The Design Structure Matrix (DSM)

Sequencing a DSM

Sequencing is the reordering of the DSM rows and columns such that the new DSM arrangement does not contain any feedback marks, thus transforming the DSM into an upper triangular form. For complex engineering systems, it is highly unlikely that simple row and column manipulation will result in an upper triangular form. Therefore, the analyst's objective changes from eliminating the feedback marks to moving them as close as possible to the diagonal (this form of the matrix is known as block triangular). Equally, it is possible to learn about what elements of the system might possibly have to be reworked (e.g. split into two elements or perhaps removed) to achieve a better process architecture.

There are several approaches used in DSM sequencing. However, they are all similar with a difference in how they identify cycles (loops or circuits) of coupled elements. All sequencing algorithms proceed as follows:

1. Identify system elements (or tasks) that can be determined (or executed) without input from the rest of the elements in the matrix. Those elements can easily be identified by observing an empty column in the DSM. Place those elements to the left of the DSM. Once an element is rearranged, it is removed from the DSM (with all its corresponding marks) and step 1 is repeated on the remaining elements.

2. Identify system elements (or tasks) that deliver no information to other elements in the matrix. Those elements can easily be identified by observing an empty row in the DSM. Place those elements to the right of the DSM. Once an element is rearranged, it is removed from the DSM (with all its corresponding marks) and step 2 is repeated on the remaining elements.

3. If after steps 1 and 2 there are no remaining elements in the DSM, then the matrix is completely partitioned; otherwise, the remaining elements contain information circuits (at least one).

4. Determine the circuits by one of the following methods:

- **Path Searching**
- **Powers of the Adjacency Matrix Method**

  (http://www.dsmweb.org/en/understand-dsm/technical-dsm-tutorial/partitioning.html#c329)
- **The Reachability Matrix Method**

  (http://www.dsmweb.org/en/understand-dsm/technical-dsm-tutorial/partitioning.html#c330)

- **Triangularization Algorithm**

  (http://www.dsmweb.org/en/understand-dsm/technical-dsm-tutorial/partitioning.html#c331)
- Tarjan’s Depth First Search Algorithm (Tarjan 1972)

5. Collapse the elements involved in a single circuit into one representative element and go to step 1.

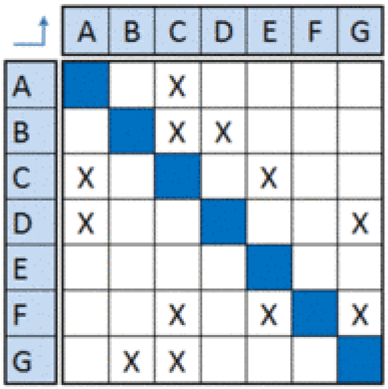
Identify loops by path Searching

Compare *Gebala, D. A. and Eppinger, S. D., Methods for analyzing design procedures, in Proceedings of 3rd International ASME Conference on Design Theory and Methodology, 1991, pp. 227-233*

In path searching, information flow is traced either backwards or forwards until a task is encountered twice. All tasks between the first and second occurrence of the task constitute a loop of information flow. When all loops have been identified, and all tasks have been scheduled, the sequencing is complete and the matrix is in block triangular form.

The figure, below, is a simple example. The path searching partition proceeds as follows:

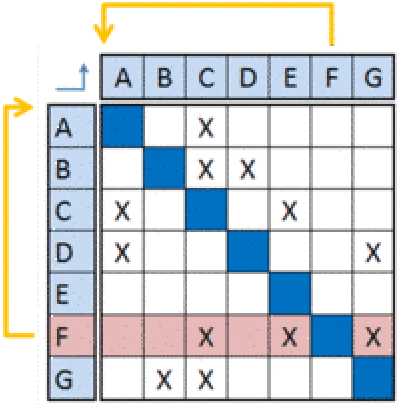
1. The unpartitioned matrix is in its original order.



	A	B	C	D	E	F	G
A	■		X				
B		■	X	X			
C	X		■		X		
D	X			■			X
E					■		
F			X		X	■	X
G		X	X				■

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild16.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=d4db1b21568de7ae177022819422d1fe)

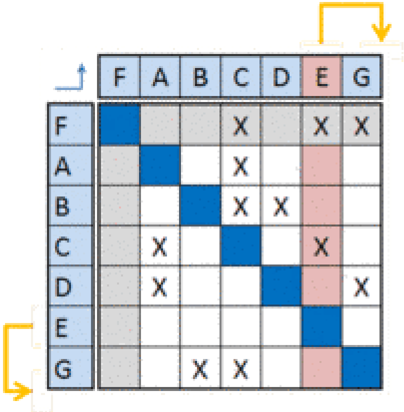
2. Task F does not depend on information from any other tasks as indicated by an empty column. Schedule task F first in the matrix and remove it from further consideration.



	A	B	C	D	E	F	G
A	■		X				
B		■	X	X			
C	X		■		X		
D	X			■			X
E					■		
F			X		X	■	X
G		X	X				■

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild17.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=2258cd84b0c4281b5e53bca453afe3b4)

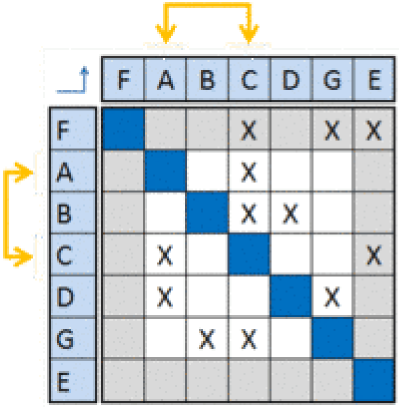
3. Task E does not provide information to any tasks in the matrix as indicated by an empty row. Schedule task E last in the matrix and remove it from further consideration.



	F	A	B	C	D	E	G
F	■			X		X	X
A		■		X			
B			■	X	X		
C		X		■		X	
D		X			■		X
E						■	
G			X	X			■

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild18.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=b357df452a29f7dbd8aca0c23b3b1ef2)

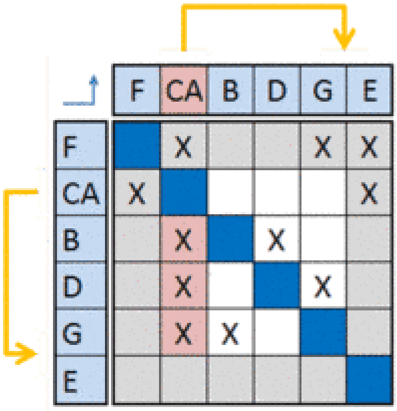
4. Now, no tasks have empty rows or columns. A loop exists and can be traced starting with any of the remaining tasks. In this case, we select task A (arbitrary) and trace its dependence on task C. Task C is simultaneously dependent upon information from task A. Since task A and task C are in a loop, collapse one into the other and represent them in a single, composite task (i.e. task CA).



	F	A	B	C	D	G	E
F				X		X	X
A				X			
B				X	X		
C		X					X
D		X				X	
G			X	X			
E							

[(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild19.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=36e3a3768db7b3354a4549f8faab1507)](http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild19.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=36e3a3768db7b3354a4549f8faab1507)

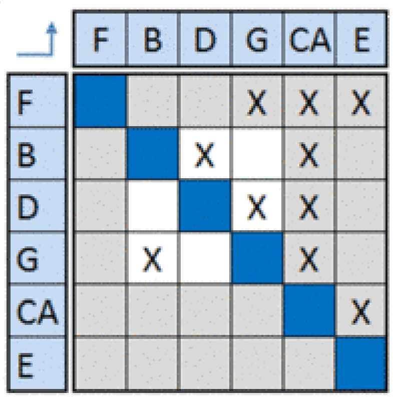
5. Task CA has an empty column indicating that it is not part of any other loop. Schedule it last and remove it from further consideration.



	F	CA	B	D	G	E
F		X			X	X
CA	X					X
B		X		X		
D		X			X	
G		X	X			
E						

[(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild20.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=4c782e42da53dd88a6d557508afa46ec)](http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild20.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=4c782e42da53dd88a6d557508afa46ec)

6. Trace dependency starting with any unscheduled task: task B depends on task G which depends on task D which depends on task B. This final loop includes all the remaining unscheduled tasks.

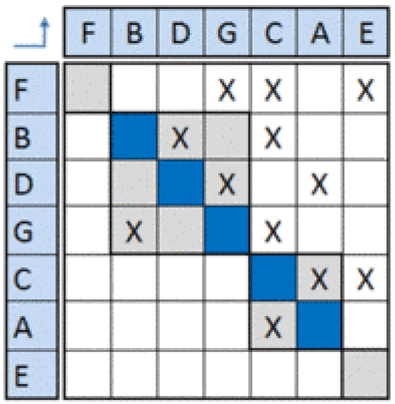


	F	B	D	G	CA	E
F				X	X	X
B			X		X	
D				X	X	
G		X			X	
CA						X
E						

[(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild21.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href](http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild21.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href)

%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=7e5bdf5ea12cd20b0c128a3f9a550e9c)

7. The final partitioned matrix.



	F	B	D	G	C	A	E
F				X	X		X
B			X		X		
D				X		X	
G		X			X		
C						X	X
A					X		
E							

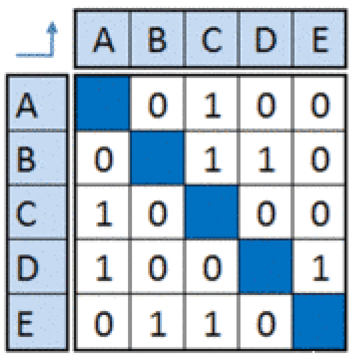
(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild22.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=91b57e1103dd1c973d3cbdaee2b9aff8)

Powers of the Adjacency Matrix Method

The Adjacency matrix is a binary DSM where an empty cell is replaced with a "zero" and a non-empty cell is replaced by "one".

Raising the DSM to the n-th power shows which element can be reached from itself in n steps by observing a non-zero entry for that task along the diagonal of the matrix. For example squaring the DSM (below) shows that tasks A and C are involved in a two-step loop. Note that in the resultant square matrix, cells with a value of greater than one were replaced by a value of one. Similarly, cubing the DSM, as shown below, shows that tasks B, D and E are involved in a three-step loop. The higher powers of the DSM reveal no other loops in the system.

DSM

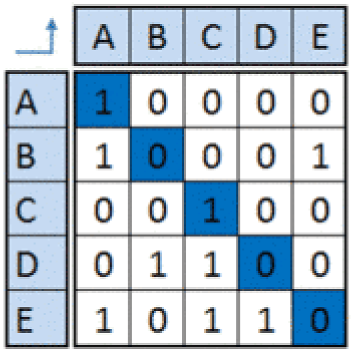


	A	B	C	D	E
A		0	1	0	0
B	0		1	1	0
C	1	0		0	0
D	1	0	0		1
E	0	1	1	0	

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild23.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href

%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=7bebf34674b6c73ed7a2afc8eba3e395)

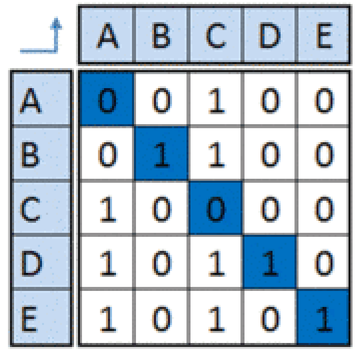
Square



	A	B	C	D	E
A	1	0	0	0	0
B	1	0	0	0	1
C	0	0	1	0	0
D	0	1	1	0	0
E	1	0	1	1	0

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild24.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=99e323782ca352d4753be268d089a9ff)

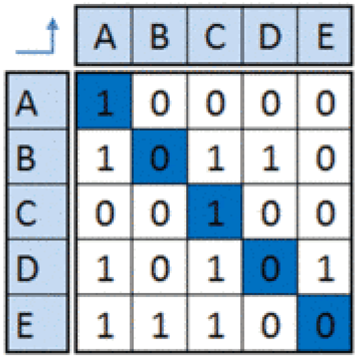
Cube



	A	B	C	D	E
A	0	0	1	0	0
B	0	1	1	0	0
C	1	0	0	0	0
D	1	0	1	1	0
E	1	0	1	0	1

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild25.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=9eb3b9fe28d1275139c7648fea7761c4)

Power 4

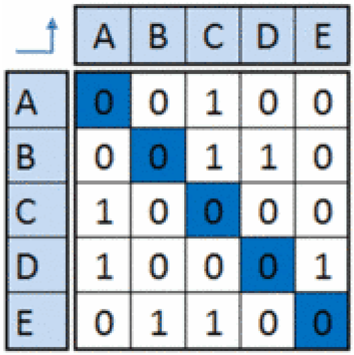


	A	B	C	D	E
A	1	0	0	0	0
B	1	0	1	1	0
C	0	0	1	0	0
D	1	0	1	0	1
E	1	1	1	0	0

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild26.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href

%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=648f4218fa4183667cf8c201f03aa76c)

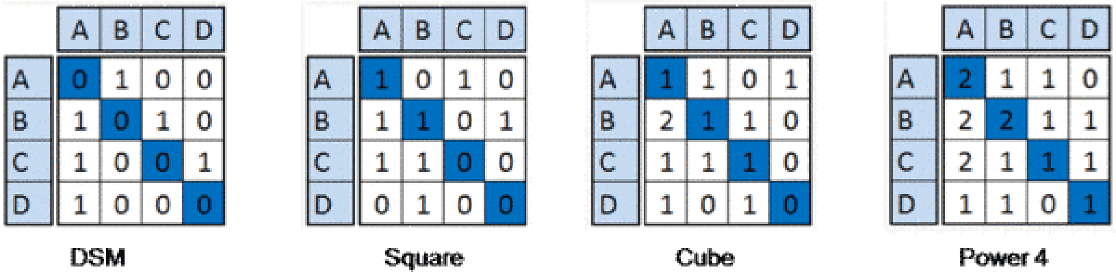
Power 5



	A	B	C	D	E
A	0	0	1	0	0
B	0	0	1	1	0
C	1	0	0	0	0
D	1	0	0	0	1
E	0	1	1	0	0

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild27.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=14dba63d8438a4c8328c1d91922dbfb0)

The following example shows, however, that this method has some limits (compare *Maurer, M.: Structural Awareness in Complex Product Design, PhD Thesis, Institute of Product Development, Munich, 2007*):



	A	B	C	D
A	0	1	0	0
B	1	0	1	0
C	1	0	0	1
D	1	0	0	0

DSM

	A	B	C	D
A	1	0	1	0
B	1	1	0	1
C	1	1	0	0
D	0	1	0	0

Square

	A	B	C	D
A	1	1	0	1
B	2	1	1	0
C	1	1	1	0
D	1	0	1	0

Cube

	A	B	C	D
A	2	1	1	0
B	2	2	1	1
C	2	1	1	1
D	1	1	0	1

Power 4

(http://www.dsmweb.org/index.php?eID=tx_cms_showpic&file=uploads%2FRTEmagicP_Bild28.png&width=800&height=600m&bodyTag=%3Cbody%20style%3D%22margin%3A0%3B%20background%3A%23fff%3B%22%3E&wrap=%3Ca%20href%3D%22javascript%3Aclose%28%29%3B%22%3E%20%7C%20%3C%2Fa%3E&md5=dfed97f4091965b93d2d0cef30217246)

The square matrix resulting from the original DSM (left side) possesses two entries on the diagonal, which indicate that the elements A and B form a feedback loop. Cubing the matrix results in three entries on the diagonal showing that the elements A, B, and C form another feedback loop. The deficiency of this approach can be seen, if the fourth power of the matrix is computed (matrix on the right hand side). Because of a feedback loop including all four elements of the initial structure (A, B, C, and D), all diagonal cells show non-empty entries; however, the diagonal cells of the elements A and B each contain the value “2”. This result is obtained because these elements form their own feedback loop (comprising only these two elements), which is run through two times in the fourth power of the matrix. This shows that comprehensive feedback loops of larger structures cannot be analyzed by this approach, because feedback loops containing the same elements cannot be distinctively traced back to their implied elements. The method only allows for determination of the existence, but not the detailed identification of feedback loops. Furthermore, this approach does not always permit the length and quantity of the existing feedback loops to be acquired. If in a matrix, feedback loops have to be identified that span six elements, feedback loops comprising two elements are processed three times and feedback loops comprising three elements are processed two times when the matrix is raised to the 6th power. All these feedback loops are added to the resulting values in the diagonal cells of the DSM, making the identification of a specific feedback loop almost impossible.

The Reachability Matrix Method

The reachability matrix is a binary DSM with the diagonal elements equal to “ONE”.

The method calls for finding a multi-level hierarchical decomposition for the matrix. The top level in this hierarchy is composed of all elements that require no input or are independent from all other elements in the matrix. Any two elements at the same level of the hierarchy are either not connected to each other or are part of the same circuit at that level. Once the top level set of elements is identified, the elements in the top level set and their corresponding from/to connections are removed from the matrix leaving us with a sub-matrix that has its own top level set. The top level set of this sub-matrix will be the second level set of the original matrix. Proceeding in this manner, all the levels of the matrix can be identified.

The steps of the method are:

1. Construct a table with four columns.

- In the first column, list all the elements in the matrix.
- In the second column, list the set of all the input elements for each row in your table. This set can easily be identified by observing an entry of “ONE” in the corresponding column in the DSM. (Include the element itself as an input).
- In the third column, list the set of all output elements for each row in your table. This set can easily be identified by observing an entry of “ONE” in the corresponding row in the DSM. (Include the element itself as an output).
- In the fourth column, list the intersection of the input and output sets for each element in your table.

2. Identify top level elements and remove them from the table. An element is in the top level hierarchy of the matrix if its input set is equal to the intersection set.

3. Go to step 1.

An excellent step-by-step example can be found in *J. N. Warfield, “Binary matrices in system modeling,” IEEE Trans. Syst., Man, Cybern., vol. 3, pp. 441-449, 1973*.

Algorithmic example

A **Triangularization algorithm** example is the one by **A. Kusiak**.