27th INTERNATIONAL DEPENDENCY AND STRUCTURE MODELING CONFERENCE, DSM 2025
HOBOKEN, NJ, USA, 24 – 26 September, 2025

Factored Dependency Structure Matrix for Representation of Multi-Connection Systems

Heekun Roh[1], Lilly Etzenbach[1], Alexia Oltramare[1], Johannes Norheim[2], Olivier L. de Weck[1] [1]Massachusetts Institute of Technology, United States
[2]University of Strathclyde, United Kingdom

Abstract: The Dependency Structure Matrix (DSM) is a simple yet powerful tool in modeling the connectivity structure between system components. A plain DSM only assumes one-to-one connections between subsystems, which often prohibits its representation power. In order to alleviate such limits, we propose the use of the Factor DSM (FDSM), which expands the DSM into a network of variables and constraints. By such expansion, we allow multiple connections between several components to be modeled, expressed in constraint equations. We compare the FDSM with other types of augmented DSMs that target multi-connection modeling. Then, we demonstrate that the FDSM is a superset of the DSM, such that a DSM can be derived from an FDSM using merge and multiplication operations. Lastly, we show the utility of the FDSM on simple operational amplifier (opamp) feedback circuit examples, where a classic DSM struggles to represent the full system architecture and behavior.

Keywords: DSM, Factor Graph, MIMO, Circuit

1 Introduction

The Dependency Structure Matrix, the Design Structure Matrix, or the Design and Structure Modeling (DSM) is a 2D matrix representation of a system architecture or an organizational process to facilitate the decision-making of the system designer. The DSM exhibits the connection structure between various elements of the system (components, processes, organizational groups, etc.) in a matrix format (Browning, 2001), which illustrates the overall architecture of the system and enables valuable analysis, such as system clustering. For instance, for the case of the component-wise DSM, the constituent components of the system are listed over both the row and column of the matrix. If there is a connection between two components A and B, the corresponding matrix element at (A, B) (and (B, A) too if the DSM is undirected) is marked as nonzero, with or without weight. If there is no connection between two components, the corresponding matrix element is unmarked or left as zero. By looking at the location of nonzeros, the connection structure of the system is revealed. By observing the closeness of the nonzeros, one can extract the cluster structure of the system.

One inherent limit of the DSM is that its representation is limited to a 2D square matrix of a single type. (e.g. only 'component' type elements on both row and column) The DSM only accepts the same list of elements for both rows and columns (hence square), and it only accepts one input and output connection between two components. In order to alleviate such limits, many extensions of the DSM were proposed. For instance, the Domain Mapping Matrix (DMM) extends the DSM by employing different row and column types (Danilovic and Browning, 2007). Similarly, the Multi Domain Matrix (MDM) enumerates all the elements with different types on the rows and columns of a square matrix (Maurer, 2007). On an MDM, one can specify connections not only from an element of type A to type A, but also connections from type A to B, A to C, C to C, and more.

In this paper, we seek to extend the representation power of architectural DSMs by incorporating the concept of variables and constraint equations. This is inspired by the factor graph approach used in signal processing (Loeliger et al., 2017) and robotics (Dellaert and Kaess, 2017), which coined the term Factor DSM (FDSM). Specifically, we list all the variables of the component on the column, and we enumerate all the connection constraints as row elements. We mark the entry where the (column) variable is part of the (row) constraint with a nonzero. As a result, a sparse binary (0 or 1) rectangular matrix is formed with all the dependency data between variables. For example, if an output A is linked to input B with a constraint C, we mark the entries at (C, A) and (C, B) as one and mark all other entries zero. By this way, the system modeler can represent (i) multiple number of connections between components (through multiple variables assigned to one component), and (ii) multiple types of connection between components (through different types of constraint equations). Since a DSM can be extracted from an FDSM by merging variables into components, the FDSM encapsulates a higher resolution version of the DSM. An FDSM preserves all the (muti)connection information of a system encoded as multiple constraints, which is not a feature of a classic DSM.

This paper is organized as follows: in Section 2, we give an overview of DSM variants that enable either multi-type or multi-connection modeling capabilities. In section 3, we introduce the construction of the FDSM with an engineering example. In section 4, we exhibit a case where a traditional DSM struggles to capture the change of connections between elements, and where an FDSM could facilitate better clustering. Section 5 summarizes and concludes the paper.

2 Survey of Multi-Connection DSMs

In this section, we present a survey of DSMs that (partially) enable multi-type and multi-connection representations of a system. Often, several flavors of DSMs are named 'DSM' without additional descriptors. Yet, in order to differentiate multiple features of the DSM and its variants, we abide by the narrow definition of the DSM, DMM, and MDM as presented below, and then introduce new flavors of DSMs with specific descriptive names on them.

The classic DSM, as presented by (Steward, 1981), (Browning, 2001), and (Browning, 2016), is a 2D matrix with an identical list of elements on its row and column, with matrix entries marking connections between two elements. Although the DSM is used for representing various types of systems, such as architectures, organizations, or processes (Browning, 2001), often its row and column elements are composed of the same type. If it is an architectural DSM, the row/column elements are system components. If it is an organizational DSM, the elements are the teams. If it is a procedural DSM, the elements are processes. DSMs can also assign scalar weights to the entries to signify the importance of each connection. By allowing asymmetry, a DSM can represent the (bi) direction of the connection.

One of the most common extensions of the DSM is the DMM. The DMM extends the DSM by allowing two distinct types of row elements and column elements (Danilovic and Browning, 2007). For example, components on the row and functions on the columns. It allows us to visualize the mapping between two domains, hence the name "Domain Mapping" Matrix. DMMs come in rectangular matrices, unlike square DSMs. Often, DMM is used without the notion of direction, only representing the undirected connection between domain elements. Yet, using multiple (transposed) DMMs can also represent directed flows.

The MDM is one of the most expressive types of DSM extensions, by allowing multiple types of elements on both the row and column. The MDM is expressed as a square matrix (thus, identical row and column element lists) with multiple types of elements present on its axes (Maurer, 2007). MDMs can be thought of as a merger of multiple DSMs and DMMs on a single matrix, which allows both the connections between the same types of elements (DSM) and between different types of elements (DMM). The typical layout of the DSM, DMM, and MDM is presented in Fig. 1.

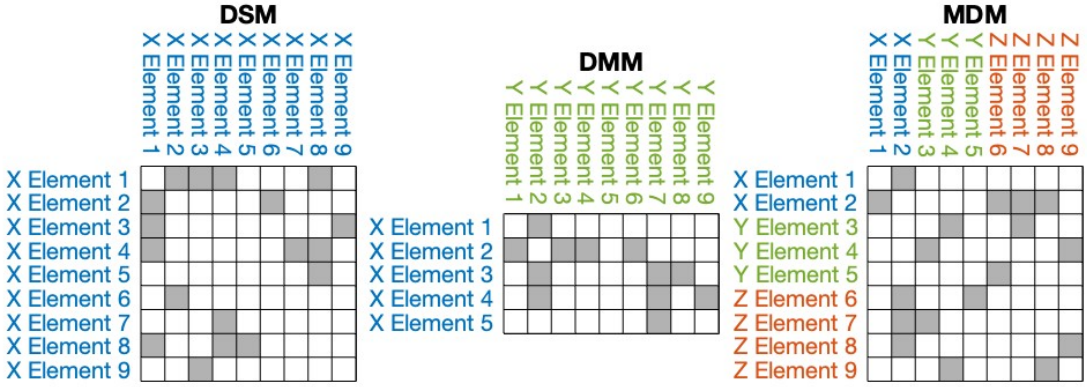

Figure 1. Typical layout of the DSM, DMM, and MDM (grey: nonzero entries)

Several extensions of the DSM were proposed to widen the application area of the classic DSM. Color coding is often used to express multiple domains. Some implementations of MDMs can be thought of as a color-coded version of DSMs, and the 1.5 domain DSM (Eppinger and Browning, 2012) color codes a 'secondary domain' on one axis to highlight connection to the secondary area of focus.

Several extension methods are suggested to add dimensions to each matrix entry. The binary-coded DSM (James, 2011) uses a binary-coded entry value that encodes multiple types of different connections between the components. The DSM with subcells (Helmer et al., 2008) uses divided subcells for each matrix entry to show five different ways of connection and their influence pattern. The High-Definition DSM (HDDSM) (Tilstra et al., 2012) uses multiple layers of DSMs to represent different types of connections on each layer.

Similar ideas can be applied to the DMM and MDM. The Functional Flow - Design Mapping Matrix (FF-DMM) (Bonjour et al., 2013) divides a single matrix entry into several matrices, similar to the subcell approach, to denote multiple functional flows from a single function. To our knowledge, there is limited publication related to the extended MDM, yet all these core techniques (color coding, subcells, entry encoding, etc) can also be used with the MDM to give it a multi-connection capability.

The proposed FDSM tackles the representation of multi-connections through a novel approach. The FDSM introduces the concept of variables and constraints into the DSM. The FDSM, a rectangular 2D matrix, is populated with constraints on the row, variables on the column, and their dependencies on the matrix entries (example in Section 3). This is a different approach from adding dimensions to the DSM (color coding, subcells, etc) which relies on the fact that most 'connections' can be modeled as constraint equations. For example, a pipe connection can be modeled as an equality constraint of output water flux from A to the input water flux to B. Multiple connections can be modeled with multiple equations on the row. Connection between multiple objects at once can be also represented on the FDSM, as long as the connection is easily converted into a mathematical equation. Hence the FDSM can represent multi-type, multi-connection systems. The summary of the features of DSM variants and FDSM is presented in Table 1.

Table 1. Comparison between various DSMs and the FDSM

	DSM	DMM	MDM	1.5 Domain DSM	Binary Coded DSM	DSM with Subcells	HDDSM	FF-DMM	FDSM
Multiple Types of Elements		O	O	(2)				O	O
Multiple Number of Connection Between Elements					O	O	O	O	O
Multiple Types of Connection Between Elements					O	O	O	O	O
Connection Between Distinct Types of Element		O	O	O				O	O
Directed Connection (In/Out)	O	(1)	O	O	O	O	O	O	(3)
Weighted Connection	O	O	O	O		O	O	O	(3)
Concept of Equation / Constraint									O
Concept of Variable									O

1) Directivity may be included using multiple DMMs, but often DMM is used without the notion of directivity.
2) Some of elements may carry secondary types, but their primary type stays the same in both row and column.
3) FDSMs can theoretically accept directivity (using two or more FDSMs) and weighted entries, but their utility is not yet established.

3 Construction of FDSM

In this section, we describe how the FDSM is inspired from factor graphs, and then show how to construct an FDSM for an electric circuit example.

3.1 Factor Graph and FDSM

Factor graphs represent the division (factorization) of a certain global function. (Loeliger, 2004) Let us assume that we have a global function f parameterized with five parameters $p_1, \ldots, p_5$, which can be factored into smaller sub-functions f_1, f_2, f_3 as below:

$$f(p_1, p_2, p_3, p_4, p_5) = f_1(p_1, p_2)\, f_2(p_3, p_4, p_5)\, f_3(p_2, p_3, p_4) \quad (1)$$

Often the global function models a probability distribution, where the factored dependency structure arises naturally. The factor graph associated with Eq. (1) describes the factorization structure of the function using a form of bipartite graph, as shown in Fig. 2. Here, a function node parameterized with a certain parameter is connected to that particular parameter node, and vice versa. For example, the function node $f_1(p_1, p_2)$, parameterized with p_1, p_2 is connected to both parameter nodes p_1, p_2.

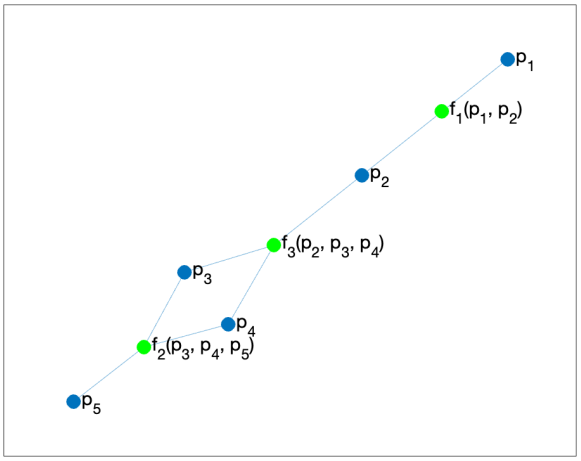

Figure 2. Factor graph representation of Eq. (1)

One important observation here is that although the factor graph originally intended to model the factorization structure of the function, the resulting factor graph does not explicitly display the (factorized) relation between the sub-functions. That being said, the exact same method can be used to display all kinds of functional relations between parameters, whether or not they are a result of a functional factorization. For instance, let's say that a system can be described five parameters $p_1, \ldots, p_5$, with three operational constraints f_1, f_2, f_3 . Even in this case, we can derive the same factor graph without a notion of 'factorization.' This extended concept of factor graph is widely used for signal processing (Loeliger et al., 2017), robotics (Dellaert and Kaess, 2017), and many more. Thus, we use such an extended concept of factor graph notation to derive the FDSM.

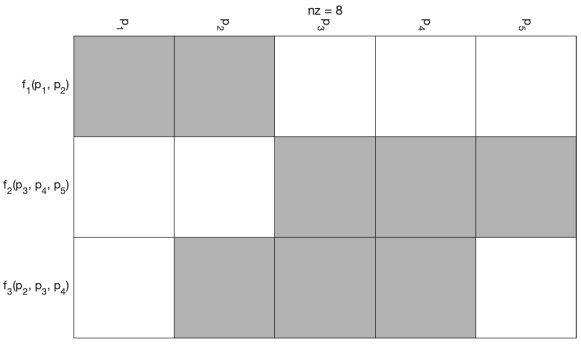

Figure 3. Matrix representation of the factor graph (grey: nonzero entries)

Since the factor graph comes in a bipartite graph form, it is natural to think about the matrix representation of such a graph. Often, an undirected bipartite graph can be converted into a single rectangular matrix, one type of node being row elements, and the other type of node being the column elements. In this paper, we only consider a binary adjacency matrix: we mark a connection with 1, and a disconnection with 0. For example, the factor graph in Fig. 2 can be converted into a matrix in Fig. 3. We seek to use such rectangular matrices from bipartite factor graphs as a basis for the FDSM. By visualizing the system in a matrix form, we can start applying all the methods developed around the concept of the DSM directly onto the FDSM.

3.2 FDSM Construction Example

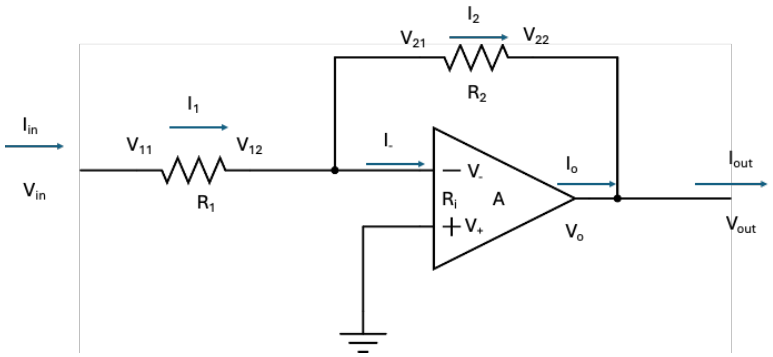

Figure 4. Opamp feedback circuit example

Let us consider an opamp feedback circuit depicted in Fig. 4. The circuit is composed of three circuit elements: resistor $R_1$and R_2, with an opamp OP_1. On a DSM, these three elements appear as fully connected: R_1 connected to R_2, R_2 connected to OP_1, and OP_1 connected to R_1. The DSM resulting from the example is given in Fig. 5. It is worth observing that some elements are interconnected through multiple ports: for example, R_2 is not only connected to OP_1 through the (-) input port (left top side of the opamp), but also to the output port (rightmost corner of the opamp). Still, on a DSM, these multiple connections are often rendered as a single (weighted) matrix entry, unless more entries are defined for all the electrical nodes

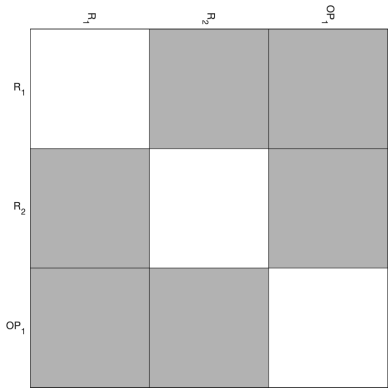

Figure 5. Opamp feedback circuit DSM (grey: nonzero entries)

On the other hand, a factor graph can be generated from the following set of governing equations:

$$V_{11} - V_{12} = I_1 R_1 \quad (2)$$

$$V_{21} - V_{22} = I_2 R_2 \quad (3)$$

$$V_- - V_+ = I_- R_i \quad (4)$$

$$V_o = A(V_+ - V_-) \quad (5)$$

$$V_{in} = V_{11} \quad (6)$$

$$I_{in} = I_1 \quad (7)$$

$$V_{12} = V_{21} \quad (8)$$

$$V_{12} = V_- \quad (9)$$

$I_1 = I_- + I_2$ *(10)*

$V_+ = 0$ *(11)*

$V_{22} = V_o$ *(12)*

$V_{22} = V_{out}$ *(13)*

$I_2 + I_o = I_{out}$ *(14)*

The equations are derived from general electric circuit principles, such as Ohm's law (Eq. (2, 3, 4)) or Kirchoff's law (Eq. (7, 10, 14)). The first four equations are governing equations for each component, and the others are generated from wired electric connections. The resulting FDSM from the governing equations is in Fig. 6. Here, multiple electric connections are rendered as multiple entries with several variables and constraints, unlike a single entry on a DSM. On an FDSM, even multiple constraints arising from a single connection are captured. For example, both the voltage condition and the current condition from a single electric wiring are fully represented with an FDSM.

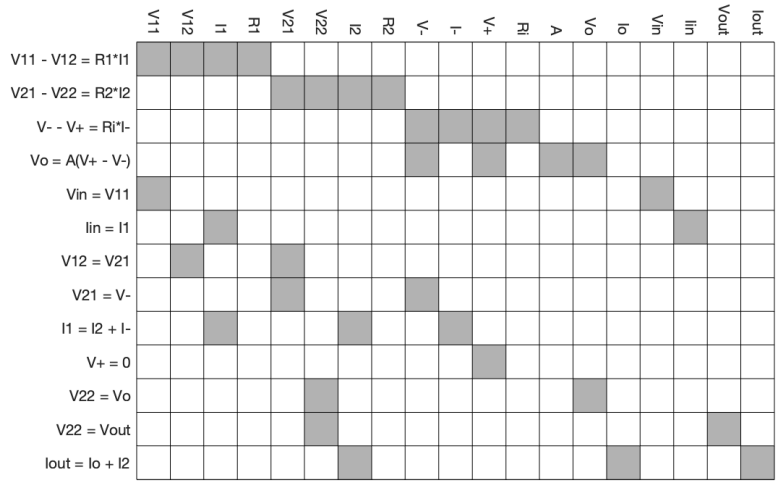

Figure 6. Opamp feedback circuit FDSM (grey: nonzero entries)

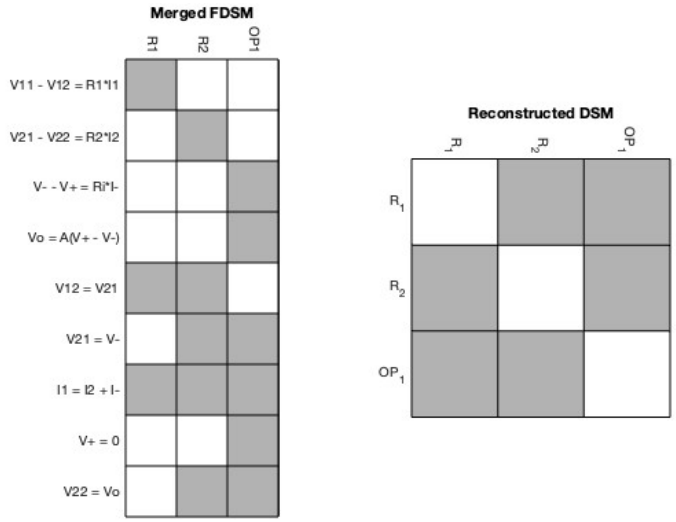

Figure 7. Merged opamp circuit FDSM and Reconstructed DSM (grey: nonzero entries)

The resulting FDSM holds all the electrical connection information in the DSM. The preceding DSM can be reconstructed from FDSM by (i) merging the variables from each components into one (and removing system variables such as V_{out}),

(ii) performing (merged FDSM)T × (merged FDSM) and removing diagonal entries. The reconstruction process is identical to the DMM to DSM conversion, except the added merge operation due to the multi-variable nature of the FDSM. The reconstructed DSM, which is identical to the original, is given in Fig. 7.

3.3 FDSM Benefits and Limitations

In terms of benefit, the FDSM holds an abundance of information about the system compared to the classic DSM. Since a DSM can be losslessly extracted from an FDSM, the FDSM can be considered as a system information superset of the DSM. Furthermore, many types of components or connections can be included in the FDSM as long as they are abstracted into the form of mathematical equations. For example, a physical connection (position constraint) and an electric connection (voltage and current constraints) can be both represented in the FDSM without any specific modification. Different types of system components, such as physical components versus organizational components, can be both represented on the FDSM without additional type setting or coloring.

The FDSM also exhibits some fundamental limitations. First, the definition of a single variable and a single constraint can be fuzzy. The DSM can suffer from fuzzy delineation of components, and the FDSM can similarly suffer from 'drawing the line'. For instance, two equal-to-zero constraints can be merged into one equality constraint with the sum of the original constraints in absolute value form. It is unclear if two binary variables encoded into one parameter vector should appear as two separate columns or one. Second, since it only records the dependency structure between variables and constraints, it cannot distinguish the constraints with different forms but the same dependency structure. For example, an affine constraint versus a nonlinear constraint with the same dependency relation cannot be distinguished on the FDSM except for their row labels.

3.4 FDSM Related Concepts in Systems Engineering

Although the FDSM was originally inspired by the factor graph notation in the signal processing domain, it is possible to find similar concepts in the domain of Systems Engineering. Donald Steward, the inventor of the term DSM, gleans over the concept of 'structural matrix,' which models a large system of equations into a dependency matrix with equations on the row, variables on the column (Steward, 1965). The structural matrix was demonstrated in the context of equation partitioning and tearing. A similar concept is also visible from the Multidisciplinary Design Optimization (MDO) domain, labeled Functional Dependence Table (FDT) (Chen et al., 2005, Tosserams et al., 2010). The FDT carries design functions on the row, and the design variables on the column. The FDT is used in MDO problem decomposition and reconstruction (Chen et al., 2005, Tosserams et al., 2010). EXtended DSM (XDSM, Lambe and Martins, 2012) is also a concept that explicitly displays variables and functions. Yet, these concepts were not utilized and examined in the context of the DSM. The FDSM, which is targeted for a (physical) system modeling, mirrors the components visible on the DSM by multiple state variables and can be converted to a DSM (Section 3.2), which may not be a feature for the other concepts. Also, we observe the FDSM under the concept of (physical) system clustering (Section 4.2), which is the key deviation from similar concepts.

4 FDSM Advantages

In this section, we give two examples demonstrating the advantages of an FDSM over a conventional DSM. First, we show the advantage of the FDSM when the connection between components is modified and the expected system behavior changes. Second, we show the benefit of the FDSM on clustering when multiple components are connected with multiple ports.

4.1 FDSM With Modified Connections

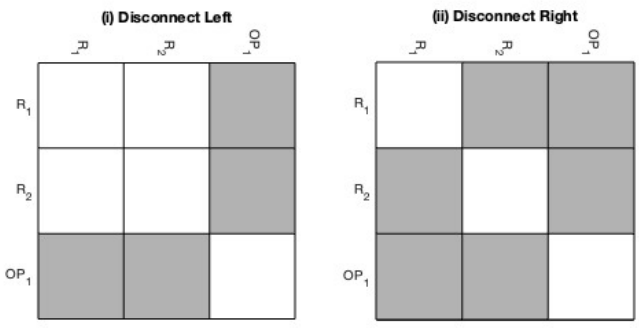

Figure 8. DSM for opamp broken feedback cases (grey: nonzero entries)

Consider the same opamp example from section 3. If the feedback connection through R_2 is broken, the opamp circuit goes into feed-forward mode, and the output voltage goes to infinity. Yet, there are two ways to break the feedback: (i) disconnect the left leg of R_2 which is connected to R_1 and OP_1, or (ii) disconnect the right leg of R_2 which is connected to OP_1 only. Either way, the expected circuit behavior is the same (infinite output), but the resulting DSMs are considerably different (Fig. 8). This can be considered as an artifact of the existence-of-connection centric approach of the DSM. The DSM considers components and connections only, but does not encode many details about connections themselves. However, for certain cases like the opamp example, the inner workings of the connection (which is modeled with variables and constraints in the FDSM) heavily govern the system characteristic rather than the existence of the connection between components.

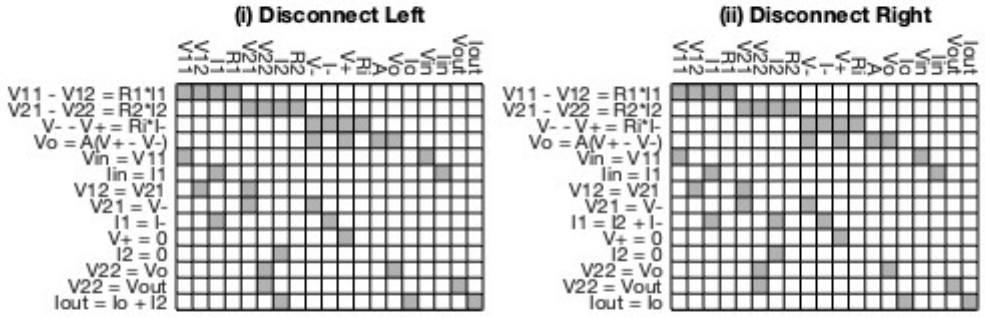

Figure 9. FDSM for opamp broken feedback cases (grey: nonzero entries)

The FDSM representation of both cases (Fig. 9) exhibits only minimal differences (2 entries out of 35 entries) and the difference captures only the change in the current constraint on the different breakage point. The overall sparsity structure of the two FDSMs are similar, signalling that the behavior of two systems must be close to each other. The exact behavior of the circuit can be further extracted from the list of constraints, too. As a consequence, we argue that the FDSM is better suited for the mathematical modeling phase of the system, contrary to the DSM which is better suited for engineering communication thanks to its simplicity.

4.2 FDSM for Cluster Analysis

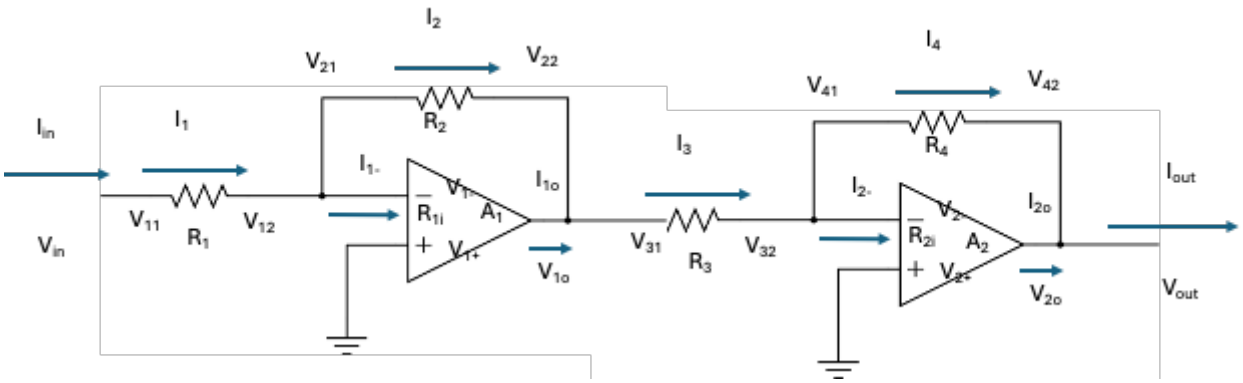

Figure 10. A double gain stage circuit

The most common analysis performed on DSMs is cluster analysis. For certain cases, the FDSM can shed light on the finer details of the multi-connection network of components, which can result in better clustering. Consider the double gain stage opamp circuit example below (Fig. 10). We illustrate that an FDSM may generate better clustering than a DSM that fits the general intuition of electrical engineers.

For electrical engineers, it is natural to see the circuit as two distinct gain stages. The first one being the group of R_1, R_2, and OP_1 with a gain of -R_2/R_1. The second stage being the group of R_3, R_4, and OP_2 with a gain of -R_4/R_3. If the representation by DSM/FDSM is sound, we can expect that the two separate gain stages would be visible as two clusters on the DSM.

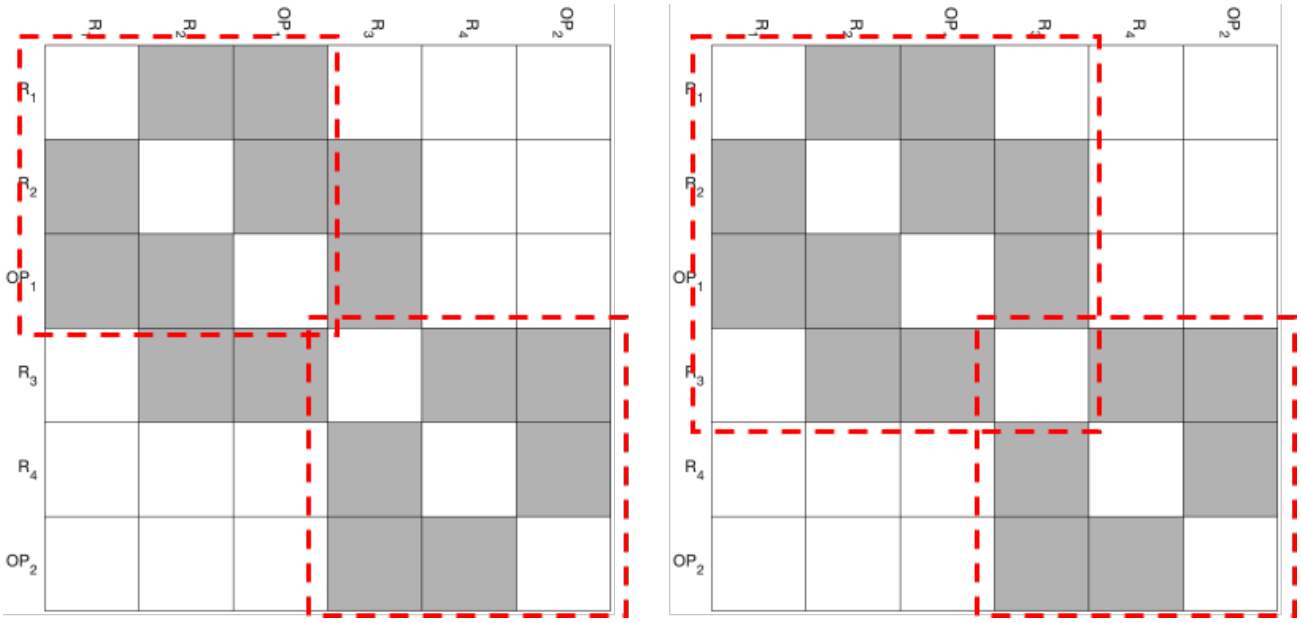

Figure 11. Clustered double gain stage circuit DSM, Left: (R_1, R_2, OP_1) - (R_3, R_4, OP_2), Right: (R_1, R_2, OP_1, R_3) - (R_3, R_4, OP_2)

Fig. 11 depicts two different clusters along six components. The left being the desired clustering result, and the right being the alternative clustering result. Although the fitness of the clustering result can vary by the figure of merit, it is worth noting that the alternative clustering (right) confines more nonzeros within the cluster boundary, thus it can be deemed as a desirable clustering result. Yet, in terms of electronic circuitry, the clustering of (R_1, R_2, OP_1, R_3) is awkwardly unnatural, since R_3 performs no role for the first gain stage (R_1, R_2, OP_1).

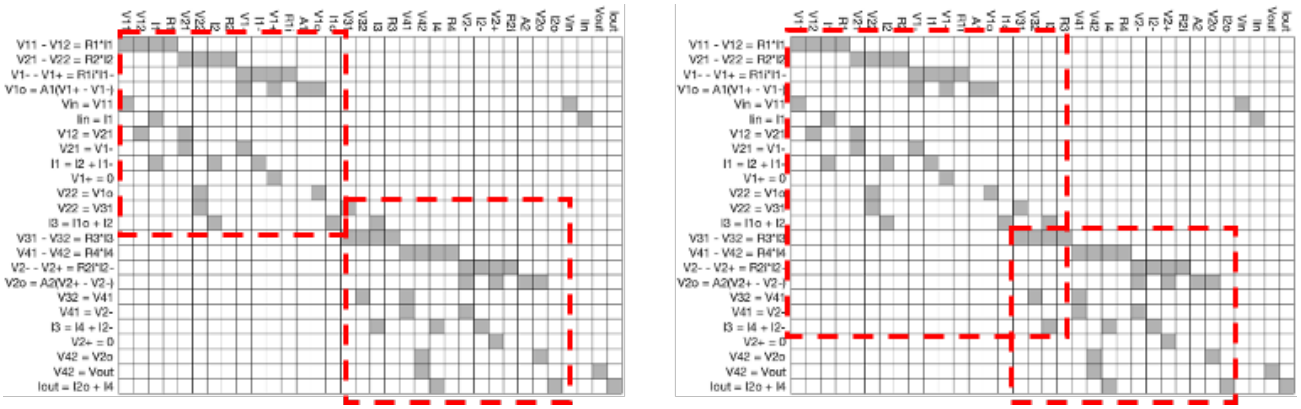

Figure 12. Clustered double gain stage circuit FDSM, Left: (R_1, R_2, OP_1) - (R_3, R_4, OP_2), Right: (R_1, R_2, OP_1, R_3) - (R_3, R_4, OP_2)

On the other hand, for the FDSM representation of the twin gain stage circuit, the desired (R_1, R_2, OP_1) by (R_3, R_4, OP_2) clusters naturally arise from the FDSM (Fig. 12, left). The concept of cluster on the FDSM is slightly different from DSM, since the rows and columns are not equivalent on an FDSM. The (R_1, R_2, OP_1, R_3) clustering (right) generates an awkwardly sparse cluster, which would not be the desired result of any clustering algorithm. Here, we considered the constraint row to be the part of the cluster if it contains any of the column variables from the cluster, thus allowing membership of a single row to multiple clusters.

Here, we can observe that the FDSM generates clusters somewhat more familiar to the engineer (at least for this particular example) than the DSM. We contend that this is due to the fact that the intuition of 'cluster' for engineers is often formed

around the functional relation of the components (which is better represented in terms of variables and constraints) rather than the concept of connections (which is better represented in terms of connections on a DSM).

5 Conclusion

In this paper, we proposed a novel approach of using the concept of constraints and variables for constructing a Factor DSM, inspired by the factor graph approach used in estimation problems. We contend that classic DSMs often only represent the existence of connections between components, but not the detailed information on the specific nature of the connections. By modeling the connections using constraints and detailing the components with multiple variables, we demonstrate that FDSMs can be useful in representing a system with multiple connections, multiple variables, and multiple constraints. We demonstrate that the exact DSM can be extracted from the FDSM using a few matrix operations, hence, FDSM encodes (undirected) DSM without loss, but with more system constraint information. We display the advantages of the FDSM over the classic DSM using two examples, and show that an FDSM could be better at robustly representing a multi-connection network of components under connection modification, and be better at displaying the cluster structure more akin to that of the engineering intuition. We argue that this is due to the fact that the intuition of linkage and cluster is often generated based on the functional role of components, which is better represented by the network of variables and constraints than the existence of (physical) connection.

References

Bonjour, E., Deniaud, S., and Micaelli, J.-P., 2013. "A method for jointly drawing up the functional and design architectures of complex systems during the preliminary system-definition phase," Journal of Engineering Design, vol. 24, pp. 305–319.

Browning, T. R., 2001. "Applying the design structure matrix to system decomposition and integration problems: a review and new directions," IEEE Transactions on Engineering Management, vol. 48, no. 3, pp. 292–306.

Browning, T. R., 2016. "Design structure matrix extensions and innovations: A survey and new opportunities," IEEE Transactions on Engineering Management, vol. 63, no. 1, pp. 27–52.

Chen, L., Ding, Z., and Li, S., 2005. "Tree-based dependency analysis in decomposition and re-decomposition of complex design problems," ASME Journal of Mechanical Design, vol. 127, no. 1, pp. 12–23.

Danilovic, M., and Browning, T. R., 2007. "Managing complex product development projects with design structure matrices and domain mapping matrices," International Journal of Project Management, vol. 25, no. 3, pp. 300–314.

Dellaert, F., and Kaess, M., 2017. Factor graphs for robot perception, Foundations and Trends® in Robotics, vol. 6, no. 1–2, pp. 1–139.

Eppinger, S. D., and Browning, T. R., 2012. Design Structure Matrix Methods and Applications, MIT Press, Cambridge, MA, USA.

Helmer, R., Yassine, A., and Meier, C., 2008. "Systematic module and interface definition using component design structure matrix," Journal of Engineering Design, vol. 21, no. 6, pp. 647–675.

James, D. H., 2011. Influence of system architecture changes on organizational work flow and application to geared turbofan engines, Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.

Lambe, A.B., Martins, J.R.R.A., 2012. "Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes," Structural and Multidisciplinary Optimization, No. 46, pp. 273–284.

Loeliger, H.-A., 2004. "An introduction to factor graphs," IEEE Signal Processing Magazine, vol. 21, no. 1, pp. 28–41.

Loeliger, H.-A., Dauwels, J., Hu, J., Korl, S., Ping, L., and Kschischang, F. R., 2007. "The factor graph approach to model-based signal processing," Proceedings of the IEEE, vol. 95, no. 6, pp. 1295–1322.

Maurer, M. S., 2007. Structural awareness in complex product design, PhD thesis, Technische Universität München, Munich, Germany.

Morelli, M. D., Eppinger, S. D., and Gulati, R. K., 1995. "Predicting technical communication in product development organizations," IEEE Transactions on Engineering Management, vol. 42, no. 3, pp. 215–222.

Steward, D. V., 1965. "Partitioning and tearing systems of equations," Journal of the Society for Industrial and Applied Mathematics: Series B, Numerical Analysis, vol. 2, no. 2, pp. 345–365.

Steward, D. V., 1981. "The design structure system: A method for managing the design of complex systems," IEEE Transactions on Engineering Management, vol. EM-28, no. 3, pp. 71–74.

Tilstra, A. H., Seepersad, C. C., and Wood, K. L., 2012. "A high-definition design structure matrix (HDDSM) for the quantitative assessment of product architecture," Journal of Engineering Design, vol. 23, no. 10–11, pp. 767–789.

Tosserams, S., Hofkamp, A.T., Etman, L.F.P. et al., 2010. "A specification language for problem partitioning in decomposition-based design optimization," Structural and Multidisciplinary Optimization, no. 42, pp. 707–723.

Contact: H. Roh, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 77 Massachusetts Avenue, Cambridge, Massachusetts, United States, (617) 460-6485, hroh@mit.edu