 10]^T$ order.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}[6][8][9][3]\begin{bmatrix}[5]\\[1]\end{bmatrix}[2][4][13][12][7]$	The structure is randomly shuffled; activities [5] and [1] are interdependent and able to make a crystal with $[1 \quad 5]^T$ order.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}[12]\begin{bmatrix}[3]\\[2]\end{bmatrix}[8][9][4][13]\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}[6]$	The structure is randomly shuffled; activities [2] and [3] are interdependent and able to make a crystal with order $[3 \quad 2]^T$; activity [7] and column $[5 \quad 1]^T$ are coupled and able to grow the crystal with order $[7 \quad 5 \quad 1]^T$.
$[13][4][12][6]\begin{bmatrix}[3]\\[2]\end{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\begin{bmatrix}[11]\\[10]\end{bmatrix}$	The structure is randomly shuffled; activities [8] and [9] and column $[7 \quad 5 \quad 1]^T$ are coupled and able to grow the crystal. The preferred order is $[7 \quad 5 \quad 1 \quad 8 \quad 9]^T$.
$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\\\begin{bmatrix}[3]\\[2]\end{bmatrix}\end{bmatrix}\begin{bmatrix}[4]\\[13]\end{bmatrix}\begin{bmatrix}[11]\\[10]\end{bmatrix}[6][12]$	The structure is randomly shuffled; there is a chance for columns $[3 \quad 2]^T$ and $[7 \quad 5 \quad 1 \quad 8 \quad 9]^T$ to form a larger crystal with order $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$. Activities [13] and [4] may form a new crystal with $[4 \quad 13]^T$.
$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\\\begin{bmatrix}[3]\\[2]\end{bmatrix}\end{bmatrix}\\\begin{bmatrix}[4]\\[13]\end{bmatrix}\end{bmatrix}[6][12]\begin{bmatrix}[11]\\[10]\end{bmatrix}$	Columns $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$ and $[4 \quad 13]^T$ are merged to form crystal $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13]^T$.
$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\\\begin{bmatrix}[3]\\[2]\end{bmatrix}\end{bmatrix}\\\begin{bmatrix}[4]\\[13]\end{bmatrix}\end{bmatrix}\\\begin{bmatrix}[11]\\[10]\end{bmatrix}\end{bmatrix}[6][12]$	The crystal is grown to $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13 \quad 11 \quad 10]^T$ after merging $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13]^T$ and $[11 \quad 10]^T$.

Table 1. An exemplary application of Nabat algorithm to Fajr F.3 GA aircraft case study (determining steps) (continued).

Structure	Description
$\left[\left[\left[\left[\left[7\right]\right]\right]\right]\right]$ $\left[\left[\left[\left[\left[5\right]\right]\right]\right]\right]$ $\left[\left[\left[\left[\left[1\right]\right]\right]\right]\right]$ $\left[\left[\left[\left[8\right]\right]\right]\right]$ $\left[\left[\left[\left[9\right]\right]\right]\right]$ $\left[\left[\left[3\right]\right]\right]$ $\left[\left[\left[2\right]\right]\right]$ $\left[\left[4\right]\right]$ $\left[\left[13\right]\right]$ $\left[\left[11\right]\right]$ $\left[\left[10\right]\right]$ $\left[6\right]$ $\left[12\right]$	The final agglomeration of all activities is brought in this column. All coupled activities are ordered, and it is possible to find an execution plan.

for the activities in a CBA. The code attempts to grow the crystals to achieve the lowest rework level. In order to reach this goal, the solutions for several runs were found and compared. The most suitable solutions were chosen according to the two basic criteria of partitioning: less feedback and feedbacks closer to the main diagonal. A third criterion was added to consider the numerical values into the feedbacks; the farther a feedback value is from the main diagonal, the more it is encouraged to be smaller.

These three criteria are summarized in the following cost function:

$$cf = \sum_{i=1}^{n} \sum_{j=i+1}^{n} \left(A_{ij} \times (j-i)^2\right), \tag{2}$$

where cf is the cost function, and A_{ij} is the entry of WTM situated in row i and column j. Accordingly, three sample configurations involving rather low costs are illustrated as examples in Table 2. In the next section, the results of 100 samples have been used to create a plot.

Although the aforementioned cost function was devised to find the most probable solutions for restructuring the CBA, the function with the lowest value of this parameter is not necessarily the best. Therefore, a discrete-time simulation is required.

6. Simulation and discussion

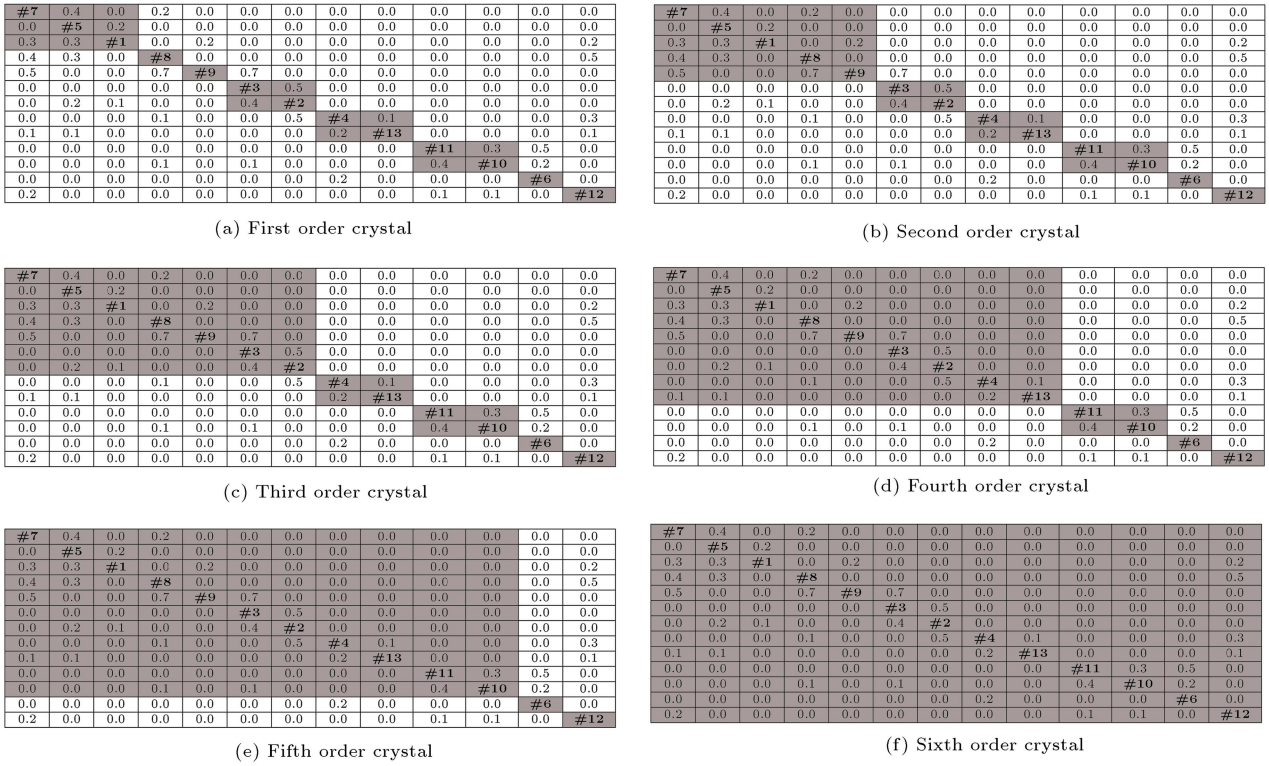
The performance of the Nabat algorithm has been expressed with some necessary commentaries. Firstly, the output of the Nabat algorithm is more than just a sequence of activities with possible breaking points, while a certain sequence of activities involved in a design process have a unique meaning in the existing methods; in the Nabat algorithm, it may represent several structures since the crystallization procedure can vary. In brief, it can be said that the Nabat algorithm generates solutions that are richer in information than the conventional methods.

In the discrete-time simulation, the existing (conventional) methods usually follow a single (sequential) or several (parallel) branches of execution. However, it has been shown that neither is optimal, and recent efforts are based on the partial concurrency among the activities [32]. The Nabat algorithm is better at classifying the partial concurrency of the activities. In addition, even though the Nabat algorithm does not permit the engagement of all activities in the couplings, it does not ignore the feedbacks permanently. As a result, although this algorithm may result in longer completion times compared to the existing breaking schemes, it definitely lowers the risk of rework.

We compared the performance of the Nabat algorithm with three other methods:

- Sequential Stochastic (SS);
- Fully Parallel Stochastic (FPS);
- Random Breaking Strategy (RBS).

Whereas in the SS method, the duration of each activity and the probability of feedbacks are stochastically changed, in the FPS method, similar decisions are made with respect to the preference of the activities performed. Regarding these two methods, the WTM is partitioned using the classic method proposed by Steward [11]. The RBS does not generally address a set of solutions and tries to follow an optimization criterion. In order to perform a better comparison,



(a) First order crystal

(b) Second order crystal

(c) Third order crystal

(d) Fourth order crystal

(e) Fifth order crystal

(f) Sixth order crystal

Figure 6. Exemplar crystal growth stages using Nabat algorithm.

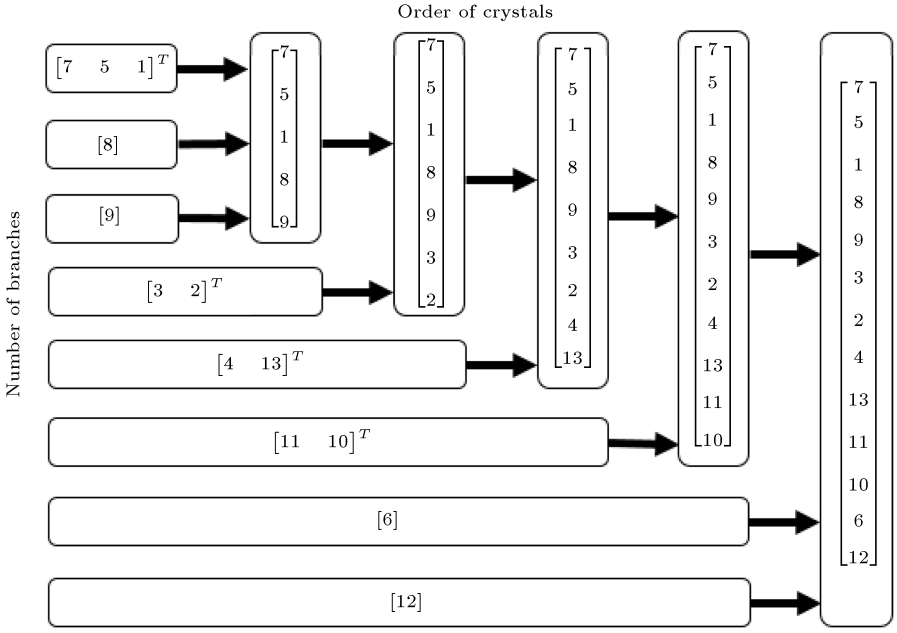


Figure 7. Exemplar crystal growth using Nabat algorithm; the execution plan.

several possible solutions are extracted by relaxing the optimization criteria. Soltammohammad and Malaek used the same case study to develop such a model [27].

Figure 8 shows an overlaid plot of results for the SS, FPS, RBS, and Nabat algorithms. For each case, 20 pseudo-optimum data were introduced. For all the data, the cost function in Eq. (2) was calculated, and the results were plotted in a (total time of completion versus cost function) scatter.

Furthermore, the performance of the Nabat algorithm was compared to each of the three other methods for better clarity. Figures 9, 10, and 11 compare the

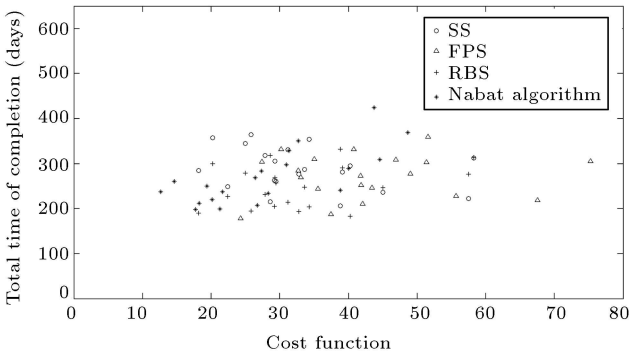


Figure 8. Total time of completion versus cost function for SS, FPS, RBS, and Nabat algorithm.

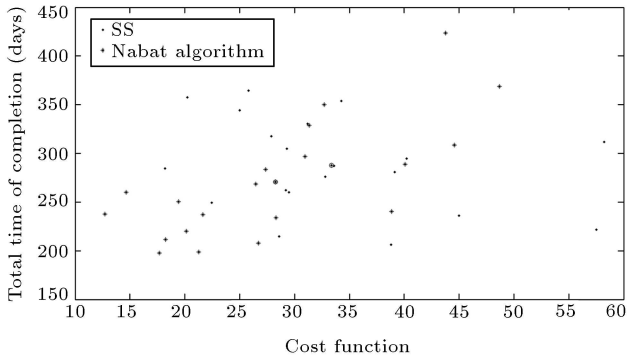


Figure 9. Total time of completion versus cost function for SS and Nabat algorithm.

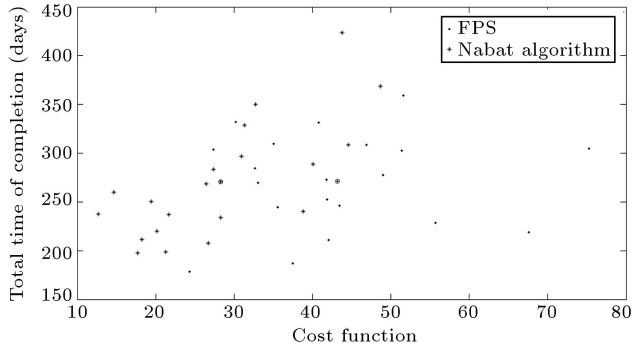


Figure 10. Total time of completion versus cost function for FPS and Nabat algorithm.

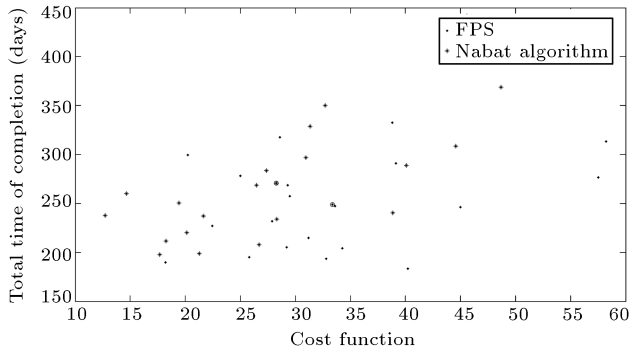


Figure 11. Total time of completion versus cost function for RBS and Nabat algorithm.

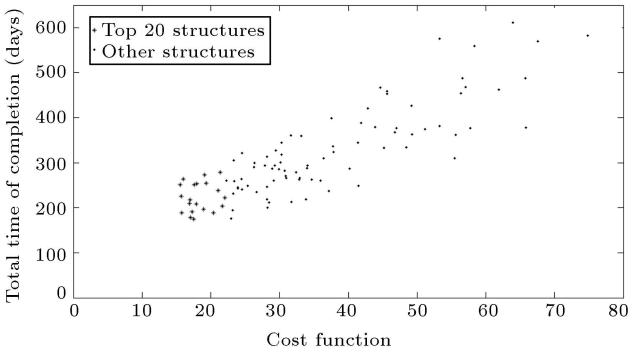


Figure 12. Total time of completion versus cost function (2).

data obtained from the Nabat algorithm with those from the SS, FPS, and RBS methods, respectively.

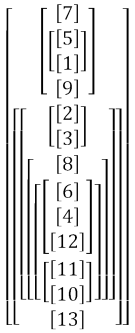
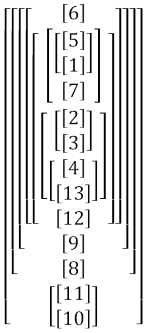
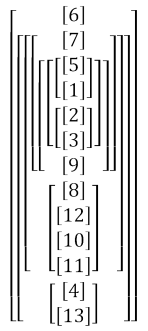
The Nabat algorithm performed considerably better than the SS method (271 to 288 days on average). The superiority is evident in both the total time of completion and better partitioning (lower cost function: 28.3 to 33.3 average) (Figure 9). In comparison with the FPS method, the Nabat algorithm represents almost no meaningful improvement regarding the total time of completion (271 to 273 days on average); however, the structures are much better partitioned (lower cost function: 43.2 to 33.3 average) (Figure 10). In comparison with the RBS method, the Nabat algorithm performs better in partitioning (lower cost function 28.3 to 33.3 on average); however, the total time of completion was lower for RBS (271 to 249 days on average). This is justifiable due to the higher levels of rework risk that arise due to ignoring the intra-CBA feedbacks. This is the reason for not retaining any rework risk in the Nabat algorithm (Figure 11).

To measure the optimality of a solution, the best 20 configurations (illustrated with '*') in the cost function of Eq. (2) were compared with 80 other randomly chosen structures (marked with '.'). Figure 12 shows the relation between the defined cost function and the simulated time of completion.

As observed, there is a direct and rather strong connection between the cost function in the Nabat algorithm and the lower times of completion. Although the top 20% of the results are situated within [15.54, 22.06], it constitutes only 8.17% of the distribution interval. In this case, the highest simulated time for the top 20 structures (278 days) is only 24% above its mean value (223 days), which is generally a good approximation.

Choosing the best structure is a matter of organization. As we have assumed that an ideal information exchange exists between all the design teams, there may be some restrictions or delays caused by a certain group or team, which does not seem problematic in the abstract view. As a result, it is recommended to choose

Table 2. Three samples of pseudo-optimum solutions and their prevalent structures and cost functions.

Sample	WTM													Structure	cf
I	**#7**	0.4	0	0	0	0	0.2	0	0	0	0	0	0		17.7
	0	**#5**	0.2	0	0	0	0	0	0	0	0	0	0		
	0.3	0.3	**#1**	0.2	0	0	0	0	0	0.2	0	0	0		
	0.5	0	0	**#9**	0	0.7	0.7	0	0	0	0	0	0		
	0	0.2	0.1	0	**#2**	0.4	0	0	0	0	0	0	0		
	0	0	0	0	0.5	**#3**	0	0	0	0	0	0	0		
	0.4	0.3	0	0	0	0	**#8**	0	0	0.5	0	0	0		
	0	0	0	0	0	0	0	**#6**	0.2	0	0	0	0		
	0	0	0	0	0.5	0	0.1	0	**#4**	0.3	0	0	0.1		
	0.2	0	0	0	0	0	0	0	0	**#12**	0.1	0.1	0		
	0	0	0	0	0	0	0	0.5	0	0	**#11**	0.3	0		
	0	0	0	0	0	0.1	0.1	0.2	0	0	0.4	**#10**	0		
	0.1	0.1	0	0	0	0	0	0	0.2	0.1	0	0	**#13**		
II	**#6**	0	0	0	0	0	0.2	0	0	0	0	0	0		24.4
	0	**#5**	0.2	0	0	0	0	0	0	0	0	0	0		
	0	0.3	**#1**	0.3	0	0	0	0	0	0.2	0	0	0		
	0	0.4	0	**#7**	0	0	0	0	0	0	0	0	0		
	0	0.2	0	0	**#2**	0.4	0	0	0	0	0	0	0		
	0	0	1.0	0	0.5	**#3**	0	0	0	0	0	0	0		
	0	0	0	0	0.5	0	**#4**	0.1	0.3	0	0.1	0	0		
	0	0.1	0	0.1	0	0	0.2	**#13**	0.1	0	0	0	0		
	0.5	0	0	0.2	0	0	0	0	**#12**	0	0	0.1	0.1		
	0	0	0	0.5	0	0.7	0	0	0	**#9**	0.7	0	0		
	0	0.3	0	0.4	0	0	0	0	0.5	0	**#8**	0	0		
	0.5	0	0	0	0	0	0	0	0	0	0.5	**#11**	0.3		
	0.2	0	0	0	0	0.1	0	0	0	0	0.1	0.4	**#10**		
III	**#6**	0	0	0	0	0	0	0	0	0	0	0.2	0		29.2
	0	**#7**	0.4	0	0	0	0	0	0	0	0	0	0		
	0	0	**#5**	0.2	0	0	0	0	0	0.2	0	0	0		
	0	0.3	0.3	**#1**	0	0	0.2	0	0	0	0	0	0		
	0	0	0.2	1.0	**#2**	0.4	0	0	0	0	0	0	0		
	0	0	0	0	0.5	**#3**	0	0	0	0	0	0	0		
	0	0.5	0	0	0	0.7	**#9**	0.7	0	0	0	0	0		
	0	0.4	0.3	0	0	0	0	**#8**	0.5	0	0	0	0		
	0.5	0.2	0	0	0	0	0	0	**#12**	0.1	0.1	0	0		
	0.2	0	0	0	0	0.1	0	0.1	0	**#10**	0.4	0	0		
	0.5	0	0	0	0	0	0	0.5	0	0.3	**#11**	0	0		
	0	0	0	0	0.5	0	0	0.1	0.3	0	0	**#4**	0.1		
	0	0	0.1	0	0	0	0	0	0.1	0	0	0.2	**#13**		

the structures that are better suited to the respective organizational restrictions.

7. Conclusion

The authors proposed a heuristic method for restructuring a time-based WTM in this paper. Although a plethora of partitioning/tearing methods are available at present, the method introduced in this paper uses a gradual combination that conforms to acceptable standards in terms of the total completion time. In order to demonstrate its applicability and superiority, an industrial case study of the Fajr F.3 GA aircraft was investigated.

The Nabat algorithm proposes more than just a sequence of activities with breaking points. It entails an execution plan, which is explicitly contained in it. Obtaining an execution plan makes it possible to plan simulations that are considerably more efficient. Regarding the mentioned case study, it was demonstrated that the code, including the algorithm, results in configurations with generally lower costs and lower total times of completion.

Although the proposed method is mainly derived from an academic point of view, with a better extension of the model, it is possible to manage more complex and realistic problems. In order to enable this method to be augmented with more realistic data from complex systems, this method was applied to one case study. Without loss of generality, it is possible to apply it to other engineering cases.

Organizational limitations are not considered in the algorithm. For example, though it may be proposed that all tasks should cooperate, a situation may arise where tasks are assigned to departments/groups in which there is a lack of cooperation due to certain problems. One of the main probable extensions of the method in future works involves the consideration of organizational limitations. These considerations will help detect the most suited execution plan among the pseudo-optimum solutions presented with respect to the case study.

References

1. Browning, T.R. "Process integration using the design structure matrix", *Systems Engineering*, **5**(3), pp. 180-193 (2002).

2. Baccarini, D. "The concept of project complexity - A review", *International Journal of Project Management*, **14**(4), pp. 201-204 (1996).

3. Suh, N.P., *Complexity: Theory and Applications*, Oxford: Oxford University Press (2005).

4. Yassine, A.A., Falkenburg, D., and Chelst, K. "Engineering design management: an information structure approach", *International Journal of Production Research*, **13**(37), pp. 2957-2975 (1999).

5. Browning, T.R. "Design structure matrix extensions and innovations: a survey and new opportunities", *IEEE Transactions on Engineering Management*, **63**(1), pp. 27-52 (2016).

6. Browning, T.R. "Applying the design structure matrix to system decomposition and integration problems: a review and new directions", *IEEE Transactions on Engineering Management*, **48**(3), pp. 292-306 (2001).

7. Spinner, M.P., *Improving Project Management Skills and Techniques*, Englewood Cliffs, Prentice Hall (1989).

8. Shishko, R., Aster, R., and Cassingham, R.C., *NASA Systems Engineering Handbook*, Washington, D.C.: National Aeronautics and Space Administration (1995).

9. Karniel, A. and Reich, Y. "From DSM-based planning to design process simulation: a review of process scheme logic verification issues", *IEEE Transactions on Engineering Management*, **4**(56), pp. 636-649 (2009).

10. Smith, R.P. and Eppinger, S.D. "Identifying controlling features of engineering design iteration", *Management Science*, **3**(43), pp. 276-293 (1997).

11. Steward, D.V. "Planning and managing the design of systems" In *Proceedings of Portland International Conference on Management of Engineering and Technology* (1991).

12. Horowitz, E., Sahni, S., and Mehta, D.P., *Fundamentals of Data Structures in C*, Summit, NJ: Silicon Press (2007).

13. Lawler, E., *Combinatorial Optimization: Networks and Matroids*, Mineola, NY: Dover Publications (2004).

14. Kingston, J.H., *Algorithms and Data Structures: Design, Correctness, Analysis*, Reading, MA: Addison-Wesley (2004).

15. Meier, C., Yassine, A.A., and Browning, T.R. "Design process sequencing with competent genetic algorithms", *Journal of Mechanical Design*, **6**(129), p. 566 (2007).

16. Sen, C., Ameri, F., and Summers, J.D. "An entropic method for sequencing discrete design decisions", *Journal of Mechanical Design*, **10**(132) (2010). DOI: 10.1115/1.4002387

17. Tsai, J., Fang, J., and Chou, J. "Optimized task scheduling and resource allocation on cloud computing environment using improved differential evolution algorithm", *Computers & Operations Research*, **12**(40), pp. 3045-3055 (2013).

18. Maurer, M.S. "Structural awareness in complex product design", PhD Dissertation, TUM (2007).
19. Warfield, J.N. "Binary matrices in system modeling", *IEEE Transactions on Systems, Man, and Cybernetics*, **5**(SMC-3), pp. 441-449 (1973)
20. Yang, Z., Zhou, J., Deng, C., and Shao, X. "Development of a design structure matrix partitioning method towards effective design collaboration", *International Journal of Manufacturing Technology and Management*, **4**(25), p. 177 (2012).
21. Shacham, M. and Kehat, E. "Converging interval methods for the iterative solution of a non-linear equation", *Chemical Engineering Science*, **12**(28), pp. 2187-2193 (1973).
22. Kusiak, A. and Wang, J. "Efficient organizing of design activities", *International Journal of Production Research*, **4**(31), pp. 753-769 (1993).
23. Eppinger, S.D., Whitney, D.E., Smith, R.P., and Gebala, D.A. "Organizing the tasks in complex design projects", *Lecture Notes in Computer Science Computer-Aided Cooperative Product Development*, pp. 229-252 (2005).
24. Zhang, H., Qiu, W., and Zhang, H. "An approach to measuring coupled tasks strength and sequencing of coupled tasks in new product development", *Concurrent Engineering*, **4**(14), pp. 305-311 (2006).
25. Su, J.C., Chen, S., and Lin, L. "A structured approach to measuring functional dependency and sequencing of coupled tasks in engineering design", *Computers & Industrial Engineering*, **1**(45), pp. 195-214 (2003).
26. Xu, C., Li, L., and Liu, X. "Tearing method of coupled activity set within concurrent design process (in Chinese)", *Journal of Tongji University Natural Sciences*, **38**, pp. 427-431 (2010).
27. Soltanmohammad, B. and Malaek, S.M. "A new method for design cycle period management in aircraft design process", *Aircraft Engineering and Aerospace Technology*, **80**(5), pp. 497-509 (2008).
28. Bashir, H.A., Alzebdeh, K., and Abdo, J. "An eigenvalue based approach for assessing the decomposability of interdependent design project tasks", *Concurrent Engineering*, **17**(1), pp. 35-42 (2009).
29. Xiao, R., Chen, T., and Chen, W. "A new approach to solving coupled task sets based on resource balance strategy in product development", *International Journal of Material and Product Technology*, **39**, pp. 251-270 (2010).
30. Browning, T.R., Deyst, J.J., Eppinger, S.D., and Whitney, D.E. "Adding value in product development by creating information and reducing risk", *IEEE Transactions on Engineering Management*, **4**(49), pp. 443-458 (2002).
31. Kosari, A., Haji Jafari, A., and Fakoor, M. "On equivalency between numerical process DSM and state-space representation", *IEEE Transactions on Engineering Management*, **4**(63), pp. 404-413 (2016).
32. Cho, S. and Eppinger, S.D. "A simulation-based process model for managing complex design projects", *IEEE Transactions on Engineering Management*, **52**(3), pp. 316-328 (2005).
33. Available at: http://www.airframer.com/aircraft_detail.html?model=Fajr_F-3. [Accessed: 18-Jun-2017].

Appendix

In this appendix, a whole run of Nabat algorithm is shown (Table A.I). Due to realistic situations, some steps may seem repetitive.

Table A.I. An exemplar application of Nabat algorithm to Fajr F.3 GA aircraft case study (all steps).

Structure	Description
[1][2][3][4][5][6][7][8][9][10][11][12][13]	Initial structure; activities are ordered numerically.
[5][2][10][6][13][7][4][1][3][12][8][9][11]	The structure is randomly shuffled; no chance for crystal formation/growth.
[4][5][11][2][7][12][9][8][6][1][10][3][13]	The structure is randomly shuffled; no chance for crystal formation/growth.
[8][6][12][4][3][7][11][10][2][5][9][13][1]	The structure is randomly shuffled; activities [10] and [11] are interdependent and able to make a crystal.
[8][6][12][4][3][7] $\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [2][5][9][13][1]	Activities [10] and [11] form a crystal; the order of $[11 \quad 10]^T$ is better than $[10 \quad 11]^T$ and, hence, is replaced.
[13][3][4][2][6][9][7][12][5][8][1] $\begin{bmatrix}[11]\\[10]\end{bmatrix}$	The structure is randomly shuffled; no chance for crystal formation/growth.

Table A.I. An exemplar application of Nabat algorithm to Fajr F.3 GA aircraft case study (all steps) (continued).

Structure	Description
$\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [6][8][9][3][5][1][2][4][13][12][7]	The structure is randomly shuffled; activities [5] and [1] are interdependent and able to make a crystal.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [6][8][9][3] $\begin{bmatrix}[5]\\[1]\end{bmatrix}$ [2][4][13][12][7]	Activities [5] and [1] form a crystal; the order of $[5 \quad 1]^T$ is better than $[1 \quad 5]^T$ and, then, preserved.
[2][13] $\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [12][4][7][9] $\begin{bmatrix}[5]\\[1]\end{bmatrix}$ [3][8][6]	The structure is randomly shuffled; no chance for crystal formation/growth.
[2][3][7] $\begin{bmatrix}[5]\\[1]\end{bmatrix}$ [6][4][12][8][9][13] $\begin{bmatrix}[11]\\[10]\end{bmatrix}$	The structure is randomly shuffled; no chance for crystal formation/growth.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [12][2][3][8][9][4][13][7] $\begin{bmatrix}[5]\\[1]\end{bmatrix}$ [6]	The structure is randomly shuffled; activities [2] and [3] are interdependent and able to make a crystal; activity [7] and column $[5 \quad 1]^T$ are coupled and able to grow the crystal.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [12] $\begin{bmatrix}[2]\\[3]\end{bmatrix}$ [8][9][4][13] $\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}$ [6]	Crystal $[2 \quad 3]^T$ is formed and crystal $[5 \quad 1]^T$ is grown to $[7 \quad 5 \quad 1]^T$, whose order is so be checked.
$\begin{bmatrix}[11]\\[10]\end{bmatrix}$ [12] $\begin{bmatrix}[3]\\[2]\end{bmatrix}$ [8][9][4][13] $\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}$ [6]	$[7 \quad 5 \quad 1]^T$ has the best order among all possible combinations (here, $[7 \quad 5 \quad 1]^T$ and $[5 \quad 1 \quad 7]^T$; the order of $[5 \quad 1]^T$ from previous stage remains intact). Order $[3 \quad 2]^T$ is preferred over $[2 \quad 3]^T$ and, therefore, is chosen.
[13][4][12][6] $\begin{bmatrix}[3]\\[2]\end{bmatrix}$ [8] $\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}$ [9] $\begin{bmatrix}[11]\\[10]\end{bmatrix}$	The structure is randomly shuffled; activities [8] and [9] and column $[7 \quad 5 \quad 1]^T$ are coupled and able to grow the crystal.
[13][4][12][6] $\begin{bmatrix}[3]\\[2]\end{bmatrix}$ $\begin{bmatrix}[8]\\[7]\\[5]\\[1]\\[9]\end{bmatrix}$ $\begin{bmatrix}[11]\\[10]\end{bmatrix}$	Activities [8] and [9] and column $[7 \quad 5 \quad 1]^T$ are merged to form a larger crystal.

Table A.I. An exemplar application of Nabat algorithm to Fajr F.3 GA aircraft case study (all steps) (continued).

Structure	Description
$[13][4][12][6]\begin{bmatrix}[3]\\[2]\end{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\begin{bmatrix}[11]\\[10]\end{bmatrix}$	$[8\quad 7\quad 5\quad 1\quad 9]^T$ is replaced by $[7\quad 5\quad 1\quad 8\quad 9]^T$ with the best possible order (among $\begin{bmatrix}7\\5\\1\\8\\9\end{bmatrix}, \begin{bmatrix}7\\5\\1\\9\\8\end{bmatrix}, \begin{bmatrix}8\\7\\5\\1\\9\end{bmatrix}, \begin{bmatrix}8\\9\\7\\5\\1\end{bmatrix}, \begin{bmatrix}9\\7\\5\\1\\8\end{bmatrix}$ & $\begin{bmatrix}9\\8\\7\\5\\1\end{bmatrix}$; $\begin{bmatrix}7\\5\\1\end{bmatrix}$ is preserved in all).
$\begin{bmatrix}[3]\\[2]\end{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}[13][4]\begin{bmatrix}[11]\\[10]\end{bmatrix}[6][12]$	The structure is randomly shuffled; there is a chance for columns $[3\quad 2]^T$ and $[7\quad 5\quad 1\quad 8\quad 9]^T$ to form a larger crystal. Activities [13] and [4] may form a new crystal, too.
$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\\\begin{bmatrix}[3]\\[2]\end{bmatrix}\end{bmatrix}\begin{bmatrix}[4]\\[13]\end{bmatrix}\begin{bmatrix}[11]\\[10]\end{bmatrix}[6][12]$	Larger crystals are formed; $[7\quad 5\quad 1\quad 8\quad 9\quad 3\quad 2]^T$ is preferred over $[3\quad 2\quad 7\quad 5\quad 1\quad 8\quad 9]^T$ and, therefore, replaces it. Similarly, $[4\quad 13]^T$ replaces $[13\quad 4]^T$. Columns $[7\quad 5\quad 1\quad 8\quad 9\quad 3\quad 2]^T$ and $[4\quad 13]^T$ may be combined.
$\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\\\begin{bmatrix}[3]\\[2]\end{bmatrix}\end{bmatrix}\\\begin{bmatrix}[4]\\[13]\end{bmatrix}\end{bmatrix}[6][12]\begin{bmatrix}[11]\\[10]\end{bmatrix}$	Columns $[7\quad 5\quad 1\quad 8\quad 9\quad 3\quad 2]^T$ and $\begin{bmatrix}4\\13\end{bmatrix}$ are merged to form crystal $[7\quad 5\quad 1\quad 8\quad 9\quad 3\quad 2\quad 4\quad 13]^T$ that is preferred over $[4\quad 13\quad 7\quad 5\quad 1\quad 8\quad 9\quad 3\quad 2]^T$ in order.
$[13][4][12][6]\begin{bmatrix}[3]\\[2]\end{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}\begin{bmatrix}[11]\\[10]\end{bmatrix}$	$[8\quad 7\quad 5\quad 1\quad 9]^T$ is replaced by $[7\quad 5\quad 1\quad 8\quad 9]^T$ with the best possible order (among $\begin{bmatrix}7\\5\\1\\8\\9\end{bmatrix}, \begin{bmatrix}7\\5\\1\\9\\8\end{bmatrix}, \begin{bmatrix}8\\7\\5\\1\\9\end{bmatrix}, \begin{bmatrix}8\\9\\7\\5\\1\end{bmatrix}, \begin{bmatrix}9\\7\\5\\1\\8\end{bmatrix}$ & $\begin{bmatrix}9\\8\\7\\5\\1\end{bmatrix}$; $\begin{bmatrix}7\\5\\1\end{bmatrix}$ is preserved in all).
$\begin{bmatrix}[3]\\[2]\end{bmatrix}\begin{bmatrix}\begin{bmatrix}[7]\\[5]\\[1]\end{bmatrix}\\[8]\\[9]\end{bmatrix}[13][4]\begin{bmatrix}[11]\\[10]\end{bmatrix}[6][12]$	The structure is randomly shuffled; there is a chance for columns $[3\quad 2]^T$ and $[7\quad 5\quad 1\quad 8\quad 9]^T$ to form a larger crystal. Activities [13] and [4] may form a new crystal, too.

Table A.I. An exemplar application of Nabat algorithm to Fajr F.3 GA aircraft case study (all steps) (continued).

Structure	Description
$\left[\left[\left[\left[7\right]\right]\right]\right]\left[\left[4\right]\right]\left[\left[11\right]\right]\left[6\right]\left[12\right]$	Larger crystals are formed; $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$ is preferred over $[3 \quad 2 \quad 7 \quad 5 \quad 1 \quad 8 \quad 9]^T$ and, therefore, replaces it. Similarly, $[4 \quad 13]^T$ replaces $[13 \quad 4]^T$. Columns $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$ and $[4 \quad 13]^T$ may be combined.
$\left[\left[\left[\left[\left[7\right]\right]\right]\right]\right]\left[6\right]\left[12\right]\left[\left[11\right]\right]$	Columns $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$ and $\begin{bmatrix}4\\13\end{bmatrix}$ are merged to form crystal $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13]^T$ that is preferred over $[4 \quad 13 \quad 7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2]^T$ in order.
$\left[\left[\left[\left[\left[7\right]\right]\right]\right]\right]\left[\left[11\right]\right]\left[12\right]\left[6\right]$	The structure is randomly shuffled; there is a chance for columns $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13]^T$ and $\begin{bmatrix}11\\10\end{bmatrix}$ to merge and form a larger crystal.
$\left[\left[\left[\left[\left[7\right]\right]\right]\right]\left[\left[\left[4\right]\left[13\right]\left[\left[11\right]\right]\right]\right]\right]\left[6\right]\left[12\right]$	The crystal is grown to $[7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13 \quad 11 \quad 10]^T$, which is more favorable than the other possible order of $[11 \quad 10 \quad 7 \quad 5 \quad 1 \quad 8 \quad 9 \quad 3 \quad 2 \quad 4 \quad 13]^T$. There is a chance to form a larger crystal containing all activities.
$\left[\left[\left[\left[\left[\left[7\right]\right]\right]\right]\left[\left[\left[4\right]\left[13\right]\left[\left[11\right]\right]\right]\right]\right]\left[6\right]\left[12\right]\right]$	The final agglomeration of all activities is brought in this column. All coupled activities are ordered, and it is possible to find an execution plan.

Biographies

Mohammad Haji Jafari (S'08-M'10) received the BS and MS degrees in Aerospace Engineering from Sharif University of Technology (SUT), Tehran, Iran. He managed to receive his PhD degree in Aerospace Engineering at Faculty of New Sciences and Technologies (FNST), University of Tehran (UT), Tehran, Iran in 2018. His current research interests include new product strategy and management of complex engineering processes. Dr. Haji Jafari has some periods of practical experience in design, commissioning, and operating of renewable energy sources (especially wind turbines), beside works in management of life cycle cost estimation of aerospace systems.

Amirreza Kosari (S'98-M'00) received the BS in Aerospace Engineering from Amirkabir University of Technology, Tehran, Iran, and MS and PhD degrees in Aerospace Engineering from Sharif University of Technology (SUT), Tehran, Iran. He is an Assistant Professor at University of Tehran (UT), where he has been teaching and doing research in the system design, flight dynamics, and robotic navigation. His research interests also include optimal control and hybrid system analysis and design. He is the author of some research papers in the areas indicated above. Dr. Kosari has been the Head of Mechatronics Group of Faculty of New Sciences and Technologies (FNST) from 2012 to 2014.

Mehdi Fakoor (M'01-SM'03) is a member of Iranian Society of Mechanical Engineers (ISME) and an Assistant Professor of Aerospace Engineering at University of Tehran (UT). He has received the BS and PhD degrees in Mechanical Engineering and MS degree in Space Engineering from Iran University of Science & Technology (IUST), Tehran, Iran. Dr. Fakoor's domains of interest include space structure design and analysis and developing complex aerospace systems.