“Selfish” algorithm for reducing the computational cost of the network survivability analysis

Svetlana V. Poroseva

Mechanical Engineering, MSC01 1150, 1 University of New Mexico, Albuquerque, NM 87131-0001

1(505)2771493

1(505) 2772761

poroseva@unm.edu

Abstract: In Nature, the primary goal of any network is to survive. This is less obvious for engineering networks (electric power, gas, water, transportation systems, etc.) that are expected to operate under normal conditions most of the time. As a result, the ability of a network to withstand massive sudden damage caused by adverse events (or *survivability*) has not been among traditional goals in systems design. Reality, however, calls for the adjustment of design priorities. As modern networks develop toward increasing their size, complexity, and integration, the likelihood of adverse events also increases due to technological development, climate change, and activities in the political arena, among other factors. Under such circumstances, a network failure has an unprecedented effect on lives and economy. To mitigate the impact of adverse events on network operability, the survivability analysis must be conducted at an early stage in network design. Such analysis requires the development of new analytical and computational tools. A computational analysis of network survivability is an exponential time problem that makes the analysis computationally unfeasible for large-scale networks. The current paper describes a new algorithm in which reduction of the computational cost is achieved by mapping an initial network topology with multiple sources and sinks onto a set of simpler smaller topologies with multiple sources and a single sink. Steps for further reducing the time and space expenses of computations are also discussed.

Keywords: networks, survivability, resilience, computational complexity

1 Introduction

In Nature, the primary goal of any network is to survive, that is, if not to avoid, then withstand life-threatening damage. To be life threatening, damage should be sudden and either be very precise to destroy vital network elements or be of a large scale to eliminate the possibility of a network recovering in the short term. It is rather clear that the latter scenario of massive sudden damage is more likely to be successful.

Survival as a goal is less obvious in the case of man-made networks, particularly those that are designed for utilitarian purposes (electric power, gas, water systems, etc.). These networks are expected to operate under normal conditions most of the time. Operational faults associated with such conditions, manufacturing faults, fatigue cracking, and maloperation for example (Lancaster 2000), are considered to be predictable, random, and repairable within an estimated time. That is, such faults are manageable and, therefore, not an ultimate threat for networks.

Reality, however, changes fast and dramatically. Unprecedented development in science and technology, changes in economical models on global and local scales, population growth and changes in population distribution over the planet have brought new challenges for engineers to face. Among the most difficult and costly challenges is the increased likelihood of massive, unexpected damage to networks caused by adverse events: natural disasters (hurricanes, earthquakes, floods, wild fires), hostile disruptions (physical destructions, electronic intrusions), man-made errors, and unforeseen combinations of events.

Certainly, some of these threats are not new, but their likelihood is visibly increasing due to climate change, events in the political arena, easy public access to information and technologies, and various other factors. Many new technologies along with perspectives of life improvement bring opportunities to misuse them, thus further contributing in the increased likelihood of adverse events.

Due to the increased scale, complexity, and integration of modern networks, the impact of their failure on lives and the economy also shows unmatched growth (Lancaster 2000). Some networks, such as telecommunication and financial systems, have already reached the global level. World economical

crises are the best and arguably most unfortunate examples that demonstrate how failures in hyper-large networks affect societies worldwide.

Network integration has yet to reach the global level, but it has already been implemented on the national scale. Seven of the Nation's eight critical infrastructures - telecommunications, natural gas and oil, banking and finance, transportation, water supply systems, government services, and emergency services – are directly linked to the electric power infrastructure (President's Commission 1997). Yet, multiple studies (see, e.g., NAERC 2007; U.S. DOE 2002, 2005) indicate that in its current state, the modern electric power infrastructure is not prepared to withstand many forms of large-scale damage. A NASA-funded study (CSEISSWE 2008) reported that loss of electricity would cause "water distribution affected within several hours; perishable foods and medications lost in 12-24 hours; loss of heating/air conditioning, sewage disposal, phone service, fuel re-supply and so on." As an example, the blackout on August 14, 2003, the biggest in the U.S. history, affected two countries, left 60 million people in the dark, caused multiple deaths, and resulted in economic losses estimated to be from 6 to 10 billion (NextGen ECMIS, 2008).

While adverse events in engineering networks cannot be avoided, their scale and impact can and should be mitigated. This can be done by analyzing a network's ability to withstand massive sudden damage (*survivability*) at an early stage in the network design and choosing a network with the maximum survivability among proposed designs.

Many factors contribute to network survivability (Oliveto 1998; Hill 2005). This is probably one of the reasons why there is no generally accepted definition of survivability (see Poroseva et al. (2009) and Saleh and Castet (2010) for more discussion on this topic). In our research, the effect of a network's topology on the network's survivability is analyzed in networks with multiple sources and multiple sinks. A "source" is defined as a network element generating a quality or service of interest and a "sink" is an element consuming this quality (or service). Many engineering networks fall into this category. Survivability of such a network can be defined as the continuing supply of a quality of interest from sources to sinks in an amount sufficient to satisfy the sinks' demand for this quality in the presence of multiple faults in the network's elements.

Depending on the origin and evolution of faults, the amount of the quality of interest available to each sink may vary with time. Intuitively however, the word "survivability" is associated not with a process, but with the final state of a network after all possible damage has occurred and before any recovery action has taken place. Indeed, one declares "I survived a fire (war, tornado etc.)" only after the event took place, not during its development.

The current study adopts the same approach, that is, the final steady state of a network is considered only after all faults (including cascaded and secondary) have occurred and before any repair has been accomplished. Since the origin and development of faults are not the focus of the current study, this research is applicable to any type and intensity of adverse event in which multiple faults have resulted either due to multiple events (independent or correlated) or due to a single event that caused cascaded/secondary faults in multiple network elements.

In the final steady state, one determines whether a network survived an adverse event by comparing the amount of a quality of interest available in a transformed network with the demand for this quality/service.

The mathematical problem to be solved involves establishing sources that survived faults, determining their capacity, and verifying their connection to sinks. This problem should be solved for all possible combinations of faults. The traditional engineering approach of identifying the worst-case scenario is not applicable to modern networks due to their scale and complexity.

Previously, we suggested (Poroseva et al. 2005) a probabilistic framework for quantifying the network survivability and developed (Poroseva et al. 2009) a computational algorithm for conducting the survivability analysis of small- to medium-sized networks with multiple sources and a single sink. However, in larger networks with multiple sinks, the algorithm becomes computationally intractable.

The main factor is the size of the problem. Indeed, in a network with M elements, the total number of possible combinations of faults is 2^M. (In the final steady state, each element can only be in one of two states: available or faulty). Engineering networks may include several thousands of elements or more. From a computational complexity point of view, the problem belongs to the class of exponential time (EXP) problems (Garey and Johnson 1979; Goldreich 2008;

Papadimitriou 1994). That is, the problem is more complex than more familiar NP-complete problems (see, e.g., Papadimitriou 1994).

The second contributor to the problem complexity is the graph search algorithm that runs for every combination of faults to establish a connection between survived sources and sinks. Presently, this is accomplished using the standard depth-first search traversal algorithm (see, e.g., Cormen at al. 2001). Although actual time required to run this algorithm may vary depending on its application, the theoretical upper bound is linear in the graph size, here, *M*. Other search algorithms are either comparable in performance (e.g., breadth-first search algorithm) or logarithmic in the graph size (Goldreich 2008).

A rough estimate of the combined contribution of these two factors is $M \cdot 2^M$.

To conduct the network survivability analysis of real-life networks in a manageable amount of time and to use this analysis for maximizing their survivability, the requirement for computational time should be substantially reduced. Some steps towards achieving this goal were suggested in Poroseva et al. (2009) and Poroseva (2010a). The current paper presents a new "selfish" algorithm[1] where the required computational time is decreased drastically by reducing the domain for the graph search algorithm application. This is accomplished by mapping the initial topology of a large-scale network with multiple sources and sinks onto a set of smaller topologies (or *sub-topologies*) with multiple sources and a single sink. Then, the search algorithm is used only on the set of sub-topologies.

2 General framework of the survivability analysis

2.1 Network representation

Networks of practical interest may include a variety of different elements, but for the survivability analysis, only four types of elements are of importance: three types of nodes and links that connect nodes with one another. The three types of nodes are: sources that generate a quality (or service) of interest, sinks that

[1] Initially, the algorithm was presented in the conference paper (Poroseva 2010b) with application to a power system. The current paper, however, is an independent work for the most part.

consume this quality, and interconnections through which the quality is transported, ideally without a change in the amount. The flow direction through interconnections may vary depending on the network state. Other network elements that are connected in series between any two of the three specified types of nodes are represented by a single link between the two nodes. Since a fault in any of the elements connected in series leads to interruption of the quality flow through all elements, such simplification of a network is valid. This simplification also reduces the scale of the analyzed network, *M*, and thus, the size of the problem.

An example of an analyzed network is shown in Fig. 1. The figure is a modified standard power system diagram of a notional Medium Voltage DC shipboard power system (IEEE 2010) in which all network elements in series (with the exception of sinks, sources, and intersections) are represented by a single link. In the figure, the circles with "-" correspond to sinks (loads), the circles with "+" show sources (generators), and the small dark circles are interconnections.

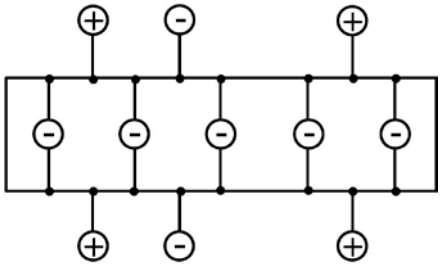


Fig1 Initial network representation for the survivability analysis

Although the network representation shown in Fig. 1 is appealing due to its visual clarity, the number of possible faulty elements – four sources, seven sinks, 16 interconnections, and 32 links, 59 in total – is already a computational challenge (2^{59} possible final steady states in the network). This number can be further reduced for the purposes of the survivability analysis. Indeed, if a source or a sink is connected to the network by a single link, then the node and the link are connected in series and, therefore, can be represented by a single link. If there are several links connecting a source or a sink to the network, faults in all of them isolate the node from the network, which is equivalent to a fault in the node. Thus, there is no need to consider faults in nodes separately, and these faults can be

removed from the network. Similarly, faults in interconnections can also be removed from consideration.

Figure 2 shows the result of such a simplification on the network in Fig. 1. The new representation of the network now includes only 32 links. Faults in these links completely reproduce all possible faults in the network in Fig. 1.

There are three different types of links in the network in Fig. 2: links adjacent to sources (vertical “VT” links), links adjacent to sinks (vertical “VB” links), and links between interconnections (horizontal “H” links). The VB- and VT-links are shown in the figure as arrow-headed links. The four VT-links (VT1-VT4) correspond to the four sources in Fig. 1. The seven sinks in Fig. 1 are represented by twelve VB-links that connect the sinks to the network. Five of the sinks are connected to the network by two links each (VB11 and VB12, VB21 and VB22, VB31 and VB32, VB41 and VB42, and VB51 and VB52). Each of the links VB6 and VB7 connects one sink to the network.

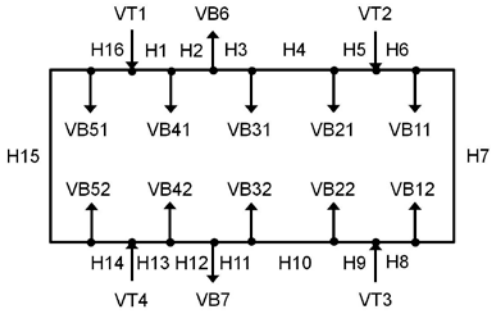


Fig2 Final network representation for the survivability analysis

To recognize the three types of links mathematically, the VB- and VT-links are assigned a weight. The sign of the weight shows the flow direction through a link: the VT-links have “+” weight and the VB-links have “-” weight. No sign is assigned to the H-links because the flow direction through such links may vary depending on the network state. The value of the weight assigned to a vertical link is based on its capacity or the maximum amount of a quality of interest it can deliver in relation to the total demand.

Currently, for simplicity purposes only, it is assumed that if multiple VT-links are adjacent to a source, each link is capable of delivering the total amount of a quality of interest generated by the source to the network. In a similar manner, it is assumed that the capacity of each VB-link adjacent to a sink is such

that the total demand from a sink for the quality can be satisfied through a single link. The capacity of the H-links allows any required amount of the quality (service) to flow through. These assumptions can be easily adjusted when the analysis is applied to a specific network. Also, additional weights can be added to the links. However, since the concern of the current paper is to reduce the computational cost of the survivability analysis, the simplified assumptions serve their purpose.

Notice that the network representation described above puts the emphasis on faults in links rather than faults in nodes. This is an essential difference in our survivability analysis from the traditional reliability/availability analysis. In the latter, the main concern is the availability of the system equipment, whereas the availability of wires (links) is assumed. Under many adverse conditions, however, damage originates externally and results in physical damage to network elements. Of all network elements, links are the most vulnerable to physical damage for reasons including their extent, protection cost, and exposure to elements, to mention just a few. In some adverse events, such as power grid blackouts, links may not be destroyed but are tripped off, which also makes them unavailable to the grid. Thus, with the possible exception of wireless networks and damage initiated and spread within a network (e.g., cyber attacks in communication systems), faults in links are the dominant concern when a network experiences an adverse event. Since faulty links can separate perfectly functioning sources and sinks from one another, reliability of equipment (availability of nodes) becomes a less important factor for the survivability analysis. Moreover, as discussed above, faults in nodes can be represented by faults in adjacent links.

Mainstream network analyses also focuses on the effect of removing nodes rather than links from a network (see, e.g., Newman et al. 2006). After attempting to follow this path, we realized that massive, unpredicted damage affecting any element in a network of arbitrary topology could be more accurately represented by faults in links. Indeed, if a node has multiple adjacent links and only some of them fail, then the situation is not easy to describe by removing the node. Moreover, for survivability of a real-life network, it can be of crucial importance to identify exactly which links are faulty from those adjacent to the node. Engineering parameters specific to a particular real-life network (individual link

capacity, weight, length, cost, probability of failure, etc.) are also easier to assign in network representations using links.

2.2 Fault scenarios

As we are concerned with the final steady state of a network after damage, faults in network elements are assumed to be failures that cannot be recovered in the short term. Therefore, multiple faults are viewed as simultaneous events, and only one fault can occur in a given element. A faulty element is completely unavailable to the flow of a quality of interest.

The combination of faults in the final steady state of a network is called the fault scenario.

For a given fault scenario, the general problem formulation is to establish

- how many sources of a quality of interest remain in operational conditions (survived);
- how many survived sources are connected to sinks;
- whether survived sources connected to sinks can satisfy their demand for the quality of interest.

In regards to demand, a fault scenario can be one of three types depending on the network response: “no response”, reconfiguration, and complete failure. In a “no response” scenario, faults do not reduce the supply of the quality of interest from sources to sinks. A fault scenario, in which the supply is reduced but not interrupted, requires a network reconfiguration. That is, some sinks must be intentionally disconnected from the network to balance the demand and the available amount of the quality of interest. Such a scenario is called a reconfiguration scenario. If faults completely isolate sinks from sources, the fault scenario is that of a complete network failure.

To determine the fault scenario type, one has to compare the available amount of the quality of interest with the demand for this quality. Let $Q_D = \sum_{b=1,\dots,B} Q_{D_b}$ be the demand existing in a network before faults, and $Q_G = \sum_{t=1,\dots,T} Q_{G_t}$ be the total amount of the quality that can be produced by available sources. Here, B and T are the numbers of sinks and sources, respectively. In Figs. 1-2, B = 7 and T = 4. Before damage, it is typical that $Q_G \geq Q_D$. If after faults $Q_G \geq Q_D$, this is a “no response” fault scenario. If $\min[Q_{D_b}] \leq Q_G < Q_D$, this

is a reconfiguration scenario. If $Q_G < \min[Q_{D_b}]$, this is a case of complete network failure. One may need to adjust these criteria when considering a specific network.

Notice that the amount of the quality of interest available after damage should be compared with the demand existing in the network before the faults occurred. This corresponds to the most challenging case when all sinks have survived faults and now have a request for the quality. As such, this case provides a perfect opportunity to test the network's ability to withstand damage.

2.3 Network response probabilities

In a network with M elements, there are $N = 2^M$ possible fault scenarios. The total number N of all fault scenarios is the sum of S "no response" fault scenarios, R reconfiguration scenarios, and F scenarios of complete network failure: $N = S + R + F$.

The number of fault scenarios leading to a specific network response can be used to determine the response probability P of a network to such scenarios: $P(S) = S / N$ (probability of "no response" scenarios), $P(R) = R / N$ (probability of reconfiguration scenarios), and $P(F) = F / N$ (probability of complete failure).

In a similar manner, one can define the network response probabilities at a given number of faults m:

$$P_m(S) = S(m) / N(m),\ P_m(R) = R(m) / N(m),\ P_m(F) = F(m) / N(m), \qquad (1)$$

where $\sum_{m=1,\ldots,M} N(m) = N$, $\sum_{m=1,\ldots,M} R(m) = R$, $\sum_{m=1,\ldots,M} S(m) = S$, $\sum_{m=1,\ldots,M} F(m) = F$.

The response probabilities sum to unity.

Such definition of the response probabilities is valid under the assumption that each fault scenario is equally likely. Since adverse conditions are by nature unexpected, and therefore, it is impossible to predict a fault scenario that will come to realization, this is a justified assumption. However, this is not a requirement of the analysis. If the information is available, the probability of each fault scenario can be incorporated into the analysis. Generally though, such information is proprietary, and therefore, such a case is not considered here.

The three response probabilities completely characterize survivability of a network and can be used to compare the performance of alternative network designs in regard to desirable design features, which include high $P(S)$, low $P(F)$, and a high ratio of $P(S)/P(R)$. The latter is important because a higher ratio indicates a higher ability of a network to withstand damage without requiring reconfiguration. Under adverse events, reconfiguration itself may not be possible or may be too costly, as it requires sharing valuable resources such as control and communication with other vital network functions.

The objective of a network survivability analysis is to determine the network response probabilities.

This requires knowledge of numbers *S*, *R*, and *F* or, equivalently, of numbers $S(m)$, $R(m)$, $F(m)$, and $N(m)$. The total number of fault scenario for a given number of faults can be calculated analytically:

$$N(m) = M!/m!(M-m)! \qquad (2)$$

Numbers $S(m)$, $R(m)$, and $F(m)$ can be calculated analytically only in very small and simple networks (Poroseva et al. 2005). Computational analysis is required for all real-life engineering networks.

3 Computational analysis

3.1 Brute-force algorithm

A brute-force algorithm for calculating the network response probabilities has the structure shown in Fig. 3.

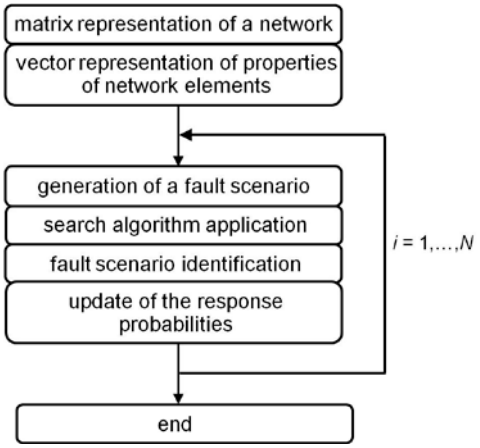


Fig3 Brute-force algorithm for the survivability analysis

The analysis starts by transforming a network into a structured adjacency matrix or an adjacency list (Poroseva et al. 2009). A standard adjacency matrix is a $M \times M$ symmetric matrix **X** that describes the connectivity of M network elements. Here, network elements are the three types of links (see Fig. 2 for an example). If two elements i and j are connected with one another, we say that $X_{i,j} = 1$ and the two elements are adjacent. If elements i and j are not directly connected with one another, then, $X_{i,j} = 0$. Each row (column) of **X** is a complete description of all adjacent and non-adjacent elements.

A structured adjacency matrix (list) is an adjacency matrix where columns (rows) are ordered. That is, positions $i, j = 1,...,VT$ are reserved for VT-links, and $i, j = VT + 1,...,VT + VB$ are kept for VB-links. Here, VT and VB are the numbers for VT- and VB-links in the network, respectively. The remaining elements ($i, j = VT + VB + 1,..., M$) are H-links. The use of a structured matrix (list) reduces computational time required by the search algorithm to establish the connection between sources and sinks after faults (Poroseva et al. 2009).

For representing sparse networks, such as power systems, the use of an adjacency list is recommended over using an adjacency matrix. The adjacency list organizes adjacent elements into lists or sets. That is, for each network element, there exists a corresponding list that contains only elements adjacent to the element. In sparse networks, each set is relatively small. Indeed, if L_i represents the size of the adjacency list of the i-element, then one has to store M vectors of the size $\max[L_i]$ or less, rather than the $M \times M$ matrix. Since typically

$\max[L_i] \ll M$, savings in storage space can be substantial. Further reduction in computational time can be achieved by generating a structured adjacency list, that is, a list with elements ordered as in a structured matrix.

Manual generation of the adjacency matrix (list) for large-scale networks is an exceedingly time-consuming and error-prone process, with no guarantee that the outcome will be correct. Errors left unnoticed will lead to wrong conclusions, and ultimately to costly mistakes in the network design and operation. To avoid this situation, a Java application was developed in Neumayr and Poroseva (2011) to automatically convert a network diagram into a structured adjacency matrix or list.

Different properties of network elements can be represented by a set of vectors: one property, one vector. The vector size corresponds to the number of network elements, *M*. Currently, only one vector is constructed. This vector represents the link capacity and flow direction of a quality of interest through a link (see discussion in Section 2.1). The vector is updated for every fault scenario to reflect faulty elements. If an element becomes unavailable, the property is set to zero for this element. Both vectors, before and after damage, are used to determine the fault scenario type in the "fault scenario identification" step (Fig. 3). The procedure is described in detail in Poroseva et al. (2009).

The following steps of the algorithm – generating a fault scenario, applying the search algorithm, identifying the fault scenario, and updating the response probabilities – are combined into a single algorithmic block.

Within the block, generating a single fault scenario is a challenge in itself (see discussion in Poroseva et al. 2009). The procedure, however, is not computationally expensive.

Once a fault scenario is generated and survived sources are determined, the search algorithm is applied to determine connectivity between operational sources and sinks. Presently, this is accomplished using the standard depth-first search traversal algorithm (see, e.g., Cormen at al. 2001). Although the actual time required to run this algorithm may vary depending on the application, the theoretical upper bound is linear in the graph size, here, *M*.

After connectivity of sources and sinks in the network with faulty elements has been established, a procedure for identifying the network response is

conducted. Neither this procedure nor updating the responses probabilities is computationally expensive.

Notice, that the network response probabilities are re-calculated for each fault scenario. This seems to be somewhat inconsistent with expressions (1). The straightforward use of these expressions is prone to integer overflow for larger networks. Therefore, a recurrence procedure was suggested in Poroseva et al. (2009) to resolve this problem.

These block steps are repeated until all fault scenarios have been analyzed, that is, 2^M times in the deterministic approach. Notable reduction in computational time can be achieved by using the adaptive algorithm (Poroseva et al. 2009), in which the deterministic approach is combined with the Monte Carlo technique. It is difficult, however, to evaluate the computational cost of applying the adaptive algorithm in an arbitrary network because the number of generated fault scenarios depends not only on the number of network elements, but also on the required accuracy of calculations. Specifically, at a given number of faults *m*, the algorithm compares $N(m)$ given by (2) and $n(m) \sim \frac{1}{error^2}$, where "error" is the desired accuracy of calculations. If $n(m) \geq N(m)$, then scenarios are generated deterministically. For example, in the network shown in Fig. 2, applying the adaptive algorithm to calculate the response probabilities with tolerance $error \sim 10^{-4}$ reduces the number of generated fault scenarios from $2^{32} \sim 4.3 \cdot 10^9$ to $2 \cdot 2^{11} + 9 \cdot 10^8 \sim 9 \cdot 10^8$, which is roughly 5 times smaller (or down to $2^{29.7}$).

We will use 2^M as the upper bound of computational time required to cycle the block.

Overall, a rough estimate of the required time by the brute-force algorithm is $T_{BF} \sim M \cdot 2^M$. For the network in Fig. 2, it is $T_{BF} \sim 1.4 \cdot 10^{11}$.

To apply the survivability analysis to large networks with multiple sources and multiple sinks, additional steps to reduce computational complexity of the problem are necessary.

3.2 "Selfish" algorithm

In mainstream network analyses of network robustness (see, e.g., Newman et al. 2006), one can say that a network's performance is considered from an operator's point of view. The operator observes the entire network from the side and values

the network operability as a whole more than the availability of a network to an individual customer. A similar viewpoint was also adopted in our previous studies (Poroseva et al. 2005, 2009) and is the foundation of the brute-force algorithm described above.

A change in the perspective, however, can be very helpful in reducing the computational costs of a network survivability analysis. Indeed, all engineering networks have been designed not for the sake of their existence, but for the specific purpose of serving the needs of customers. Customers pay for the network design, construction, and operation. If their needs are not satisfied, a network will be of no use. Therefore, network survivability should also be considered from the customer's perspective, which is quite different from the operator's.

Specifically, a customer is not interested in the network operability as a whole. All that is important is whether the customer has access to a quality of interest and in an amount sufficient to satisfy the customer's demand. Whether other customers have access to the quality is not a concern. Such a viewpoint eliminates other customers and the multitude of links connecting the customer to available sources. Instead, links are substituted with a few paths between the customer and sources. For obvious reasons, such view on the network operability can be called selfish.

Using this perspective, the network survivability problem for a single customer (sink) is reformulated to establish the number of paths from a given sink to survived sources in a given fault scenario. If the problem is solved simultaneously for all sinks, the whole network state in a given fault scenario can be determined. Notice that this problem is well suited for parallelization and, thus further reducing computational time.

The numerical algorithm structure to solve this problem is shown in Fig. 4. The following sub-sections describe the algorithm in detail.

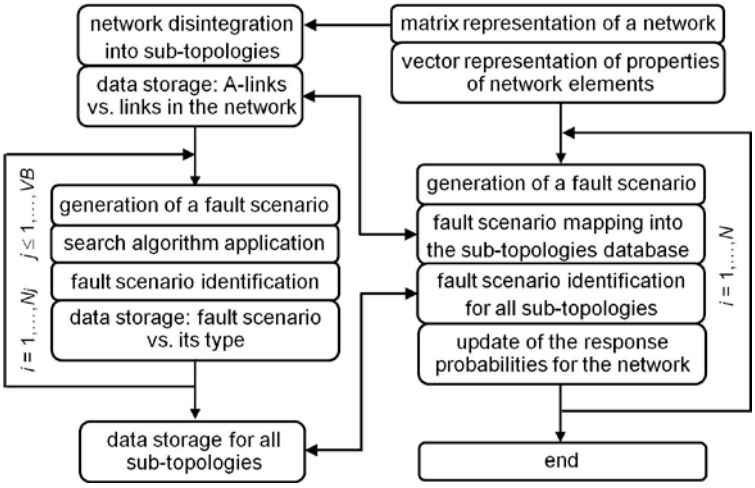


Fig4 "Selfish" algorithm for the survivability analysis. Here, *VB* is the number of the VB-links in the initial network and *Nj* is the number of fault scenarios in the *j*-th sub-topology

3.2.1 Network disintegration

The algorithm starts by disintegrating (or decomposing) the initial network with multiple sources and sinks into a set of simpler networks (or sub-topologies), with each sub-topology containing all sources but a single sink. This step is performed *before* generating fault scenarios for the entire network.

Network disintegration is achieved by first removing all VB-links belonging to other sinks from the initial network. Faults in vertical links connecting other sinks to the network cannot interrupt the flow of a quality of interest to the sink under consideration. Then, in each sub-topology links connected in series are combined into a single link. Since faults in links connected in series are equivalent to a fault in a single link, such a procedure is justified for the survivability analysis. Moreover, it reduces the number of links in a sub-topology.

In the general case, the number of sub-topologies is equal to the number of sinks in the initial network. However, one can expect that for some sinks the network will reduce to the same sub-topology. Therefore, the actual number of sub-topologies under consideration will be much less.

The last step in this procedure is to disintegrate sub-topologies with sinks that are connected to the network through multiple VB-links into sub-topologies with sinks that are connected to the network by a single VB-link. Not only does this step reduce the scale of sub-topologies further, but it will also, most likely, reduce the number of different sub-topologies to analyze.

In the worst-case scenario where no similarity between sub-topologies is found, the final number of sub-topologies is equal to the number of VB-links, that is, *VB*. As only one VB-link is included in a sub-topology, each sub-topology can be uniquely identified by a VB-link.

A general procedure for disintegrating a network with multiple sources and sinks into a set of sub-topologies is shown in Fig. 5.

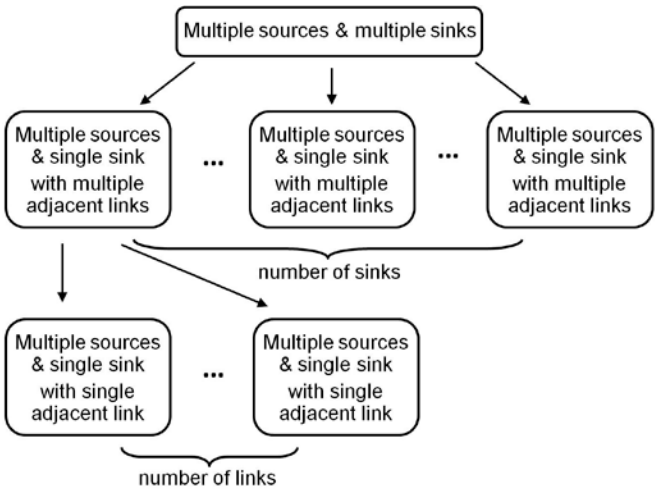
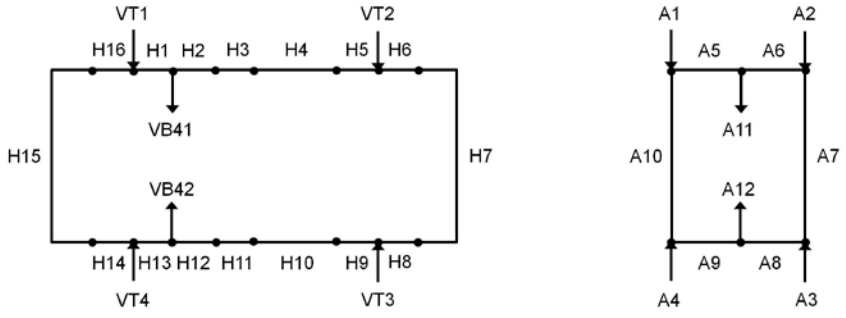


Fig5 The procedure for disintegrating a network with multiple sources and sinks into a set of sub-topologies with multiple sources and a single sink connected to the network by a single link

Let us consider the disintegration of the network in Fig. 2 as an example. Fig. 6a shows the initial step for the sink connected to the network by vertical links VB41 and VB42. Figure 6b shows the sub-topology after combining the links connected in series into corresponding single links.



(a) **(b)**

Fig6 A network sub-topology for a single sink: (a) after removing other sinks, (b) after combining links connected in series into a single link

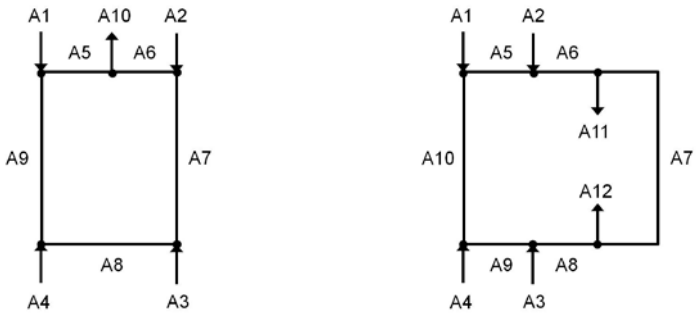
In the figure, the links are re-labeled in such a way that links A1-A4 correspond to links VT1-VT4 and links A11 and A12 correspond to links VB41 and VB42. Links A7 and A10 represent three links each, that is, H6-H8 and H14-H16, respectively. Links A6 and A8 represent four links each: H2-H5 and H9-H12, respectively. There is one-to-one correspondence between links H1 and A5 and H13 and A9. Re-labeling is necessary to simplify the computational analysis, which will be explained later in this section.

Two more sinks "see" the network in Fig. 2 as the sub-topology shown in Fig. 6b. Overall, there are only three unique sub-topologies to consider for the seven sinks in Fig. 2. These sub-topologies are shown in Figs. 6b and 7.

Each of the three sub-topologies contains 12 or fewer elements when compared to the 32 elements in the network in Fig. 2.

Further disintegration of the sub-topologies in Figs. 6b and 7b show that both sub-topologies can be reduced to the sub-topology shown in Fig. 7a.

Thus, instead of 12 sub-topologies (*VB* = 12 in the network in Fig. 2), the network can be reduced to a single sub-topology (Fig. 7a) that contains only 10 links.



(a) **(b)**

Fig7 Additional two sub-topologies for the network in Fig. 2

The correspondence of the A-links in Fig. 7a to the links in the initial network in Fig. 2 depends on the VB-link in a sub-topology. This information is stored in table form (Table 1). In the table, the A10-link is uniquely reserved for a VB-link. For example, if the analyzed sub-topology contains the VB11-link, one finds the "VB11" cell in the A10-column in Table 1. Cells in the row containing "VB11" give information related to the correspondence of the A-links and the links in Fig. 2.

3.2.2 Analysis of sub-topologies

After the initial network is disintegrated and a table similar to Table 1 is generated, the survivability analysis of each sub-topology is conducted separately. That is, for each sub-topology all possible fault scenarios are generated, a search algorithm is applied, and every fault scenario is determined. When a sink is initially connected to the network by multiple links, fault scenarios for each sub-topology where a sink is connected to the network by a single link are analyzed separately. A table is generated for each unique sub-topology with information about each fault scenario and its type. For example, table 2 shows a few rows from the final table for the Fig.7a sub-topology.

Tables for all sub-topologies constitute a database that is created before generating and analyzing fault scenarios for the entire network.

Without taking into account symmetries in the initial network, a rough approximation of the computational time upper bound required for analysis of sub-topologies is $\sum_{j=1,...,VB} M_j 2^{M_j}$, where M_j is the number of links in the *j*-th sub-topology and 2^{M_j} is the number of fault scenarios *Nj* in each sub-topology.

One can show (see Appendix A) that

$$\sum_{j=1,...,VB} M_j 2^{M_j} \le VB \max[M_j] 2^{\max[M_j]} < M 2^M , \qquad (3)$$

for any $1 < VB \le M-1$, $M \ge 3$, and $\max[M_j] \le M-1$. The equality $\sum_{j=1,...,VB} M_j 2^{M_j} = VB \max[M_j] 2^{\max[M_j]}$ holds true when a network is disintegrated into a set of unique sub-topologies that all contain the same number of links ($\max[M_j] = M_j$) and when the number of the sub-topologies is exactly *VB*.

The following criteria determine the limits of the values of *VB*, *M*, and $\max[M_j]$. The algorithm can be applied to a network that has at least 2 VB-links (two sinks each connected by a single link to the network). Thus, $VB > 1$. There is an upper bound on the value of *VB* because a network should have at least one source (VT-link). The value of $\max[M_j]$ cannot exceed $M-1$, which provides another example of a network with two VB-links and no possibility of combining horizontal links after removing one of the VB-links. The number of network

elements *M* cannot be less than three because a network should contain at least two VB-links and one VT-link.

Actual time required for the sub-topologies survivability analysis is expected to be much less than provided by (3). Let us again consider the network in Fig. 2 as an example. As shown previously, only one sub-topology should be analyzed for this network. The total number of fault scenarios in the sub-topology is $2^{10} \sim 10^3$. That is, the domain of the graph search algorithm application is reduced remarkably in comparison to the one in the brute-force algorithm (2^{32}). Thus, $j = 1$ and the time required to generate and analyze all fault scenarios in this sub-topology is $10 \cdot 2^{10} \sim 10^4$, which is quite a change from $T_{BF} \sim 1.4 \cdot 10^{11}$.

3.2.3 Matching fault scenarios in the network and in the sub-topologies

After the sub-topologies survivability analysis is conducted and all relevant data is stored in computer memory, the algorithm returns to the generation and analysis of fault scenarios in the network.

The procedure for generating fault scenarios is the same as in the brute-force algorithm (see Section 3.1). However, after a fault scenario is generated, no search algorithm is used to determine the connectivity of survived sources and sinks in the network. Instead, the fault scenario is matched with fault scenarios already existing in the sub-topologies database. First, one uses a table similar to Table 1 to find the correspondence of faulty links in the initial network to the A-links in each sub-topology.

Once a fault scenario is identified for every sub-topology, tables similar to Table 2 are used to determine a fault scenario type. The matching procedure is not expensive in time and space as all data in the table are ordered.

Let us show how the matching procedure works using the network in Fig. 2. As an example, faults are generated in the VT1, H1, H2, H5, and H7 links of the network. Using Table 1, one can find that these faults in the network's links correspond to faults in A4, A6, and A9 links in the Fig. 7a sub-topology that contains the VB11 vertical link. For the sub-topology containing the VB12 link, faults are in the A4, A5, and A9 links. For the sub-topology containing either the VB21 link or the VB41, the fault scenario is the same: faulty links are A1, A5, A6, and A7. In a similar manner, one matches all rows in Table 1. Next, Table 2 is used to determine the fault scenario type for a sub-topology with a VB-link.

After matching a fault scenario in the network with those in the sub-topologies, the algorithm combines this information i) for each sink connected to the network by multiple VB-links to determine its individual state and ii) for all sinks to determine the state of the entire network (see discussion in Section 2.2). The procedure for updating the network response probabilities is the same as in the brute-force algorithm.

3.2.4 Time requirements

The upper bound of computational time required to conduct the network survivability analysis using the "selfish" algorithm can be obtained as the sum of time required for the analysis of sub-topologies and for generating all fault scenarios, that is, $T_{selfish} \sim \sum_{j=1,...,VB} M_j 2^{M_j} + 2^M \le VB \max[M_j] 2^{\max[M_j]} + 2^M$. Appendix B shows that indeed

$$T_{selfish} < T_{BF} = M 2^M \tag{4}$$

for any $1 < VB \le M-1$, $M > 3$, and $\max[M_j] \le M-1$.

Actual advantages of the "selfish" algorithm in terms of computational time can be much higher. For the network in Fig. 2, a rough estimate gives $T_{selfish} \sim 10 \cdot 2^{10} + 2^{32} \sim 2^{32}$ *vs.* $T_{BF} \sim 32 \cdot 2^{32}$. With an increase in the network scale, the difference in running time required by the two algorithms is expected to be larger.

Conclusions

Computational analysis of survivability of engineering networks with multiple sources and sinks may be, at the very least, categorized as an exponential time problem because the impact of all possible combinations of faults should be analyzed. The brute-force algorithm (Poroseva et al. 2009) uses a graph search algorithm to establish network connectivity in all possible combinations of faults. Connectivity of survived sources and sinks determines a network's response to each combination of faults. The search procedure significantly increases the total computational time required by the brute-force algorithm.

This paper describes a new "selfish" algorithm that reduces computational time required for conducting a network survivability analysis by substituting a search problem, which applies a search procedure to every fault combination in the network, with a decision problem (see Goldreich (2008) for more discussion on search and decision problems). In the decision problem, the network response to a given combination of faults is determined by requesting information from a previously created database.

This database is created by mapping the initial topology of a large-scale network with multiple sources and sinks onto a set of smaller sub-topologies with multiple sources and a single sink connected to the network by a single link. The search algorithm has to be used only on this set of sub-topologies, and the scale of sub-topologies is expected to be much less than that of the initial network. The number of sub-topologies is also expected to be small. In the example considered in this paper, the initial network of 32 elements was reduced to a single sub-topology of 10 elements. Thus, the domain of the application of the search procedure was reduced drastically. As a result, the computational time required by a search procedure was reduced from $\sim 10^{11}$ in the brute-force algorithm to $\sim 10^{4}$ in the new algorithm.

It has also been proven that the total time required by all steps in the new algorithm was reduced. In the example considered in the paper, a reduction linear to the network size was achieved. Actual time-savings are expected to be larger in applications to large-scale networks.

The "selfish" algorithm has a broad application as it can be used for the analysis and design of any engineering network with sources and sinks to maximize their survivability. Application to wireless networks is currently being researched and will be reported elsewhere.

Acknowledgements

The author would like to thank anonymous reviewer #1 for the thorough review and helpful suggestions.

References

Committee on the Societal and Economic Impacts of Severe Space Weather Events: A Workshop, National Research Council (2008) Severe Space Weather Events--Understanding Societal and Economic Impacts: A Workshop Report (2008). The National Academies Press. http://books.nap.edu/openbook.php?record_id=12507&page=1

Cormen TH, Leiserson CE, Rivest RL, and Stein C (2001) Introduction to Algorithms. 2nd edn. MIT Press and McGraw-Hill, pp. 540-549

Garey M, Johnson D (1979) Computers And Intractability: A Guide To The Theory of NP-Completeness. W. H. Freeman, San Francisco

Goldreich O (2008) Computational Complexity: A Conceptual Perspective. Cambridge University Press

Hill J (2005) Survivable Architectures for Vital Systems. In: Proceedings of the ASNE Reconfiguration and Survivability Symposium, Jacksonville, FL, USA, February.

IEEE (2010) "Recommended Practice for 1 to 35kV Medium Voltage DC Power Systems on Ships," IEEE P1709/ Draft D0.8.2, New York, NY, Jan. 2010.

Lancaster J (2000) Engineering Catastrophes : causes and effects of major accidents, CRC

North American Electric Reliability Corporation (2007) Results of the 2007 Survey of Reliability Issues. http://www.nerc.com/files/Reliability_Issue_Survey_Final_Report_Rev.1.pdf

Neumayr D, Poroseva SV (2011) On Development of Computational Tools for Evaluating System Survivability Due to Its Topology. In: Proceedings of the 52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, Denver, USA, April, AIAA-2011-1818

Newman M, Barabasi A-L, Watts DJ (2006) The Structure and Dynamics of Networks. Princeton University Press

The NextGen Energy Council Management Information Services, Inc. (2008) Lights out in 2009? http://www.oe.energy.gov/DocumentsandMedia/Attachment_1_Nextgen_Energy_Council_Lights_Out_Study.pdf.pdf

Oliveto FE (1998) System efficiency/merit. In: Proceedings of the IEEE 1998 NAECON, Dayton, OH, USA, July: 51-59

Papadimitriou CH (1994) Computational Complexity. Addison-Wesley

Poroseva SV, Woodruff S L, Hussaini MY (2005) Topology of the generator bus in a warship integrated power system. In: Proceedings of the IEEE Electric Ship Technologies Symposium, Philadelphia, PA, USA, July: 141-148

Poroseva SV, Lay N., Hussaini MY (2009) Algorithm Development for Evaluating the IPS Survivability due to its Topology. In: Proceedings of the IEEE Electric Ship Technologies Symposium, Baltimore, MD, USA, April: 253-260

Poroseva SV (2010a) Computational analysis of network survivability with application to power systems. Physics Procedia 4:113-117.

Poroseva SV (2010b) Designing Power System Topologies of Enhanced Survivability. In: Proceedings of the 51st AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Orlando, FL, USA, April, AIAA-2010-2572

President’s Commission on Critical Infrastructure Protection, Critical Foundations (1997) Protecting Americas Infrastructures.http://www.fas.org/sgp/library/pccip.pdf

Saleh A JH, Castet J-F (2011) Spacecraft Reliability and Multi-State Failures. John Wiley & Sons, Ch. 8.2

U.S. DOE (2002) Transmission Grid. http://www.pi.energy.gov/documents/TransmissionGrid.pdf

U.S. DOE (2005) Overview of the Electric Grid. http://sites.energetics.com/gridworks/grid.html

Appendix A

Let us show that indeed $\sum_{j=1,\ldots,VB} M_j 2^{M_j} \le VB \max[M_j] 2^{\max[M_j]} < M 2^M$. To prove it is to show that $VB \max[M_j] 2^{\max[M_j]} < M 2^M$, when $1 < VB \le M-1$, $M \ge 3$, and $\max[M_j] \le M-1$. The values of VB and $\max[M_j]$ are not independent from one another. The relation between them is given by the following expression: $\max[M_j] \le M - VB + 1$. That gives us

$$VB \max[M_j] 2^{\max[M_j]} \le VB \cdot (M - VB + 1) 2^{M-VB+1}.$$

The next step is to show that $VB \cdot (M - VB + 1) 2^{M-VB+1} < M 2^M$. Dividing both sides of this expression by $M 2^M$ results in

$$\frac{VB \cdot (M - VB + 1)}{M \cdot 2^{VB-1}} = \frac{VB}{2^{VB-1}} \left(1 - \frac{VB-1}{M}\right) < 1.$$

Substitution of $x = VB - 1$ into this expression gives

$$\frac{x+1}{2^x} \cdot \left(1 - \frac{x}{M}\right) < 1.$$

Function $(x+1)/2^x$ is monotonically decreasing from 1 to $(M-1)/2^{M-2}$ with *x* increasing from 1 to $M-2$. Because *M* is never less than 3, the value of this function is 1 or less.

The value of *x* is always less than *M*, therefore expression $(1 - x/M)$ is always less than 1.

Thus, the product of $(x+1)/2^x$ and $(1 - x/M)$ is always less than 1, that is, indeed

$$\frac{x+1}{2^x} \cdot \left(1 - \frac{x}{M}\right) < 1.$$

Appendix B

To prove that (4) holds true is to show that

$$T_{selfish} \sim \sum_{j=1,\ldots,VB} M_j 2^{M_j} + 2^M \le VB \max[M_j] 2^{\max[M_j]} + 2^M < M 2^M .$$

We will use an approach similar to the one in Appendix A. That is, let us first make a substitution of $\max[M_j]$ with $\max[M_j] \le M - VB + 1$:

$$VB \cdot (M - VB + 1) 2^{M-VB+1} + 2^M < M 2^M .$$

After dividing both sides of this expression by 2^M

$$\frac{VB \cdot (M - VB + 1)}{2^{VB-1}} + 1 < M$$

and moving 1 from the left side to the right, one has to prove that

$$\frac{VB \cdot (M - VB + 1)}{2^{VB-1}} < M - 1 .$$

After two substitutions, such as, $x = VB - 1$ and $a = M - 1$, the final expression takes the following form

$$\frac{(x+1) \cdot (a + 1 - x)}{2^x} < a .$$

Dividing both sides of this expression by *a* gives

$$\frac{(x+1)}{2^x}\left(1 + \frac{1-x}{a}\right) < 1. \qquad (5)$$

It was shown in Appendix A that function $(x+1)/2^x$ is monotonically decreasing from 1 to $(M-1)/2^{M-2}$ with x increasing from 1 to $M-2$. Because M is never less than 3, the value of this function is 1, when $M = 3$, or less than 1 when $M > 3$.

Let us discuss the behavior of term $(1-x)/a$. This term is equal to zero when x is equal to 1 and becomes negative with x increasing. Therefore, term $\left(1+(1-x)/a\right)$ is equal to 1 at $M = 3$ ($VB = 2$) and is less than 1 for all other values of x.

Combining the properties of both terms – $(x+1)/2^x$ and $\left(1+(1-x)/a\right)$ –, one can conclude that the product of both terms is equal to 1 when $M = 3$ or less than 1 for $M > 3$.

Thus, expressions (5) and (4) are satisfied when $M > 3$. Clearly, the case of $M = 3$ (one source with two adjacent VB-links) is not of importance in designing engineering networks. Therefore, we conclude that expression (4) holds true for the purposes of the survivability analysis of engineering networks.

Figure legends

Fig1 Initial network representation for the survivability analysis

Fig2 Final network representation for the survivability analysis

Fig3 Brute-force algorithm for the survivability analysis

Fig4 "Selfish" algorithm for the survivability analysis. Here, *VB* is the number of the VB-links in the initial network and *Nj* is the number of fault scenarios in the *j*-th sub-topology

Fig5 The procedure of disintegrating a network with multiple sources and sinks into a set of sub-topologies with multiple sources and a single sink connected to the network by a single link

Fig6 A network sub-topology for a single sink: (a) after removing other sinks, (b) after combining links connected in series into a single link

Fig7 Additional two sub-topologies for the network in Fig. 2

Tables

Table 1 Correspondence of the A-links in the sub-topology in Fig. 7a (the top row) to the links in Fig. 2 for different sinks connected to the network by one of the twelve VB-links. In the sub-topology, the A10-link is uniquely reserved for the VB-links.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10
VT2	VT3	VT4	VT1	H6	H7,H8	H9-H13	H14-H16	H1-H5	VB11
VT2	VT3	VT4	VT1	H6,H7	H8	H9-H13	H14-H16	H1-H5	VB12
VT1	VT2	VT3	VT4	H1-H4	H5	H6-H8	H9-H13	H14-H16	VB21
VT4	VT3	VT2	VT1	H10-H13	H9	H6-H8	H1-H5	H14-H16	VB22
VT1	VT2	VT3	VT4	H1-H3	H5	H6-H8	H9-H13	H14-H16	VB31
VT4	VT3	VT2	VT1	H11-H13	H9,H10	H6-H8	H1-H5	H14-H16	VB32
VT1	VT2	VT3	VT4	H1	H2-H5	H6-H8	H9-H13	H14-H16	VB41
VT4	VT3	VT2	VT1	H13	H9-H12	H6-H8	H1-H5	H14-H16	VB42
VT4	VT1	VT2	VT3	H14,H15	H16	H1-H5	H6-H8	H9-H13	VB51
VT4	VT1	VT2	VT3	H14	H15,H16	H1-H5	H6-H8	H9-H13	VB52
VT1	VT2	VT3	VT4	H1,H2	H3-H5	H6-H8	H9-H13	H14-H16	VB6
VT4	VT3	VT2	VT1	H12,H13	H9-H11	H6-H8	H1-H5	H14-H16	VB7

Table 2 An example of the database for fault scenarios and their types in the sub-topology shown in Fig.7a. Total number of fault scenarios is 1024, only three of them are shown. Links inside {…} are those with faults; *S* and *F* stand for "no response" and failure scenarios, respectively. Here, to determine a fault scenario type, it is assumed that a single source can completely satisfy the sink's demand. Generally, identification of a fault scenario type should be based on parameters of a network under consideration.

Fault scenario	Type
{A4, A10}	F
{A3, A5, A9}	S
{A5, A6}	F
…..	…..