**Proceedings of the ASME 2019**
**International Design Engineering Technical Conferences**
**and Computers and Information in Engineering Conference**
**IDETC/CIE2019**
**August 18-21, 2019, Anaheim, CA, USA**

# DETC2019-98393

# STRUCTURAL CONSEQUENCE ANALYSIS: TOWARDS THE QUANTIFICATION OF COMPONENT CONSEQUENTIAL IMPORTANCE IN SYSTEM ARCHITECTURE DESIGN

**Hannah S. Walsh**
School of Mechanical, Industrial,
and Manufacturing Engineering
Oregon State University
Corvallis, OR, USA

**Mohammad Hejase**
Intelligent Systems Division
NASA Ames Research Center
Moffet Field, CA, USA

**Daniel Hulse**
School of Mechanical, Industrial,
and Manufacturing Engineering
Oregon State University
Corvallis, OR, USA

**Guillaume Brat**
Intelligent Systems Division
NASA Ames Research Center
Moffet Field, CA, USA

**Irem Y. Tumer**
School of Mechanical, Industrial,
and Manufacturing Engineering
Oregon State University
Corvallis, OR, USA

## ABSTRACT

*There is a major push in safety-critical systems to consider system risk early in the design process in order to avoid costly redesign later on. However, existing techniques, which may be labor-intensive and be subject to many sources of uncertainty, rely on failure mode and failure rate data, which can only be estimated in the early design phase. This paper proposes a network-based technique for assessing the consequential importance of a particular component to enable designers to consider hazards in the design of the system architecture without the use of estimated failure rates. Structural consequence analysis represents connectivity between components with a network and provides an explicit representation of risk prevention and mitigation techniques, such as redundancy. The network is augmented with a measure of the consequence of the failure of the "end" components, or sinks, which can be backpropagated through the network to compute the consequence associated with the failure of all components. Based on this consequence, designers can consider mitigation strategies, such as redundancy or increased component reliability. The approach is demonstrated in the design of an electric system to control an aileron of an unmanned aircraft system (UAS). It is found that structural consequence analysis can identify potentially important components without failure rate data, allowing designers to proactively design for risk earlier in the design process.*

## 1 INTRODUCTION

Risk prevention and mitigation often come at the cost of design elegance. For example, component redundancy is a typical method to meeting safety requirements, but can result in increased operational complexity, up-front design cost, and decreased performance in engineered systems. This is exemplified by the space shuttle, which had multiple computers that "voted" on the correct computation [1], adding complexity. Another example of redundancy adding complexity is in the case of cold standby redundancies, in which switching to the redundant system requires detecting a failure of the main system and switch-

This work was authored in part by a U.S. Government employee in the scope of his/her employment.
ASME disclaims all interest in the U.S. Government's contribution.


Copyright © 2019 ASME

ing over to the redundant system [2]. This additional complexity can increase development costs and time [3], and redundancies add cost and weight, the latter being of particular concern for aerospace systems. For these reasons it is important to consider a trade-off between risk design strategies and complexity, cost, and weight, as well as the consequence of failure. As a result, it is preferable to design inherent safety into the product, if possible, which eliminates the sources of risk, before considering costly protective design features [4, Section 7.3.3], and to allocate redundancy between components in a smart, efficient way.

In the risk-based design of engineered systems, it is often helpful to consider each individual component's contribution to overall risk. Two classical metrics that have been extensively used in literature to express a component's importance are the Statistical Importance and the Structural Importance. However, these metrics are based on the fault tree and not a full network mapping causes to consequences, which is required to fully represent the safety architecture of engineered systems [5].

In this research, an alternative approach, structural consequence analysis (SCA), is proposed for quantifying component importance based on its position in the causal network, as opposed to its direct reliability in a fault tree. The approach utilizes a directed network representation of the engineered system to propagate the consequences of sink nodes, which are components that do not relay information to any other components within the architecture, to quantify the consequence of failure of the other nodes in the network and thus obtain a measure of component importance. There are a few benefits to this approach. First, SCA combines information about differing consequences of failure for different sink nodes. Second, the network representation allows for an explicit representation of redundancy. Third, SCA provides greater expressiveness about common sources of hazards and component interactions that lead to failure than fault trees. Fourth, SCA allows designers to represent causal mechanisms of failure within engineered systems to better consider direct inherent safety in the system. A measure of structural importance is further provided with component consequence consequent metrics, enabling designers to consider changes to system structure in early design.

## 2 LITERATURE REVIEW

Before introducing the methodology, the literature will be reviewed. First, component importance measures will be reviewed, with an emphasis on fault tree metrics. Second, the network literature will be reviewed, with an emphasis on applications to system design. Third, a review is provided for representations of risk in engineered systems.

### 2.1 Component Importance Measures

Fault tree analysis is a commonly used method for risk assessment in engineering design. It shows various paths leading to a "top-event", an undesirable system state [6]. Fault trees, which are constructed using expert analysis and historical data, are used to assess whether a design meets risk requirements. Various fault tree metrics can be used to compute the importance of each component within the system architecture. Minimal cut sets, which are combinations of events that cause the top event, and the top event probability can be computed from the fault tree as well. This probability can be compared to risk requirements. If it is found that the top event probability does not meet risk requirements, the design can be updated by adding redundancy. Adding redundancy lowers the top event probability. The selection of the particular component to which redundancy is added to reduce this top event probability is of interest in this research.

Traditionally, the decision of where to add redundancy is made with the help of component importance metrics such as statistical importance and Birnbaum importance. Statistical importance, sometimes called Fussell-Vesely importance, is a measure of the contribution of cut sets containing the component of interest to the total system risk.

There are two forms of Birnbaum importance. One, the reliability importance, uses failure rates of components and one, the structural importance, does not use the actual failure rates of components (it instead assumes equal values for all component failure rates) [7, 8]. Neither version takes into account the failure rate of the component of interest itself.

Recent research has advanced and expanded upon fault tree analysis in various ways. Fault tree analysis has been augmented with fuzzy set theory to account for imprecision in event probabilities used to calculate the top event probability [9]. Sinnamon and Andrews proposed a technique for more accurately computing the top event probability in the case that there is a large number of minimal cutsets [10]. Dynamic fault trees have been used to study how temporal information about the failure of multiple components affects risk [11]. To implement a dynamic fault tree, Boudali et. al. and Dugan et. al. both used Markov chains [12, 13]. Others including Liant et. al. used a Monte Carlo approach [14]. Zhang et. al. used a Petri net based approach [15]. Binary decision diagrams have been used to efficiently compute importance measures [16]. Other fault tree extensions include dependent events [17] and repairable fault trees [18].

### 2.2 Network Representations of Engineered Systems

A network is a collection of nodes and edges. Nodes, sometimes called vertices, are the elements that make up a network. These nodes are connected by edges, sometimes called links. For the purposes of this paper, the terms network and graph are interchangeable. The difference between the two is that in a network, nodes represent something physical, whereas a graph is simply a


Copyright © 2019 ASME

mathematical representation. Networks have been used to study the world-wide web [19], various aspects of biology [20], and social networks [21, 22], among others.

Euler is credited with initiating the study of graphs by using a graph to solve the famous Königsberg bridge problem [23]. Erdős and Rényi later studied random graphs, in which all nodes have equal probability of connections [24]. Random graphs, however, are not found so often in the real world as compared to the scale-free networks studied by Barabási [23]. Graphs and networks are often analyzed using their degree distribution plot, which is a histogram of the degree, or number of connections, of all nodes; in a random graph, the degree distribution is a bell-curve shape while in a scale-free network, the degree distribution follows a power law [23].

More recently, networks have been applied to system design. Specifically, they has been used to identify critical design parameters [25–28], analyze system modularity [29], analyze the effect of design changes [30], study collaboration in design [31], predict customer preferences [32, 33], and model system architecture [34, 35]. The next section will cover in more detail how this research uses networks to model engineered systems.

### 2.3 Representations of Risk in Engineered Systems

Many tools have been developed in the literature to represent the causes, mechanisms, and consequences of failure in engineered systems. Failure Modes and Effects Analysis (FMEA) is most commonly used in design, to determine the components with the highest overall risk priority to direct design and operational decision-making towards the minimization of faults which occur in those components [4, Section 10.4] [36, Section 11.6]. FMEA is performed in a table, and as a result represents causality in a component as set of listed causes which lead to a mode (a specific change of state or behavior of the component) which in turn results in a set of listed effects.

Fault Tree Analysis (as discussed earlier) is another common approach to design for risk, and is used to identify all of the fault events which would lead to a single top-event [4, Section 10.3] [36, Section 11.6]. As a *tree structure*, Fault Tree Analysis is capable of representing an entire chain of causality, allowing one to deductively trace top-level system failure effects to individual failure mechanisms in failure mechanisms which occur in individual components or component interactions. Fault trees are often used for risk assessment to determine the overall probability of failure, given probabilities for each of the causes of failure [37, Chapter 12]. Reliability Block Diagrams (RBDs) are also used to calculate the overall reliability of a system given individual component reliabilities in a system architecture, often to represent different forms of redundancy in a system [37, Chapter 9]. As with FTA, the limitation of the RBD approach is that only one generic effect of component failure is considered–system failure.

Event Tree Analysis (ETA) is another tree structure that instead shows all of the effects of a given event, and is similarly capable of representing an entire set of consequences that occur as a result of an event. ETA is often used to determine the probabilities of sets of consequences, given there are a set of protective safety features (e.g. alarms, seat belts, etc.) to mitigate the consequences of an event which each have a given effectiveness. [37, Chapter 12] Bow tie diagrams are structures which combine the fault tree and event tree for a single fault. These diagrams are used to provide a comprehensive understanding of the causes and effects of a hazardous event, and are used in the final development of a safety case for a system to show that the overall risk associated with that event is minimized [38].

Model-based approaches also exist which represent the propagation of risk through a system, and a variety of modelling approaches exist to perform this [39]. Design approaches, such as [40], focus on the propagation through the function structure of the design, since this is thought to be a representation of the design that represents the flow of tasks in a system while allowing a significant amount of design freedom [4]. A general optimization approach to designing with these fault models is provided in Hulse et al. [41], while Arlitt and Van Bossuyt [42] envision a designer-oriented fault visualization approach to minimizing risk. While these approaches exist, much of the work of model-building is siloed away as model assumptions and few visualization techniques exist (other than those listed above) to give designers the insight needed to understand and act on the model.

The limitation of these approaches is that the representation of the causal network which links root causes to system-level effects is at least partially hidden from the designer, either concatenating the true causal mechanisms and effects into lists (FMEA), focusing solely on the causes or effects of a single event (FTA/ETA), or focusing on the causes and effects of a single event (Bow Ties), rather than the set of causes leading to multiple events leading to multiple effects. In this work, we seek to represent this overall flow of causality with a directed acyclic network (as recommended by Dezfuli et al. [43]), based on the recognition of Denney et. al. [5] that a full representation of risk in a system requires a view of the entire Safety Architecture–the comprehensive mapping of risk sources to risk consequences through an intermediary structure of events.

## 3 NETWORK MODELS OF SYSTEM ARCHITECTURE

In SCA, system architecture is represented as a network. This representation allows for the modeling and analysis of the connectivity between the components as well as the explicit representation of redundancy. Networks are a natural representation of system architecture due to their close association with the commonly-used design structure matrix (DSM). A DSM represents the connections between components via a matrix, which


Copyright © 2019 ASME

is equivalent to the adjacency matrix which represents the connections between nodes in a network. A network, then, is simply a graphical representation of the connections of a DSM.

### 3.1 Network Modeling Methodology

In SCA, a network is used to represent system architecture. In general, a network consists of nodes (sometimes called vertices) connected by edges (sometimes called links). Here, the nodes represent components, and the edges are *directed*, meaning they indicate that information flows from a trailing edge to a leading edge. There are also *undirected* networks, where no direction of information flow is indicated. Whether directed or undirected, an edge between nodes represents some sort of interaction between those nodes. In the case of a directed network of components, the connections represent a physical connection and a direction of flow of information through the system architecture (e.g. a signal or electric current). A simple example is given in Fig. 1.

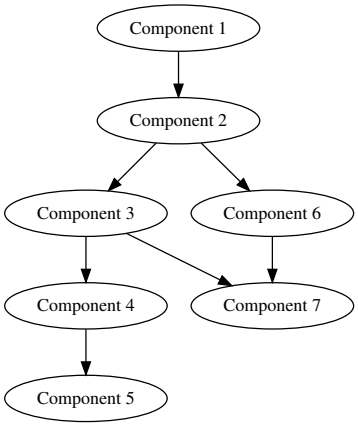


**FIGURE 1: SIMPLE EXAMPLE OF AN ARCHITECTURAL REPRESENTATION OF SYSTEM ARCHITECTURE.**

The nodes of a directed network can be categorized as either a *source*, *middle node*, or *sink*. A source, such as Component 1 in Fig. 1, has outgoing edges but no incoming edges. A middle node, such as Component 2, has both incoming and outgoing edges. A sink node, such as Component 5, has incoming edges but no outgoing edges. In a system architecture network, the source is likely to be a power source, while the sink could be a control surface or propeller.

### 3.2 Representing Redundancy

The network based representation of system architecture allows for the explicit representation of redundancy. In this representation, a component that is redundant in parallel to another component shares the same incoming and outgoing edges. In the example given in Fig. 2, Component 2 Redundant is redundant to Component 2. This representation does not itself distinguish between active and passive redundancy.

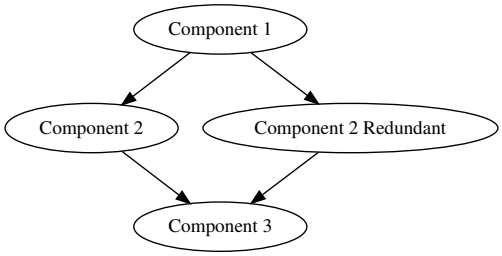


**FIGURE 2: REPRESENTATION OF REDUNDANCY, WHERE THE REDUNDANT COMPONENT IS CALLED COMPONENT 2 REDUNDANT.**

### 3.3 Eliminating Algebraic Loops

The algorithm for computing consequence severity involves propagating consequence information from one end of the network to the other. For the causal structure to be consistent with the methods used in this paper, it must be represented as a directed acyclic graph. For this reason, when generating a graph of consequences from a system diagram for preliminary analysis, it is important to eliminate algebraic loops. There are two main situations in which an algebraic loop may form: (1) when the system includes feedback, or (2) when a redundant component is shared between two other components.

To eliminate algebraic loops caused by feedback, the connection between the component providing information and the component receiving information required for feedback is removed. For example, consider a sensor that is sensing motor voltage. The connection in the network between the sensor and motor will be removed. This means that, in this case, the sensor will be treated as a source node. Removal of that edge can be justified by reasoning via the example that there is no interest in propagating a motor malfunction to the sensor since a sensor failure would lead to a motor malfunction in the worst case (an event that has already occurred).

There are also situations in which the addition of a shared redundant component leads to the creation of internal loops. For instance, consider the network in Fig. 3. In its current form in Fig. 3, BuR is redundant to Bu1 only. However, since Bu1 and Bu2 are the same type of component, BuR could be used as a redundancy for both. This form is shown in Fig. 4.

The problem with this representation of the cross redundancy is that it creates a loop that does not physically exist between Co1 and BuR. The loop does not physically exist because


Copyright © 2019 ASME

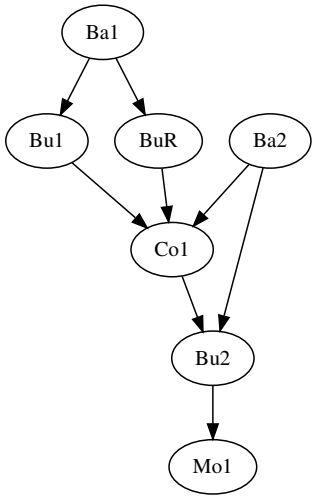


**FIGURE 3**: **NETWORK WITH A REDUNDANT COMPONENT, BuR, WHICH IS REDUNDANT TO Bu1.**

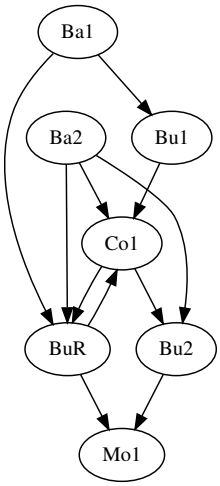


**FIGURE 4**: **BuR IS REDUNDANT TO Bu1 and Bu2. THERE IS A LOOP BETWEEN Co1 AND BuR.**

BuR cannot be redundant to both components at the same time. The solution is to create two nodes for BuR: BuRA and BuRB, as shown in Fig. 5.

Not only does this solution eliminate the algebraic loop, but it also correctly implies that BuR can only be redundant to either Bu1 or Bu2 at one point in time, even though it can be activated for either of the two components.

## 4 STRUCTURAL CONSEQUENCE

The consequence metric proposed in this paper quantifies the consequence of component failures within an architecture

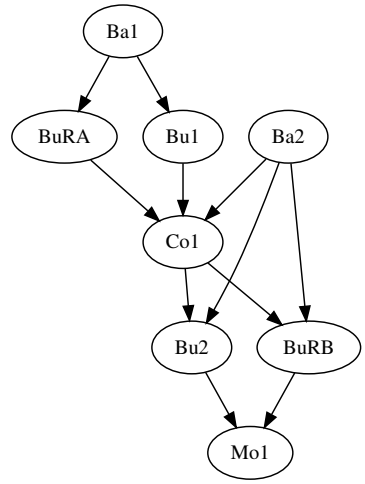


**FIGURE 5**: **ALGEBRAIC LOOP ELIMINATED BY SPLITTING BuR INTO BuRA AND BuRB.**

based on the layout of components and the flow of information/connections. The metric falls within the range of $[0,1]$, where 0 corresponds to the absence of consequences and 1 to the most severe consequences. This metric is computed through initialization of sink nodes (end components of a network) with user-defined consequences and back-propagating the consequences throughout the network via a propagation algorithm (presented in Section 4.5) and set operations.

### 4.1 Construction of the Consequence Space

A consequence space $\Omega$ is first constructed by aggregating all distinct and possible sources of cost and lumping them into a single set. The set of all defined consequences make up the consequence space and the overall cost of the entire consequence space is set to 1. Note that consequence in this paper refers to an undesirable event that occurs due to malfunctions. Within the consequence space, failure scenarios constituting different proportions of the total cost are represented. Intersections within the space represent consequences that are common to different failure scenarios. A metric $S$ can then be associated with these consequences to quantify the ordinal severity (e.g. 1-10 or cost) to allow for set operations to be performed.

In the subsections that follow, let $C_{k_1}$ represent the consequence of a node/component $k_1$ in the network, $E_{k_1:k_2}$ represent the consequence of a link/edge connecting $C_{k_1}$ to $C_{k_2}$, and $S(C_{k_1})$, $S(E_{k_1:k_2})$ represent the consequential severity (cost) of a component $k_1$ and an edge $E_{k_1:k_2}$ respectively. Note that the severity of the entire consequence space is 1 ($S(\Omega) = 1$).


Copyright © 2019 ASME

### 4.2 Scoring the Sink Nodes of the Network

The sink nodes of the network represent components at the endpoints of an architecture, meaning they do not have any outgoing edges. Components represented by sink nodes should typically be ones for which consequences can be estimated based on available data such as expert opinion, or field data. For example, let the sink node represent an aircraft aileron; although the electrical architecture operating an aileron might be novel, a cost estimate can still be deduced based on previous instances of aileron failures. Figure 6 shows an example of how consequences and costs are assigned in $\Omega$ for the sink nodes.

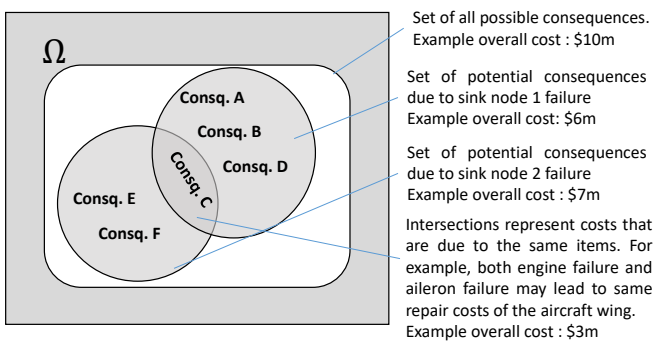


**FIGURE 6: CONSEQUENCE SPACE VISUALIZATION.**

### 4.3 Scoring the Edges of the Network

All edges included in the network are assumed to have functions necessary for the operation of the end components (sink nodes). In Section 4.3.1 the scoring scheme for edges with no redundancies is presented, and in Section 4.3.2, the scoring scheme for edges with redundancies is presented.

#### 4.3.1 Scoring Edges with No Redundancies

Failure of an edge for which no redundancies or protective factors exist leads to the potential failure of the component it feeds (at the head of the edge). This means that the failure or loss of the edge bears the same consequence as the loss of the component it feeds.

$$S(E_{i:j}) = S(C_j) \tag{1}$$

An example of this can be seen in Fig. 7. Loss of either of the edges connecting B to A, or A leads to loss of A.

*Property 1*: An edge for which no redundancies or protective features exist bears the same consequence as the component it feeds.

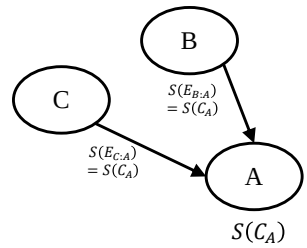


**FIGURE 7: EXAMPLE OF SCORING EDGES WITH NO REDUNDANCIES.**

#### 4.3.2 Scoring Edges with Redundancies

Although in practice redundancy is expressed in terms of components being redundant to one another, this paper represents redundancy via the edges of the network. Two edges are redundant if (i) their tails are connected to different components, (ii) their heads are connected to the same component and (iii) they share the same function.

Failure of a set of redundant edges leads to the failure of the component they feed. This means that the union of $n$ redundant edges $E_{i_1:j} \cdots E_{i_n:j}$ bears the same consequence as the component $C_j$ the edges all feed.

$$S(E_{i_1:j} \cup E_{i_2:j} \cdots E_{i_n:j}) = S(C_j) \tag{2}$$

*Property 2*: The union of a collection of redundant edges bears the same consequence as the component they feed.

*Assumption 1*: All redundant edges are assumed to have equivalent consequences, where edge consequences represent mutually exclusive subsets of the component consequence with equivalent severities.

This means that,

$$S(E_{i_1:j}) = S(E_{i_2:j}) = \cdots = S(E_{i_n:j}) \tag{3a}$$

$$E_{i_k:j} \cap E_{i_m:j} = \emptyset \quad k,m \in \{1,\ldots,n\} | k \neq m \tag{3b}$$

Using *Property 1* and *Assumption 1*,

$$S(E_{i_1:j}) = S(E_{i_2:j}) = \cdots = S(E_{i_n:j}) = \frac{S(C_j)}{n} \tag{4}$$

*Assumption 1* means that each redundant component equally shares the "responsibility" of the total consequences of all redundant edges. Let the overall consequence $C_j$ be broken down into a set of shared and mutually exclusive consequences $C_j^{i_1:j}, C_j^{i_2:j}, \cdots C_j^{i_n:j}$ corresponding to subsets of the overall consequence shared by the redundant edges $E_{i_1:j} \cdots E_{i_n:j}$ (such that $C_j^{i_k:j} = E_{i_k:j}$). This notation is introduced in order to allow proper


Copyright © 2019 ASME

propagation and traceability of consequences in the network. The following properties hold:

$$C_j^{i_l:j} \cap C_j^{i_m:j} = \emptyset \quad l,m \in \{1,\cdots,n\}\ l \neq m$$
$$C_j^{i_1:j} \cup C_j^{i_2:j} \cdots \cup C_j^{i_n:j} = C_j$$
$$C_j^{i_l:j} \cup C_j = C_j$$
$$C_j^{i_l:j} \cap C_j = C_j^{i_l:j}$$

In Fig. 8, the edges connecting D to A, and DR to A are redundant. The consequences of A are equally shared between the two edges.

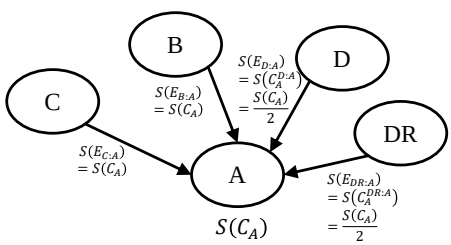


**FIGURE 8**: **EXAMPLE OF SCORING EDGES WITH REDUNDANCIES.**

### 4.4 Scoring the Nodes of a Network

Failure of a component that is not a sink leads to the failure or loss of all the pieces of information and links (edges) originating from that component.

*Property 3*: A component bears the same consequence as all of the outgoing edges from that component.

$$S(C_i) = S(E_{i:j_1} \cup E_{i:j_2} \cdots \cup E_{i:j_n}) \tag{6}$$

The severity of a component is evaluated in the same manner that the probabilities of set unions are evaluated in probability theory.

It is important to note that set operations are first used in node severity calculations. Once the sets have been fully simplified, severities are then quantified.

Figure 9 illustrates an example of scoring nodes in a network. Each of the nodes bear the same consequences as the union edges that are outgoing from them.

### 4.5 Propagation Algorithm

The propagation algorithm assumes two characteristics of the network:

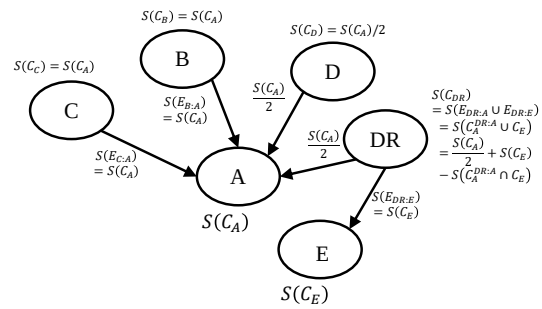


**FIGURE 9**: **EXAMPLE OF SCORING NODES.**

1. The network has no undirected edges (all edges have a direction).
2. The network has no internal loops.

If either of these assumptions do not apply, the propogation algorithm will not score nodes accurately and the network model must be revised (see Section 3.3).

The algorithm propagates the consequences of all sinks through the network until all nodes have been scored. This begins with a random sink node. First, the incoming edges associated with sink nodes are scored. Then, the propagation algorithm picks a random node that has all of its outgoing edges scored. This continues until all nodes, including source nodes, are scored. This process is represented visually in the flowchart in Fig. 10.

This algorithm ensures that the current node (the node that the algorithm is currently scoring) will always have all of its outgoing edges scored. This way, the algorithm will not have to "guess" any node scores and the computed node scores will be deterministic.

## 5 CASE STUDY

The methodology presented in the previous sections is demonstrated using a case study of a simplified electric subsystem of an all-electric fixed-wing unmanned aircraft system (UAS). The electric subsystem consists of several components and provides power to elecro-mechanical actuators used to manipulate an aileron and a flap of the UAS. A block diagram of the base (no redundancy) electric subsystem is given in Fig. 11.

Two different architectures that include redundancies and no single points of failure are presented for the electric subsystem. The first architecture, illustrated in Fig. 12, consists of the addition of a single redundancy in parallel to each of the components in Fig. 11. The second is based on a smart and re-configurable architecture with parallel and cross-redundancies (redundancies shared by different components) and is illustrated in Fig. 13. The motivation of the second architecture is based on the necessity of being able to perform comprehensive analysis on complex


Copyright © 2019 ASME

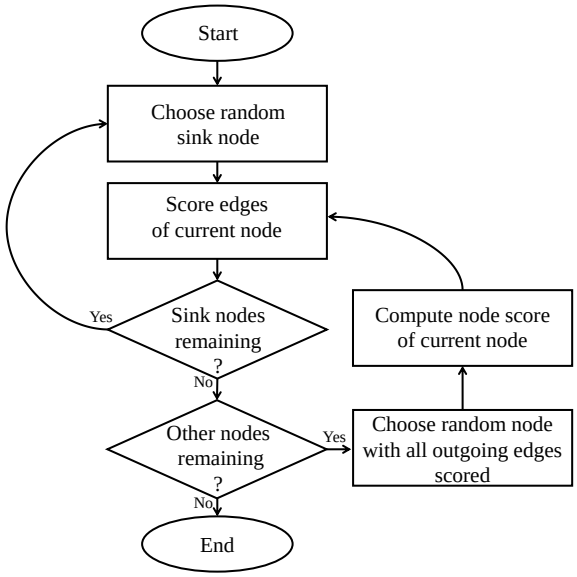


**FIGURE 10**: **FLOWCHART OF PROPAGATION ALGORITHM.**

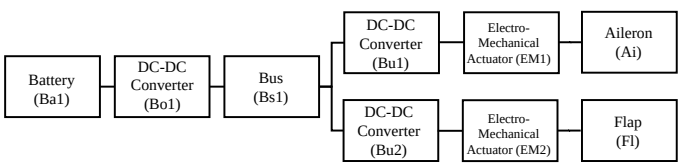


**FIGURE 11**: **ELECTRICAL SUBSYSTEM BLOCK DIAGRAM FOR BASE ARCHITECTURE WITH NO REDUNDANCIES ADDED.**

architectural designs for future aircraft that are optimized via re-configuration. The structural consequence metric is used to evaluate and quantify the structural consequence of each component within the two architectures. Note that the architectures are not based on realistic systems, they are hypothetical and developed for the purpose of demonstrating SCA.

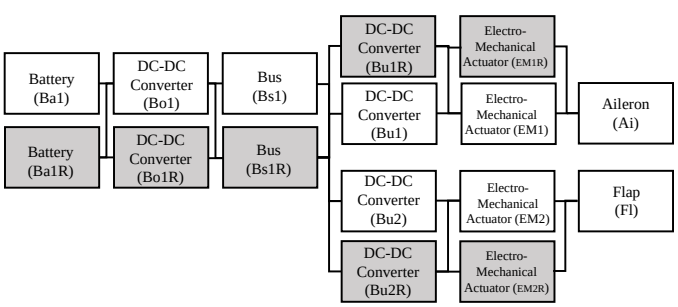


**FIGURE 12**: **ELECTRICAL SUBSYSTEM BLOCK DIAGRAM FOR ARCHITECTURE WITH PARALLEL REDUNDANCIES.**

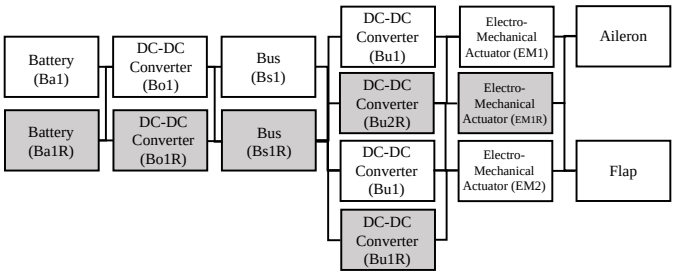


**FIGURE 13**: **ELECTRICAL SUBSYSTEM BLOCK DIAGRAM FOR ARCHITECTURE WITH RE-CONFIGURABLE REDUNDANCIES.**

As indicated by the process outlined in Section 4, a consequence space is first constructed and consequences are assigned to both the aileron and the flap. In this case study, consequences with a cumulative cost of $\$800,000$ are assigned to an aileron failure, consequences with a cumulative cost of $\$300,000$ are assigned to flap failures, and consequences with a cost of $\$100,000$ are shared between the two components.

The entire consequence space $\Omega$ has an overall cost of $\$1,000,000$, which is normalized to a severity of 1 ($S(\Omega) = 1$). Based on this normalization, the following serverities are computed:

$$S(C_{Ai}) = 0.8$$
$$S(C_{Fl}) = 0.3$$
$$S(C_{Ai} \cap C_{Fl}) = 0.1$$

It is worth mentioning that a consequence set captures a list of events (with associated costs) that occur as a result of a component failure. Each sink component has a consequence space formed by the list of events that can occur due to loss or failure, where intersections in consequence spaces of different components represent events that are mutual to both the components.

A network-based representation is first constructed for the base architecture, the parallel architecture, and the smart architecture based on Section 3. These network representations are illustrated in Figs.14–16. The white and gray portions of the nodes Bu1R and EM1R represent two re-configurable components with the same connections. Based on the properties and algorithm in Section 4, consequence sets from the sink nodes are back propagated in the three networks, and the severities (costs) of these consequences are quantified at each node based on the consequence space using operations that are analogous of probability measures in sample spaces. The computed edge and node severities for all three architectures are captured in Figs.14–16 and tabulated in Table 1.

A few examples of consequence propagation will be provided based on the parallel redundancies architecture depicted in


Copyright © 2019 ASME

Fig. 15. $EM1$ and $EM1R1$ are redundant with respect to $Ai1$. The consequence set $C_{Ai1}$ is therefore symmetrically split (Property 2, and Assumption 1) into two sets $C_{Ai1}^{EM1:Ai1}$ and $C_{Ai1}^{EM1R1:Ai1}$ corresponding to the edges $E_{EM1:Ai1}$ and $E_{EM1R1:Ai1}$ respectively.

The severities of $E_{EM1:Ai1}$ and $E_{EM1R1:Ai1}$ can be computed as:

$$S(E_{EM1:Ai1}) = S(C_{Ai1}^{EM1:Ai1}) = \frac{S(Ai)}{2} = 0.4$$
$$S(E_{EM1R1:Ai1}) = S(C_{Ai1}^{EM1R1:Ai1}) = \frac{S(Ai)}{2} = 0.4$$

The consequences of $EM1$ and $EM1R1$ are then computed using Property 3,

$$S(C_{EM1}) = S(E_{EM1:Ai1}) = 0.4$$
$$S(C_{EM1R1}) = S(E_{EM1R1:Ai1}) = 0.4$$

Similar to $S(E_{EM1:Ai1})$ and $S(E_{EM1R1:Ai1})$, the edges $S(E_{Bu1R:EM1R})$, $S(E_{Bu1:EM1R})$, $S(E_{Bu1R:EM1})$, and $S(E_{Bu1:EM1})$ all have severities of 0.2.

The consequences of $Bu1$ and $Bu1R$ can be computed via Property 3. Note that the union operator is mathematically evaluated in the same manner as probability measures in sample spaces.

$$\begin{aligned} S(C_{Bu1}) &= S(E_{Bu1:EM1} \cup E_{Bu1:EM1R}) \\ &= S(E_{Bu1:EM1}) + S(E_{Bu1:EM1R}) - \\ &S(E_{Bu1:EM1} \cap E_{Bu1:EM1R}) = 0.4 \end{aligned}$$

Similarly, $S(Bu1R) = 0.4$. Note that due to mutual exclusivity of the redundant edges, $S(E_{Bu1:EM1} \cap E_{Bu1:EM1R}) = \emptyset$.

A similar process is followed for the rest of the nodes and edges in between $Ai1, Fl1$ and $Bs1$, $Bs1R$.

The consequence of $Bs1$ and $Bs1R$ are dependent on consequences originating from both $Al$ and $Fl$. By employing Properties 1–3, and using the fact that $C_{Al}$ and $C_{Fl}$ do indeed intersect due to shared consequences, $S(C_{Bs1})$ (and similarly $S(C_{Bs1R1})$) can be computed as follows,

$$\begin{aligned} S(C_{Bs1}) &= S(E_{Bs1:BuR1} \cup S(E_{Bs1:Bu1} \cup S(E_{Bs1:BuR2} \cup S(E_{Bs1:Bu2})) \\ &= S(E_{Bs1:BuR1}) + S(E_{Bs1:Bu1}) \\ &+ S(E_{Bs1:BuR2}) + S(E_{Bs1:Bu2}) \\ &- S(E_{Bs1:Bu1} \cap E_{Bs1:Bu2}) \\ &- S(E_{Bs1:Bu1} \cap E_{Bs1:Bu2R}) \\ &- S(E_{Bs1:Bu1R} \cap E_{Bs1:Bu2}) \\ &- S(E_{Bs1:Bu1R} \cap E_{Bs1:Bu2R}) = 0.5 \end{aligned}$$

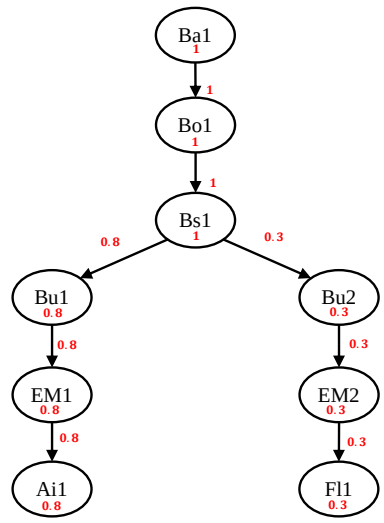


**FIGURE 14: NETWORK REPRESENTATION OF THE BASE ARCHITECTURE.**

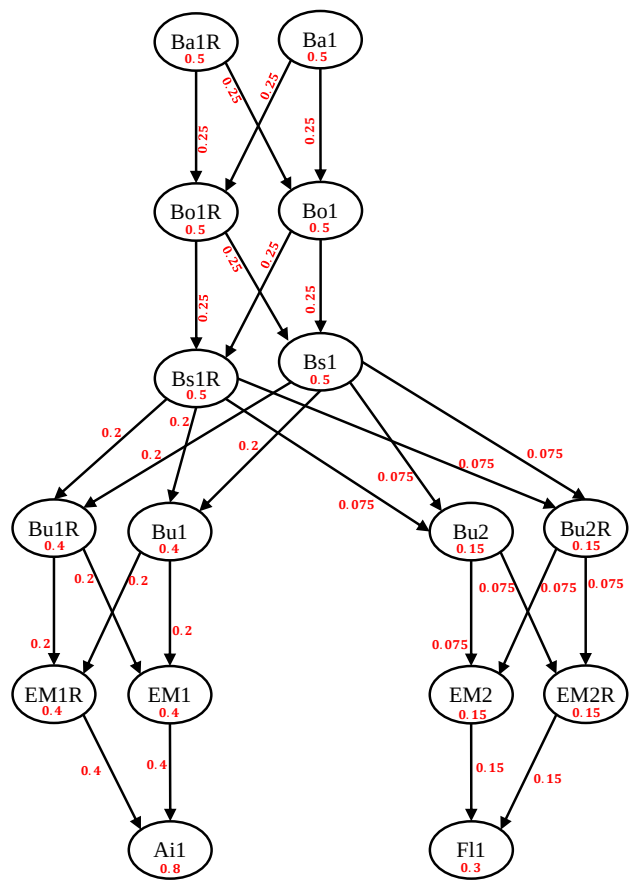


**FIGURE 15: NETWORK REPRESENTATION OF THE ARCHITECTURE WITH PARALLEL REDUNDANCIES.**


Copyright © 2019 ASME

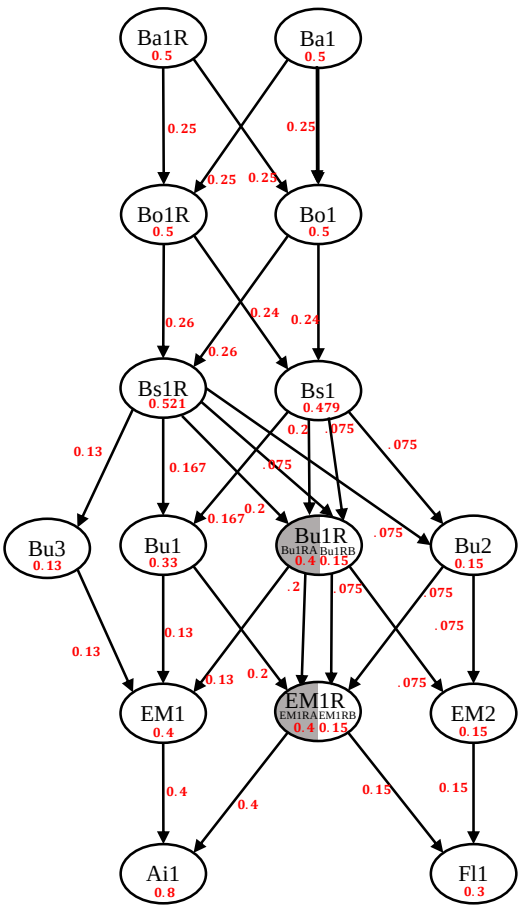


**FIGURE 16: NETWORK REPRESENTATION OF THE ARCHITECTURE WITH PARALLEL AND RE-CONFIGURABLE REDUNDANCIES (WHITE & GRAY).**

**TABLE 1: COMPONENT SCA SEVERITY SCORES FOR BASE, PARALLEL, AND RE-CONFIGURABLE ARCHITECTURES.**

| Component | Base | Parallel | Re-config. |
|---|---|---|---|
| Ba1 | 1 | 0.5 | 0.5 |
| Ba1R1 | N/A | 0.5 | 0.5 |
| Bo1 | 1 | 0.5 | 0.5 |
| Bo1R1 | N/A | 0.5 | 0.5 |
| Bs1 | 1 | 0.5 | 0.479 |
| Bs1R1 | N/A | 0.5 | 0.521 |
| Bu1 | 0.8 | 0.4 | 0.33 |
| Bu1R1 | N/A | 0.4 | 0.4 |
| Bu1R2 | N/A | N/A | 0.13 |
| Bu2 | 0.3 | 0.15 | 0.15 |
| Bu2R1 | N/A | 0.15 | N/A |
| EM1 | 0.8 | 0.4 | 0.4 |
| EM1R1 | N/A | 0.4 | 0.4 |
| EM2 | 0.3 | 0.15 | 0.15 |
| EM2R1 | N/A | 0.15 | N/A |
| Ai1 | 0.8 | 0.8 | 0.8 |
| Fl1 | 0.3 | 0.3 | 0.3 |

The results show that SCA allows the designer to identify nodes for which there are high consequences of failure based on the system architecture. Such information is valuable as it allows for quantification of component failure consequence and early identification of system weak points.

## 6 DISCUSSION

SCA provides value to the design process by giving designers the ability to represent the chain of causality that leads to various consequences in a system. Unlike fault tree analysis, this view of causality is not limited by single "top event" consequences but by the set of consequences that could occur in a system, consequences that could have different impacts to safety, maintenance and repair, availability, and convenience. This expressiveness allows the designer to assess the true system-level impacts of a failure and make design changes that would lead to different changes in the causal graph that lead to different sets of consequences.

Additionally, as shown in the case study, assessing the structural importance of components from the point of view of the consequences they produce allows the designer to consider a novel set of structural importance metrics to assess redundancy schemes. These metrics, based on the "virtual severity" of component failure, show the designer which redundancies are most consequential in the architecture, and thus which components are the weakest points from the structural standpoint. Table 1 shows the effect of different redundancy strategies on this consequence share. As shown added redundancies reduce the share of severity associated with each component, while allowing reconfiguration changes the amount of "share" each component will take, spreading the amount across several (in the case of Bu1), or concentrating it (in the case of Bs1). This shows how virtual severity can be used as a structural metric to inform design.

The difference between this virtual consequence and the direct consequence of failure should be noted. While direct consequence is defined as the direct consequence that would occur


Copyright © 2019 ASME

given a node event occurs, there is no direct consequence of a redundant component failing (to functionality, at least). This virtual consequence, the proportion of the failure consequence of the redundancy scheme associated with each component, therefore, requires some technical interpretation beyond direct severity. Nevertheless, it does show the relative "share" of importance of redundant components, which is useful to quantify for developing redundant architectures.

## 7 CONCLUSIONS

This research has proposed a novel metric for measuring the consequence of components within a system architecture that requires neither a fault tree nor failure rate information. A case study demonstrated how the metric is able to guide early design decisions and provide early estimates for redundancy placement, which can be later revised once more detailed system information becomes available. Use of this metric in early design may help designers from underestimating the costs of adding redundancy and may help them plan for size and weight considerations early in the design process. This can aid in designers' efforts to design safe systems without overly sacrificing performance or cost.

## 8 FUTURE WORK

Future work should further develop structural consequence analysis into a fully-expressive formalism to analyze the flow of causality in a system and demonstrate the implementation of this tool in software for immediate automated visualization of consequence propagation. Additionally, future work should show the application of this method to consider a more expanded case study, considering the internal causal mechanisms that produce faults in components rather than a simple system structure. Additionally, the effect of protective factors other than redundancy should be considered and factored into this method. This would be complemented by quantifying additional structural metrics to determine how inherently safe or hazardous a system is based on the graph structure.

## ACKNOWLEDGMENT

This research is supported by the System-Wide Safety (SWS) project in the Airspace Operations & Safety program (AOSP) in the NASA Aeronautics Research Mission Directorate (ARMD). The research is also partially supported by the National Science Foundation CMMI 1562027, and NASA award number 80NSSC17M0018. Any opinions or findings of this work are the responsibility of the authors and do not necessarily reflect the views of the sponsors or collaborators.

## REFERENCES

[1] Sklaroff, J., 1976. "Redundancy management technique for space shuttle computers". *IBM Journal of Research and Development,* ***20***(1), February, pp. 20–28.

[2] Coit, D., 2001. "Cold-standby redundancy optimization for nonrepairable systems". *IIE Transactions,* ***33***(6), pp. 471–478.

[3] Sinha, K., Suh, E. S., and de Weck, O., 2017. "Correlating integrative complexity with system modularity". In *ASME 2017 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2017)*, Vol. 2A: 40th Design Automation Conference. ASME, New York, p. V02AT03A048.

[4] Pahl, G., and Beitz, W., 2007. *Engineering design: a systematic approach*. Springer Science & Business Media.

[5] Denney, E., Pai, G., and Whiteside, I., 2017. "Modeling the safety architecture of uas flight operations". In International Conference on Computer Safety, Reliability, and Security, Springer, pp. 162–178.

[6] Vesely, W. E., Goldberg, F. F., Roberts, N., and Haasi, D. F., 1981. The fault tree handbook. Tech. Rep. NUREG0492.

[7] Barlow, R. E., and Proschan, F., 1975. "Importance of system components and fault tree events". *Stochastic Processes and their Applications,* ***3***(2), pp. 153–173.

[8] Birnbaum, Z., 1968. On the importance of different components in a multicomponent system. Tech. Rep. 54.

[9] Yuhua, D., and Datao, Y., 2005. "Estimation of failure probability of oil and gas transmission pipelines by fuzzy fault tree analysis". *Journal of Loss Prevention in the Process Industries,* ***18***(2), pp. 83–88.

[10] Sinnamon, R. M., and Andrews, J., 1998. "Improved accuracy in quantitative fault tree analysis". *Quality and Reliability Engineering International,* ***13***(5), pp. 285–292.

[11] Ruijters, E., and Stoelinga, M., 2015. "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools". *Computer science review,* ***15–16***, pp. 29–62.

[12] Boudali, H., Crouzen, P., and Stoelinga, M. "A compositional semantics for dynamic fault trees in terms of interactive markov chains". *International Symposium on Automated Technology for Verification and Analysis ATVA 2007: Automated Technology for Verification and Analysis*, pp. 441–456.

[13] Dugan, J., Bavuso, S., and Boyd, M., 1990. "Fault trees and sequence dependencies". *Annual Proceedings on Reliability and Maintainability Symposium*.

[14] Liang, X., Yi, H., Zhang, Y., and Li, D., 2009. "A numerical simulation approach for reliability analysis of fault-tolerant repairable system". *Proceedings of the 8th International Conference Reliability, Maintainability and Safety*, pp. 191–196.

[15] Zhang, X., Miao, Q., Fan, X., and Wang, D., 2009. "Dynamic fault tree analysis based on petri nets". *Proceedings*


Copyright © 2019 ASME

*of the 8th International Conference Reliability, Maintainability and Safety*, pp. 138–142.

[16] Dutuit, Y., and Rauzy, A., 2001. "Efficient algorithms to assess component and gate importance in fault tree analysis". *Reliability Engineering System Safety,* ***72***(2), pp. 213–222.

[17] Buchacker, K., 2000. "Modeling with extended fault trees". *Proceedings of the 5th International Symposium on High Assurance Systems Engineering*, pp. 238–246.

[18] Bobbio, A., and Codetta-Raiteri, D., 2004. "Parametric fault trees with dynamic gates and repair boxes". *Proceedings of Reliability and Maintainability Symposium*, pp. 459–465.

[19] Albert, R., Jeong, H., and Barabsi, A. L. "Diameter of the world-wide web". *Nature,* ***401***, pp. 130–131.

[20] Mason, O., and Verwoerd, M., 2007. "Graph theory and networks in biology". *IET Systems Biology,* ***1***(2), pp. 89–119.

[21] Kossinets, G., 2006. "Effects of missing data in social networks". *Social networks,* ***28***, pp. 247–268.

[22] Freeman, L. C., 1978. "Centrality in social networks conceptual clarification". *Social networks,* ***1***(3), pp. 215–239.

[23] Barabási, A.-L., 2003. *Linked: The new science of networks*. Plume.

[24] Erdős, P., and Rényi, A., 1959. "On random graphs". *Publicationes Mathematicae Debrecen,* ***6***, pp. 290–297.

[25] Haley, B., Dong, A., and Tumer, I. Y., 2014. "Creating faultable network models of complex engineered systems". In *ASME 2014 International Design and Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2014)*, Vol. 2A: 40th Design Automation Conference. ASME, New York, p. V02AT03A051.

[26] Haley, B., Dong, A., and Tumer, I., 2016. "A comparison of network-based metrics of behavioral models of complex engineered systems". *Journal of Mechanical Design,* ***138***(12), p. 121405.

[27] Walsh, H. S., Dong, A., and Tumer, I. Y., 2017. "The structure of vulnerable nodes in behavioral network models of complex engineered systems". In *ASME 2017 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2017)*, Vol. 7: 29th International Conference on Design Theory and Methodology. ASME, New York, p. V007T06A019.

[28] Walsh, H., Dong, A., and Tumer, I., 2018. "The role of bridging nodes in behavioral network models of complex engineered systems". *Design Science,* ***4***.

[29] Sosa, M., Eppinger, S., and Rowles, C., 2007. "A network approach to define modularity of components in complex products". *Journal of Mechanical Design,* ***129***(11), pp. 1118–1129.

[30] Ma, S., Jiang, Z., and Liu, W., 2016. "A design change analysis model as a change impact analysis basis for semantic design change management". *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* ***231***(13), pp. 2384–2397.

[31] Ball, Z., and Lewis, K., 2018. "Observing network characteristics in mass collaboration design projects". *Design Science,* ***4***(e4).

[32] Wang, M., Sha, Z., Huang, Y., Contractor, N., Fu, Y., and Chen, W., 2016. "Forecasting technological impacts on customers co-consideration behaviors: A data-driven network analysis approach". In *ASME 2016 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE 2016)*, Vol. 42nd Design Automation Conference. ASME, New York, p. V02AT03A040.

[33] Wang, M., Chen, W., Huang, Y., Contractor, N. S., and Fu, Y., 2016. "Modeling customer preferences using multidimensional network analysis in engineering design". *Design Science,* ***2***(e11).

[34] Baldwin, C., MacCormack, A., and Rusnak, J., 2014. "Hidden structure: Using network methods to map system architecture". *Research Policy,* ***43***(8), pp. 1381–1397.

[35] Sosa, M., Mihm, J., and Browning, T., 2011. "Degree distribution and quality in complex engineered systems". *Journal of Mechanical Design,* ***133***(10), p. 101008.

[36] Ullman, D., 2009. *The mechanical design process*. McGraw-Hill Science/Engineering/Math.

[37] Lewis, E. E., and Lewis, E. E., 1987. *Introduction to reliability engineering*, Vol. 2. Wiley New York et al.

[38] Clothier, R., Denney, E., and Pai, G. J., 2017. "Making a risk informed safety case for small unmanned aircraft system operations". In 17th AIAA Aviation Technology, Integration, and Operations Conference, p. 3275.

[39] Joshi, A., Whalen, M., and Heimdahl, M., 2005. "Model-based safety analysis final report". *NASA Techreport*.

[40] Kurtoglu, T., Tumer, I. Y., and Jensen, D. C., 2010. "A functional failure reasoning methodology for evaluation of conceptual system architectures". *Research in Engineering Design,* ***21***(4), pp. 209–234.

[41] Hulse, D., Hoyle, C., Goebel, K., and Tumer, I. Y., 2019. "Quantifying the resilience-informed scenario cost sum: A value-driven design approach for functional hazard assessment". *Journal of Mechanical Design,* ***141***(2), p. 021403.

[42] Arlitt, R., and Van Bossuyt, D. L., 2019. "A generative human-in-the-loop approach for conceptual design exploration using flow failure frequency in functional models". *Journal of Computing and Information Science in Engineering,* ***19***(3), p. 031001.

[43] Dezfuli, H., Benjamin, A., Everett, C., Feather, M., Rutledge, P., Sen, D., and Youngblood, R., 2015. "NASA system safety handbook. Volume 2: System safety concepts, guidelines, and implementation examples".


Copyright © 2019 ASME