Journal of the Korean Institute
of Industrial Engineers
Vol.24, NO.2, JUNE, 1998

A Heuristic Approach Solving for the Complex Design with Precedence Constraints in Concurrent Engineering

복합설계를 위한 동시공학적 접근방법

Moonsoo Cho* · Changyeong Kim*

조문수* · 김창영*

Abstract

Engineering design involves the specification of many variables that define a product, how it is made, and how it behaves. Before some variables can be determined, other variables must first be known or assumed. This fact implies a precedence order of the variables, and of the tasks of determining these variable consequently. Moreover, design of complex systems may involve a large number of design activities. In this paper, the activity-activity incidence matrix is considered as a representation of design activity analysis which mainly focuses on the precedence constraint. In order to analyze the activity-activity incidence matrix, a heuristic algorithm is proposed, which transforms an activity-activity, parameter-formula, and parameter-parameter incidence matrix into a lower triangular form. The analysis of the structured matrices can not only significantly reduce the overall project complexity by reorganizing few critical tasks in practice, but also aims at obtaining shorter times considering the solution structure by exploring concurrency.

1. Introduction

Producing of high quality products requires the proper design, manufacturing, development, support, and disposal of products. In response to the customer needs, the engineering design process should deliver better quality products and systems in a shorter time. The sequential design process is likely to lead to a long design cycle time. One way of reducing the design cycle is to use decomposition methods, which break the overall design tasks into smaller groups of activities that might be executed concurrently.

The subject of this paper is the tools and techniques for modeling concurrent design. The modeling problems can be generally decomposed into two sub-problems such as what is it that must be modeled, and how is it to be represented. In most industrial applications, these two sub-problems can be regarded as orthogonal, and their solutions can be based upon different techniques, quite independent of each other. Therefore, a decomposition of

---

* Dept. of Industrial Engineering, Soongsil Univeristy

design activities could be viewed as a strategy for problem formulation in terms of intelligent computer aided design.

There has been a growing realization that the interactions among the disciplines and subsystems is an important factor for successful design. Most design configuration decisions must be made before much deign detail is available [10]. Design can be defined as the process of conception, invention, visualization, calculation, marshaling, refinement, and specifying of details which determine the form of an engineering product [4]. Engineering design is especially a complex process that combines creative thinking, experience, intuition, and quantitative analysis. The requirements for the design of a complex system are diverse and often conflicting. Complex designs may involve activity variables defining a product or system, how it is made, and how it behaves.

The precedence considerations may prevent same or similar activities from being performed twice in the design of different products or systems. For the engineers, the challenges of simultaneous engineering are particularly difficult when many design and analysis activities are interdependent and cannot be performed in series or in parallel.

Creating the more detailed description we seek involves explicitly mapping out the technical aspects of the design procedure. We contend that to be most useful, the design representation must include not only the sequence of the tasks but also the many technical relationships among the tasks.

Consider two design tasks, labelled A and B. Figure 1 shows digraphs of three possible ways in which the two can be related. If task B simply requires the output of task A, then the two tasks are dependent and are typically done in series. On the other hand, the two would be entirely independent if tasks A and B could be performed simultaneously with no interaction between the designers. Finally, if task A needs information from task B, and also task B requires knowledge of task A's results, then the two tasks are interdependent [12].

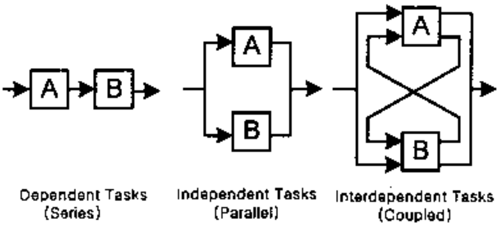


Figure 1. Three possible sequences for two design tasks

Coordinating either the dependent (series) tasks or the independent (parallel) tasks is quite straightforward. Certainly with no limitation on resources, the parallel tasks can be completed more quickly. The interdependent (coupled) tasks are much more challenging to organize, often requiring much more design time and many iterations of information transfer.

To illustrate using a familiar theme, we can envision task A to represent a product design function, and task B to represent a product manufacturing function. Then our series model depicts the outdated "throw the design over the wall" methodology. The parallel task model might then represent an idyllic view of simultaneous engineering, where both design and manufacturing functions are given the same challenge, and they develop product and process concurrently without complex interaction. The coupled task model is a more realistic diagram of simultaneous engineering, where the information transfer is essential and iteration is typical.

PERT software tools can typically analyze project sequence diagrams only if they contain no coupling loops. The representation requires the coupled tasks to be bundled into larger design tasks. If the project planner chooses to consider the tasks separately, then the essential information coupling must be neglected.

In order to perform activities concurrently that are interdependent, negotiation among specialists might be required [3]. Potentially many engineers from various disciplines must be involved in the complex decision process [11]. Strategically decoupling the major design tasks into subsystems can reduce the size of the working design groups, and this may improve performance of the design process [2,9].

Concurrent can be defined as a systematic approach to the integrated, simultaneous design of products and related processes, including manufacture and support. This definition is intended to emphasize, form the outset, the consideration of all elements of the product life cycle from concept through disposal, including quality, cost, schedule, and user requirements.

The general features of concurrent engineering include a top-down design process approach based upon system engineering principles, multi-functional design teams, optimization of product and process characteristics, and execution in engineering environment [12]. Furthermore the objective of concurrent engineering is that of simultaneously considering the life-cycle impacts during preliminary system design along with the paramount consideration of functionality [1].

Hence, the product design should be viewed as a strategic task that has major effect on the subsequent production related activities in concurrent engineering [5]. Design of products determines their quality, and 70 to 80% of the final production cost [8]. Another study shows that about 70% of the life cycle cost of a product is determined at the conceptual design stage [7].

2. Methodology of Decoupling Design Activities

The first phase in the proposed approach to decomposition and sequencing of design or analysis activities discussed here is defining the precedence relationship between design activities, and representing them with a graph or an activity-activity incidence matrix. The objective of this paper is to develop an efficient algorithm for transforming an unstructured activity-activity incidence matrix representing relationship between activities' precedence into a structure lower triangular form.

A proposed heuristic called a triangularization algorithm can be applied at various level, i.e., general or detailed activities. The applied examples presented here are divided into three cases such as activity-activity incidence matrix, parameter-parameter incidence matrix, and parameter-formula incidence matrix. The precedence relationship between design activities in Table 1 is considered in order to present the general problem in this paper.

Table 1. Precedence relationship between design activities

Design Activity	Predecessor(s)	Design Activity	Predecessor(s)
1		9	1,2,4,12
2	1,3,7,11,12	10	
3	2,4,6,7,11	11	8,9,10
4	1,2,3,7,11,12	12	1,2,3,4,7,11
5	3,6,8,9,10,13	13	
6		14	2,3,4,6,7,12
7	5,6	15	1,2,8,10,12,13
8		16	14,15

For the convenience of computation, the precedence relationship between design activities is able to represent with an activity-activity incidence matrix m_{ij} (1) according to the Table 1, where each element m_{ij} is defined as $m_{ij} = \begin{cases} * \ , & \text{if activity i follows by activity j} \\ \text{empty, otherwise} & \end{cases}$. The elements * in each row identify the activities that contribute to the completion of the activity indicated by the row number.

For example in matrix (1), the element * in column 11 and row 2 indicates that the completion of design activity 2 requires information to be transferred from activity 11, which means then the design activity 2 desires to be performed before design activity 11.

Design activities can be represented with a digraph G (N,A), where N is the set of nodes corresponding to the activities to be performed, and A is the set of arcs representing precedence relations. For each node i = 1 to

$$
\begin{array}{r|cccccccccccccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
1 & * & & & & & & & & & & & & & & & \\
2 & * & * & * & & & & * & & & & * & * & & & & \\
3 & & * & * & * & & * & * & & & & * & & & & & \\
4 & * & * & * & * & & & * & & & & * & * & & & & \\
5 & & & * & & * & * & & * & * & * & & & * & & & \\
6 & & & & & & * & & & & & & & & & & \\
7 & & & & & * & * & * & & & & & & & & & \\
8 & & & & & & & & * & & & & & & & & \\
9 & * & * & & * & & & & & * & & & * & & & & \\
10 & & & & & & & & & & * & & & & & & \\
11 & & & & & & & & * & * & * & * & & & & & \\
12 & * & * & * & * & & & * & & & & * & * & & & & \\
13 & & & & & & & & & & & & & * & & & \\
14 & & * & * & * & & * & * & & & & & * & & * & & \\
15 & * & * & & & & & & * & & * & & * & * & & * & \\
16 & & & & & & & & & & & & & & * & * & *
\end{array} \tag{1}
$$

N, define P_i is the set of arcs preceding node i, and Q_j is the set of arcs following node j. The digraph G(N,A) is then defined as strongly connected if for any two vertices i, j $\in$ N, a path from i to j exists [6].

A subset X of vertices is called a strongly connected vertex subset if for any two vertices i, j $\in$ X, there is a path from i to j in the graph, and X is contained in no other set with the same property. The subgraph generated by a strongly connected vertex subset is called a strongly connected component of the original graph, and the strongly connected cycle can be defined as follows:

IF two elements are interrelated with forward and backward from same head and tail
THEN it can be called strongly connected cycle

A project can be defined a combination of interrelated activities that must be executed in a certain order before the entire task can be completed. Design activities are interrelated in a logical sequence in the sense that some activities cannot start until others are completed. A design activity in a project is usually viewed as a job required time and resources for its completion. This concept can be applied to a project that the same sequence of activities may not be repeated.

If activities could be reordered so that the final matrix is lower triangular form which all marks * are either on or below the diagonal, then the elements could be determined one at a time by proceeding in this order. However, it is not positively possible to make such an ordering easily, because typical cycles are existed in any engineering design process. Therefore, the proposed algorithm focuses on the sequence not only groups of activity but also design activities in each group of activity in terms of project time duration.

Priority rules for grouping and critical path method can be applied to the entire arrow diagram that gives a graphic representation of the interdependencies between the decomposed groups of design activity. The triangualrization algorithm generally considers two cases of precedence relationship of each design activity as follows:

- design activity has successor and predecessor from same design activity
- design activities have a cycle interrelationship.

3. Algorithm and Priority Rules

Algorithm

Step 0. Begin with the given activity incidence matrix.

Step 1. IF there are diagonal elements m_{ii} with the corresponding $P_i=1$,
THEN place them in the most upper left corner of the structured matrix;
ELSE go to step 2.

Step 2. IF there are diagonal elements m_{ii} with the corresponding $Q_i=1$,
THEN place them in the most right lower corner of the structured matrix;
ELSE go to step 3.

Step 3. Apply the strong connected component algorithm to the unclustered elements m_{ii} in order to find the loop.

Step 4. IF there is more than one loop,

THEN sequence then beginning with the loop with the smallest P_i,

AND place them in the most upper left corner of the structured matrix.

IF some loops include identical activities,

THEN select the loop with the largest number of activities.

Step 5. Select the unclustered activities as one block and place them in the middle of the structured matrix.

IF all clustered blocks make a triangular matrix,

THEN stop;

Otherwise, go to step 3.

Priority Rule 1

IF there are number of elements with P_i=1 or Q_i=1,

THEN it does not need to consider the sequence of each elements,

AND it can be one group of activity with all the elements with P_i=1 or Q_i=1.

Priority Rule 2

IF there is no more elements with P_i=1 or Q_i=1,

AND the interrelationship between elements shows directed acyclic graph,

THEN put the largest strongly connected cycle as one group of activity.

Priority Rule 3

IF there are number of cycles among elements,

THEN search the cycle while belongs as many as elements,

AND group them in a group of activity and put it the most left corner of the substructured matrix, and go to step 2.

Priority Rule 4

IF no two activities can be identified by the same head and tail events,

THEN a situation may be advised when two or more activities can be performed concurrently.

4. Examples

4.1 Activity-Activity Incidence Matrix

Consider the activity-activity incidence matrix (2) for illustrating the algorithm.

Step 0. Begin with the activity-activity incidence matrix.

$$\begin{array}{c|cccccccccccccccc|c}
 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 & P_i \\
\hline
1 & * & & & & & & & & & & & & & & & & 1 \\
2 & * & * & * & & & & * & & & & * & * & & & & & 6 \\
3 & & * & * & * & & * & * & & & & * & & & & & & 6 \\
4 & * & * & * & * & & & * & & & & * & * & & & & & 7 \\
5 & & & * & & * & * & & * & * & * & & & * & & & & 7 \\
6 & & & & & & * & & & & & & & & & & & 1 \\
7 & & & & & * & * & * & & & & & & & & & & 3 \\
8 & & & & & & & & * & & & & & & & & & 1 \\
9 & * & * & & * & & & & & * & & & * & & & & & 5 \\
10 & & & & & & & & & & * & & & & & & & 1 \\
11 & & & & & & & & * & * & * & * & & & & & & 4 \\
12 & * & * & * & * & & & * & & & & * & * & & & & & 7 \\
13 & & & & & & & & & & & & & * & & & & 1 \\
14 & & * & * & * & & * & * & & & & & * & & * & & & 7 \\
15 & * & * & & & & & & * & & * & & * & * & & * & & 7 \\
16 & & & & & & & & & & & & & & * & * & * & 3 \\
\hline
Q_i & 6 & 7 & 6 & 5 & 2 & 5 & 6 & 4 & 3 & 4 & 5 & 6 & 3 & 2 & 2 & 1 &
\end{array} \qquad (2)$$

Step 1. IF there are diagonal elements m_{ii} with the corresponding P_i=1,

THEN place them in the most upper left corner of the structured matrix;

ELSE go to step 2.

Step 2. IF there are diagonal elements m_{ii} with the corresponding Q_i=1,

THEN place them in the most right lower corner of the structured matrix;

ELSE go to step 3.

$$\begin{array}{c|cccccccccccccccc|c}
&1&6&8&10&13&2&3&4&5&7&9&11&12&14&15&16&P_i\\
\hline
1&*&&&&&&&&&&&&&&&&1\\
6&&*&&&&&&&&&&&&&&&1\\
8&&&*&&&&&&&&&&&&&&1\\
10&&&&*&&&&&&&&&&&&&1\\
13&&&&&*&&&&&&&&&&&&1\\
2&*&&&&&*&*&&&*&&*&*&&&&6\\
3&&*&&&&*&*&*&&*&&*&&&&&6\\
4&*&&&&&*&*&*&&*&&*&*&&&&7\\
5&&*&*&*&*&&*&&*&&*&&&&&&7\\
7&&*&&&&&&&*&*&&&&&&&3\\
9&*&&&&&*&&*&&&*&&*&&&&5\\
11&&&*&*&&&&&&&*&*&&&&&4\\
12&*&&&&&*&*&*&&*&&*&*&&&&7\\
14&&*&&&&*&*&*&&*&&&*&*&&&7\\
15&*&&*&*&*&*&&&&&&&*&&*&&7\\
16&&&&&&&&&&&&&&*&*&*&3\\
\hline
Q_i&6&5&4&4&3&7&6&5&2&5&3&5&6&2&2&1&
\end{array}\qquad (3)$$

Step 3. Apply the strong connected component algorithm [13] to the unclustered elements m_{ii} in order to find the loop.

$$\begin{array}{c|cccccccccccccccc|c}
&1&6&8&10&13&2&3&4&5&7&9&11&12&14&15&16&P_i\\
\hline
1&*&&&&&&&&&&&&&&&&1\\
6&&*&&&&&&&&&&&&&&&1\\
8&&&*&&&&&&&&&&&&&&1\\
10&&&&*&&&&&&&&&&&&&1\\
13&&&&&*&&&&&&&&&&&&1\\
2&*&&&&&*&*&&&*&&*&*&&&&6\\
3&&*&&&&*&*&*&&*&&*&&&&&6\\
4&*&&&&&*&*&*&&*&&*&*&&&&7\\
5&&*&*&*&*&&*&&*&&*&&&&&&7\\
7&&*&&&&&&&*&*&&&&&&&3\\
9&*&&&&&*&&*&&&*&&*&&&&5\\
11&&&*&*&&&&&&&*&*&&&&&4\\
12&*&&&&&*&*&*&&*&&*&*&&&&7\\
14&&*&&&&*&*&*&&*&&&*&*&&&7\\
15&*&&*&*&*&*&&&&&&&*&&*&&7\\
16&&&&&&&&&&&&&&*&*&*&3\\
\hline
Q_i&6&5&4&4&3&7&6&5&2&5&3&5&6&2&2&1&
\end{array}\qquad (4)$$

According to priority rule 1, three groups of activity has been formed such as GA-1={1,6,8,10,13}, GA-2={16}, and unidentified GA-u={2,3,4,5,7,9,11,12,14,15}. Unidentified group of activity can be defined as one that all the elements in unstructured matrix are not ordered by sequence yet. By doing step 1 and 2 until there is no more elements left with $P_i=1$ or $Q_i=1$ in the unidentified elements, matrix (4) has been formed, since there is no elements with $P_i=1$ and elements 14 and 15 with $Q_i=1$. GA-3={14,15} can be placed at the most left corner in the unstructured matrix.

Since there is no more elements with $P_i=1$ or $Q_i=1$, consider the graph representation of interrelationship between each elements in GA-u={2,3,4,5,7,9,11,12} as a dotted lines in Figure 2 shows.

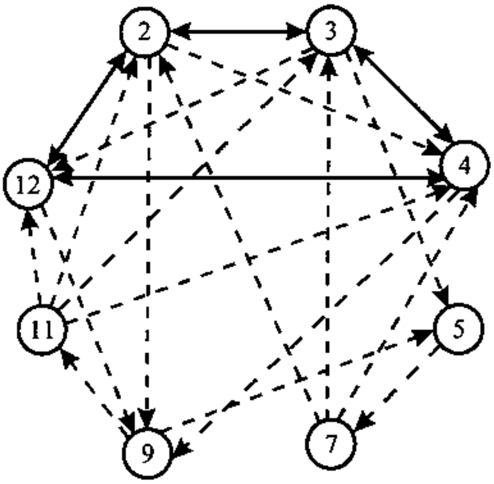
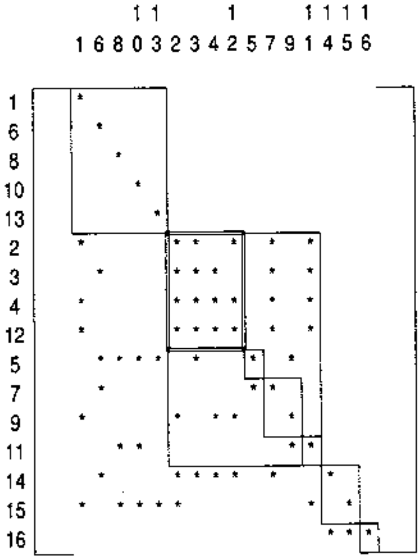


Figure 2. Interrelationship of elements in the unidentified group of activity

Step 4. IF there is more than one loop,
THEN sequence then beginning with the loop with the smallest P_i,
AND place them in the most upper left corner of the structured matrix.
IF some loops include identical activities,
THEN select the loop with the largest number of activities.

According to priority rule 2, elements {2,3,4,12} can be represented with solid lines between each element double blocked in matrix (5), and they can be one group of activity in the GA-u. Applying GA-4 and other elements 5,7,9, and 11 to priority rule 3, elements 5,7, and 9 are made one cycle.

Step 5. Select the unclustered activities as one block and place them in the middle of the structured matrix.
IF all clustered blocks make a triangular matrix, THEN stop;
Otherwise, go to step 3.



(5)

Rearranging the groups of activity and elements finally gives matrix (5) indicating four groups of activity as GA-1={1,6,8,10,13}, GA-2={16}, GA-3={14,15}, and GA-4={2,3,4,5,7,9,11,12}. The partitioned activity matrix (5) is based upon the planning the design works. In order to determine the activities to be affected by the change of a particular activity, one can analyze a particular column. Furthermore this information can be use to develop a schedule for implementing the change of design activities.

Grouping of activities can be viewed as breaking down of an original network into a number of sub-networks in terms of activity on nodes (AON) representation. Each sub-network has associated nodes and arcs. Arcs connecting two different activities which belong to different sub-networks define relationships between the sub-networks. Suppose Table 2 shows time required for each design activity.

Table 2. Project duration time for each design activity

Group	Activity	Duration	Group	Activity	Duration
1	1	10	4	11	10
1	6	15	4	2	15
1	8	10	4	3	10
1	10	15	4	4	15
1	13	10	4	12	15
2	16	15	4	5	10
3	14	10	4	7	10
3	15	15	4	9	15

Figure 3 shows that each of the four groups of activity has precedence structure, and the local schedule which has the makespan as 200 with the critical path as {GA-1}-{GA-4}-{GA-3}-{GA-2}. The global schedule generated by critical path method for the entire network notes that the analysis of relationship between groups of activity is needed in case of moderate and complicated decomposition, although there is no need to do for the ideal decomposition of a network.

However, all groups of activity except GA-4 can be applied if there is unrestricted resources for those designs according to priority rule 4. Figure 4 shows the formation process of all elements in GA-u, which has the critical role in the scheduling such as bottleneck process.

Comparing to the local makespan 200, the total global makespan produced by the proposed algorithm is 105

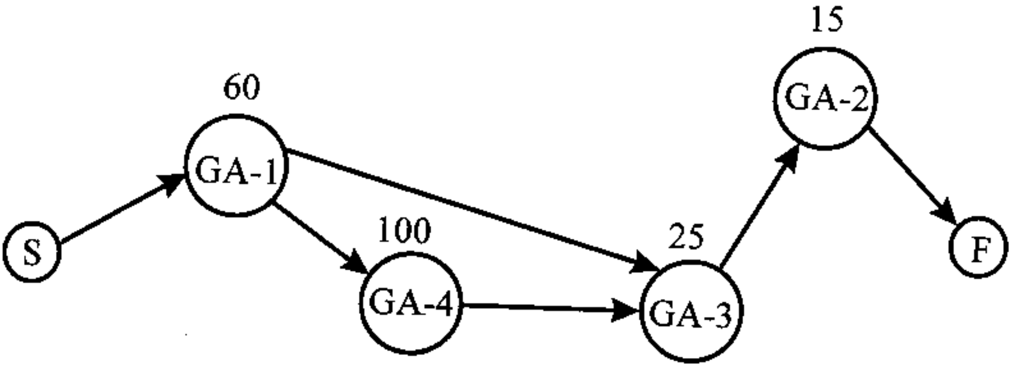


Figure 3. Four groups of activity and sequence relationship
(S: starting point, F: finished point)

which is mostly dependent upon the unidentified group of activity, which has the strongly connected cycle relationship between each elements. The proposed one can put the sequences not only each groups of activity, but also each elements in each GA-i's based upon the priority rules, which can make the design grouping problem simple, and scheduling problem concurrently.

The matrix (6) resulted from Kusiak' s algorithm described below [5] shows that the global makesapn is 125 with a given same example, which shows that the

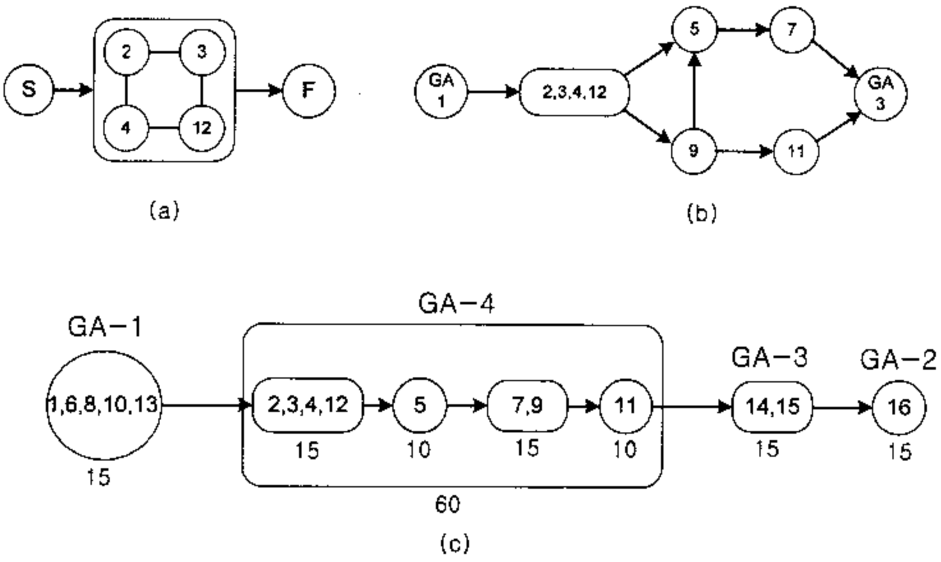


Figure 4. Formation of GA-4 and makespan

(a) priority rule 2 (b) interrelationship of all elements in GA-4
(c) makespan of the each Groups

makespan of the proposed algorithm 105 is better results than that of Kusiak's 125 [5].

Since Kusiak's algorithm decomposes the design activities in an inefficient manner, of which has the more unstructured elements as a nonzero entries out of diagonal blocks. The possible way to handle with those elements can be grouped together as one group of design activities, which can have the lower grouping efficiency.

The main reason that Kusiak's algorithm has worse makespan than the proposed here is that the Kusiak's algorithm does not specify the elements in a cycle (see step 5), while the proposed here specifies the strongly connected components in a cycle for more decomposition. The other reason can be lied on the difference between sorting rules.

Step 0. Begin [with the initial sequence of the activities (1, 2, 3, ..., m)]

Step 1. End the algorithm if all the vertices are underlined. Identify an activity which is an origin activity(OA) or a destination activity (DA). Go to Step 5 if neither an OA nor a DA is found.

Step 2. Apply the Sorting rule to the activity identified in Step 1.

Step 3. Underline the activity identified in Step 1.

Step 4. Delete the row and column associated with the underlined activity (see Step 1) from the incidence matrix, and go to Step 1.

Step 5. Find a cycle.

Step 6. Merge all the activities in the cycle into activity.

Step 7. Update the corresponding rows and columns in the incidence matrix and go to Step 1.

Sorting Rule

IF the activity is an origin activity(OA), move it to the most left position in the sequence of the activities that are not underlined.

IF the activity is a destination activity(DA), move it to the most right position in the sequence of the activities that are not underlined.

In concurrent engineering, an attempt is made to consider the design constraints simultaneously. This results in reduction of the duration of the design project, cost savings, and better quality of the final design of the unidentified groups of activity in terms of decomposition.

$$\begin{array}{c|cccccccccccccccc|}
 & 1 & 6 & 8 & 10 & 13 & 2 & 3 & 4 & 11 & 12 & 5 & 7 & 9 & 14 & 15 & 16 \\
\hline
1 & * & & & & & & & & & & & & & & & \\
6 & & * & & & & & & & & & & & & & & \\
8 & & & * & & & & & & & & & & & & & \\
10 & & & & * & & & & & & & & & & & & \\
13 & & & & & * & & & & & & & & & & & \\
2 & * & & & & & * & * & & * & * & & * & & & & \\
3 & & * & & & & * & * & * & * & & & * & & & & \\
4 & * & & & & & * & * & * & * & * & & * & & & & \\
11 & & & * & * & & & & & * & & & & * & & & \\
12 & * & & & & & * & * & * & * & * & & * & & & & \\
5 & & * & * & * & * & & * & & & & * & & * & & & \\
7 & & * & & & & & & & & & * & * & & & & \\
9 & * & & & & & * & & * & & * & & & * & & & \\
14 & & * & & & & * & * & * & * & & * & & & * & & \\
15 & * & & * & * & * & * & & & & & & & * & & * & \\
16 & & & & & & & & & & & & & & * & * & * \\
\hline
\end{array} \qquad (6)$$

However, the concurrent product design gives rise to a large scale project which might be too difficult to manage as a whole. Although the management task of the entire project is simplified due to the project decomposition, an integrated system which coordinates and analyses activities in groups is required.

4.2 Parameter-Formula Incidence Matrix

The system is allowed to perform analysis regarding the type of decomposition and the precedence constraints between groups with some priority rules. In order to illustrate that, two cases such as parameter-formula incidence matrix and parameter-parameter incidence matrix are considered. A same example of torsion bar spring in Figure 5 is considered.

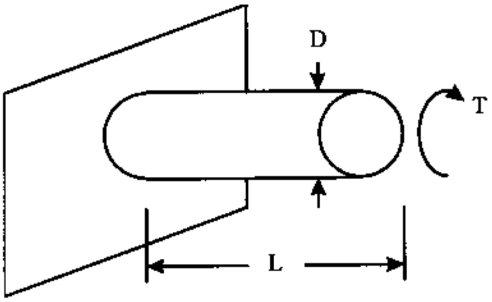


Figure 5. A torsion bar spring

The following formulas can be found in a design handbook as Table 3 shows, and the final solution matrix of the following two cases result from the proposed algorithm.

Assuming that the value of parameters G, T, L, D and n are known, then determine:

- whether it is possible to obtain values of parameters θ, τ, K, S, J, and V.
- whether the six formulas given are sufficient for determining the parameters
- the sequence of applying the equations selected for computation

The functional relationship between parameter and formulas can be represented with the bipartite graph. Applying the triangularization algorithm to the initial parameter-formula incidence matrix, the matrix (7) is shown which has a lower triangular form, providing some interesting insights into the design problem.

First of all, each of the parameters J, V, θ, and τ can be calculated in any order using the corresponding formulas, see columns of matrix (7), in any order. Secondly, each of the two remaining formulas f_5 and f_6 involves two and three parameters, respectively. However, parameter-formula such as (θ, f_5), (θ, f_6), and (τ, f_6) correspond to the nonzero entries in matrix (7) that do not belong to the diagonal blocks.

Table 3. Formulas for torsion bar spring

Description	Symbol	Formula
Volume	f_1	$V = \pi D^2 L / 4$
Stress	f_2	$\tau = 16Tn / \pi D^3$
Polar moment of inertia	f_3	$J = \pi D^4 / 32$
Twist angle	f_4	$\theta = 32TL / \pi G D^4$
Stiffness	f_5	$K = T / \theta$
Stress rate	f_6	$S = \tau / \theta$

where,

G = shear modulus of bar (psi)
τ= shear strength of bar material (psi)
T = torque on bar (lb-in)
K = torsional stiffness of bar (lb-in/rad)
D = diameter of bar (in)
θ = angular twist in bar (rad)
V = volume (in^3)
n = factor of safety for bar
S = stress rate in bar (psi / rad)
J = polar moment of inertia (in^4)
L = length of bar (in)

$$\begin{array}{cc|cccccc|} & & \multicolumn{6}{c}{\text{Formula}} \\ & & f_3 & f_1 & f_5 & f_6 & f_4 & f_2 \\ \hline & J & * & & & & & \\ & V & & * & & & & \\ \text{Parameter} & K & & & * & & & \\ & S & & & & * & & \\ & \theta & & & * & * & * & \\ & \tau & & & & * & & * \\ \hline \end{array} \tag{7}$$

4.3 Parameter-Parameter Incidence Matrix

The proposed algorithm can be applied to the parameter-parameter incidence matrix in a same manner. The precedence relationship among parameters is established as shown in Table 4.

Table 4. Precedence relationship among parameters

Parameter	Preceding Parameter
G	None
τ	T, D
T	τ
L	θ
D	τ
S	τ, θ
θ	L, G

The digraph is presented in the Figure 6. The matrix (8) is finalized according to the proposed algorithm, which has GA-1={G}, GA-2={τ,T,D}, GA-3={θ,L}, and GA-4={S}. Therefore, the sequence of the parameter-parameter incidence matrix has been done.

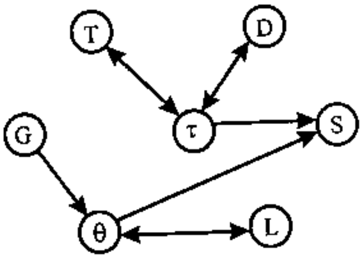


Figure 6. Digraph of relationship among parameters

$$\begin{array}{cc|ccccccc|} & & \multicolumn{7}{c}{\text{Parameter}} \\ & & G & \tau & T & D & \theta & L & S \\ \hline & G & * & & & & & & \\ & \tau & & * & * & * & & & \\ & T & & & * & * & & & \\ \text{Parameter} & D & & * & & * & & & \\ & \theta & * & & & & * & * & \\ & L & & & & & * & * & \\ & S & & * & & & * & & * \\ \hline \end{array} \tag{8}$$

5. Discussion

The study presented here can be applied to organize design activities for effective planning of the design process. The term mark * has been referred to any nonblank entry in the precedence matrices. If the variables could be reordered so that the matrix is lower triangular form, all marks are either on or below the diagonal, then proceeding in this order, the variables could be determined one at a time. As each variable is determined, all its required predecessors would be to the left of the diagonal and thus already known.

However, because of the circuits typical of any engineering design, it is not positively possible to make such an order. Therefore, the variables can be reordered to confine the marks in the matrix to appear either below the diagonal or within square blocks on the diagonal by a process known as proposed.

Once estimates are made of how many times each of the blocks are to be iterated and how long the tasks are to take in each iteration, a critical path scheduling was developed in empirical example. The design structure does not replace critical path, but provides the preliminary analysis required before a critical path scheduling for engineering can be developed.

Hence, the decomposition of design activities discussed has a number of advantages over the traditional project management tools. Perhaps the most important one is that it explicitly takes into account the structure of the design problem by allowing more detailed and specific analysis to be made. Future research will consider additional factors such as cost, process time, however, few advantages of this study can be as follows.

(a) Project decomposition allows to determine a potential groups of activity that might be scheduled simultaneously.
(b) The project scheduling and management is simplified because the management of group of activity focuses on the problems within the groups.
(c) Project decomposition creates an environment for improvement of effectiveness and efficiency of the design project.

The following statement can be further discussed.
(a) For the ideal decomposition of a system or design, each element should have completely to be independent.
(b) In terms of design and scheduling concurrency, the better quality of final design can be achieved. Although the proposed algorithm tries to decompose considering the sequence of the elements and groups of activity, it may deserve one difficulty problem for solving the cyclic interrelationship between each element.
(c) In real application of the design decomposition, it may not be possible to break out or dispense some design activities positively. Few more points are referred to critical control variables such as priority rules for grouping methodology, deletion or addition of design activities given by some numerical weight, and dummy design activity making cyclic form as one sub-system in order to make cyclic representation to acyclic form between design elements.

References

[1] Altenhof, J., Fabrycky, W., Feng, M., and Mitchiner, R., "Concurrent Mechanical System Design: A Computer-Aided Demonstration", Concurrent Engineering of Mechanical Systems, The 1990 ASME Design Technical Conference-16th Design Automation Conference, Chicago, Illinois, De-Vol.22, (1989), pp.103-109.

[2] Ancoca, D.G., and Caldwell, D.E., "Demography and Design: Predictors of New Product Team Performance", Working Paper, MIT Sloan School of Management, No.3078-89 (1989).

[3] Bucciareelli, L.L., "An Ethnographic Perspective on Engineering Design", Design Studies, Vol.9, No.3 (1988), pp.159-168.

[4] French, M., Conceptual Design for Engineers, London-:Springer-Verlag, 1985.

[5] Kusiak, A., Reengineering business, product development, and manufacturing processes, 4th IFAC Workshop on intelligent Manufacturing System, 1997.

[6] Kusiak, A., and Park, K., "Concurrent Engineering: Decomposition and Scheduling of Design Activities", International Journal of Production Research, Vol.28, No.10 (1990), pp.1883-1990.

[7] Minieka, E., Optimization Algorithms for Networks and Graphs, Mercel Dekker Inc., New York, 1978.

[8] Nevins, J.L., and Whitney, D.E., Concurrent Design of Products and Process, New York: McGraw-Hill, 1989.

[9] O'Grady, P., Ramers, D., and Bolsen, J., "Artificial Intelligence Constraints Nets Applied to Design for Economic Manufacture", Computer Integrated Manufacturing, Vol.1, No.4 (1988), pp.204-209.

[10] Quinn, J.R., "Managing Innovation: Controlled Chaos", Harvard Business Review, May-June (1985), pp.73-84.

[11] Schrage, D., McConville, J., Martin, C., and Craig, J., "An Example of Concurrent Engineering Principles Applied to the Preliminary Design of a Light

Commercial Utility Helicopter", Proceeding of the First Annual Symposium on Mechanical System Design in a Concurrent Engineering Environment, Iowa City, Vol.1, (1989), pp.513-538.

[12] Steven D. Eppinger, Daniel E. Whitney, Robert P. Smith, and David A. Gebala, Organizing the tasks in complex design projects, Massachusetts Institute of Technology Cambridge, Massachusetts.

[13] Tarjan, R., "Depth-First Search and Linear Graph Algorithms", SIAM Journal of Computing, Vol.1, No.2 (1972), pp.140-160.

98년 2월 최초 접수, 98년 4월 최종 수정