Improving the Survivability of Interdependent Networks by Restructuring Dependencies

Genya Ishigaki, Riti Gour, and Jason P. Jue
Department of Computer Science, The University of Texas at Dallas, Richardson, Texas 75080, USA
Email: {gishigaki, rgour, jjue}@utdalls.edu

***Abstract*—This paper studies a network design problem to improve the survivability of interdependent networks by restructuring the dependencies. As different types of networked systems become more integrated, the relation between distinct kinds of network devices has become more intertwined. In order to guarantee the robustness of such systems, survivability problems of interdependent networks must be addressed. A characteristic of the proposed algorithm is that the continuous availability of the entire system is guaranteed by the preservation of certain structures in the original networks during the restructuring process. Simulation results demonstrate that the restructuring heuristic can substantially enhance the survivability of interdependent networks.**

***Keywords*—interdependent networks; survivability; cascading failure; cyber-physical systems.**

I. Introduction

Recent infrastructure or cyber-physical systems are likely to demonstrate greater integrations with other types of systems in order to provide more intelligent and flexible operations. A recent study indicates that different kinds of infrastructures rely upon each other in complex manners [1]. A typical example is smart electricity grids, which exploit a computer network to control an electricity network for efficient control and management.

Additionally, the concept of Anything as a Service (XaaS), which promotes delivery of services to users without revealing physical or implementation details, has accelerated the amount of layering and obscure dependencies in networks. This tendency is likely to be more evident for next-generation network systems.

However, it has been revealed that certain types of dependencies between different networks can deteriorate robustness of the entire tangled systems [2]. Consecutive multiple failure phenomena called *cascading failures* exemplify the unique fragility of such systems. In networks without interdependencies, a failure would influence a certain part of a network. Nonetheless, in networks with interdependencies between layers, some nodes that are not directly connected to the failed portion can become nonfunctional because of the loss of service provisioning from nodes in other layers, which are directly influenced by the initial failure. Fig. 1-2 show the beginning phase of such a cascading failure, which starts at the failure of a single node v_2 and results in the entire network failure. Since node v_1 loses its supporting node (v_2), it becomes nonfunctional. This induces another loss of the supporting node of v'_2, and eventually the single node failure at v_2 causes failure of the whole network. In fact, It has been reported that a cause of the nation-wide blackout in Italy in 2003 was a cascading failure induced by the dependencies between the electricity network and control information network [3].

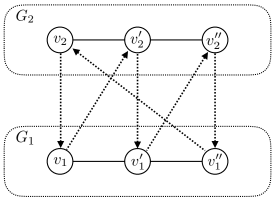

Fig. 1. An interdependent network with 2 constituent graphs.

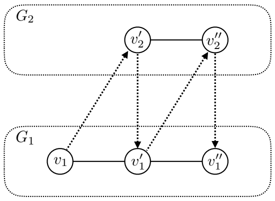

Fig. 2. Initial failure at node v_2 causing a cascading failure.

In order to understand the characteristics of such networks with complex dependencies, many contributions have been made since the first theoretical proposal on the cascading failure model by Buldyrev et al. in 2010 [4]. The pioneering work [4] focuses on analyzing the behavior of cascading failures rather than proposing design strategies. In contrast, some following works try to identify vulnerable topologies in interdependent networks to avoid such fragile structures in the design phase [5], [6]. Furthermore, other works propose design strategies in more realistic models to consider the impact of failures caused by a single component [7], integrated factors within and between layers [8], or the heterogeneity of nodes in each layer [9].

This paper discusses a design problem for interdependent networks to improve their survivability, which is a measure of the robustness against a whole network failure, by modifying an existing network topology. The contribution that contrasts our work with other related works is the consideration of existing network facilities. Our method tries to redesign only a relatively small part of the existing network to enhance the survivability so that the entire network remains operational even during the restructuring process. In order to realize this continuous availability, the special type of dependencies whose removal does not influence the functionality of the entire system is identified in the first step of our restructuring method. A heuristic to decide the relocations of these dependencies is also proposed. Our method is evaluated by the simulations in different pseudo interdependent networks.

II. Related Works

The work in [10] analyzes the survivability of interdependent networks. The authors propose heuristic and ILP based methods to approximate the survivability. The work focuses on defining and estimating the survivability, and does not discuss the design aspect of interdependent networks. Therefore, our work proposes a method improving the survivability, adopting the definition discussed in this related work.

The work in [5] identifies the influence of certain topological characteristics on cascading failures. A method to evaluate the importance of nodes for network robustness is proposed in [6] by introducing a boolean algebraic representation. The work in [8] considers dependency relations not only between layers but also within a single layer. In [9], the heterogeneity of nodes in each network is taken into account. Zhao et al. [7] formulate an optimization problem enhancing the system robustness using an ILP.

The existing works on designing interdependent networks [5]–[9] assume that an entire network is designed and constructed at the same time, though it seems difficult to redesign all the systems simultaneously in practical infrastructure networks. Thus, our paper is aimed at propounding a method to redesign a portion of an existing network so that the network can be functional even during the redesign. In general, an infrastructure network is amended over time and needs to work with the integration of new and legacy facilities. Our improvement scheme would reduce the cost of survivability improvement in contrast to the entire reconstruction of the systems.

III. Modeling and Motivating Example

This section explains a mathematical model to discuss interdependent networks and presents a motivating example of our method. Section III-B summarizes a related work [10] defining the survivability for interdependent networks, which we adopt to evaluate the networks.

A. Network Model

An interdependent network consists of k constituent graphs $G_i = (V_i, E_{ii})$ $(1 \leq i \leq k)$ that have interdependency relationships which are defined by sets of (directed) arcs A_{ij} $(1 \leq i, j \leq k,\ i \neq j)$ representing the dependency relationship between a pair of nodes in different graphs. Edges in $E_{ii} \subseteq V_i \times V_i$ are called *intra*-edges because they connect pairs of nodes in a same network. In contrast, arcs in $A_{ij} \subseteq V_i \times V_j$ $(i \neq j)$ called *inter-* or *dependency* arcs. If there exists an arc $(v_i, v_j) \in A_{ij}$ $(v_i \in V_i, v_j \in V_j)$, it means that a node v_j has dependency on a node v_i. Then, the node v_i is called the *supporting* node, and v_j is a supported node. A node v is said to be *functional* iff it has at least one functional supporting node.

In order to emphasize the dependency between constituent graphs, an interdependent network can be represented as a single layer directed graph $G = (V, A)$, where $V := \bigcup_i V_i$, and $A := \bigcup_{\{(i,j) | i \neq j\}} A_{ij}$ by abbreviating intra-edges. With this notation, a node v is said to be *functional* iff the node v satisfies $\deg_{\text{in}}(v) \geq 1$. Note that all the discussions in the rest of this paper follow this single layer graph representation.

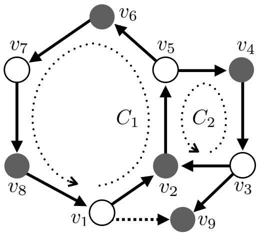

Fig. 3. Graph G with (v_1, v_9).

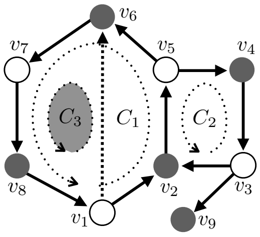

Fig. 4. Graph G' with (v_1, v_6).

B. Survivability of Interdependent Networks

Parandehgheibi et al. [10] propose an index that quantifies the survivability of interdependent networks against cascading failures exploiting the *cycle hitting set*. They prove that a graph needs to have at least one directed cycle in order to maintain some functional nodes; in other words, the existence of one cycle prevents an interdependent network from its entire failure. Thus, the survivability of interdependent networks is evaluated based on the cardinality of the minimum cycle hitting set whose removal makes the corresponding graph acyclic. Formally, a cycle hitting set H is a set of nodes such that any cycle $C = (V(C), E(C))$ in a given graph $G = (V, A)$ has at least one node in the hitting set:

$$H(G) := \{v \mid \forall C \in \mathcal{C}(G)\ \exists v \in V \text{ s.t. } v \in V(C)\}, \qquad (1)$$

where $\mathcal{C}(G)$ is the set of all cycles in the given graph.

C. Motivating Example

Adopting the survivability definition shown above, improvement of survivability would be equivalent to the increase in the number of independent cycles in a graph. Fig. 3 and 4 show an example comparing two similar interdependent networks.

In graph G in Fig. 3, there exists two cycles (C_1, C_2). If either v_2 or v_5, which are in both $V(C_1)$ and $V(C_2)$, becomes nonfunctional because of a failure, all the nodes in G eventually lose their supporting nodes and become nonfunctional. On the other hand, a cycle C_3 is resilient to the failure at node v_2 or v_5 in G' in Fig. 4. Therefore, the graph G' is more survivable than G: $1 = |H(G)| < |H(G')| = 2$ although they differ only in the destination node of one dependency arc ((v_1, v_9) or (v_1, v_6)). Supposing that G is an existing topology of a network, a method that relocates (v_1, v_9) to (v_1, v_6) can achieve the enhancement of the survivability.

IV. Problem Formulation

A. Assumptions

This paper deals with the case in which interdependent networks have two types of homogeneous constituent networks with identical dependencies ($k = 2$). However, our discussion with the restriction on k can be easily extended to more general cases. Also, in advanced network models, each constituent network can have different types of nodes, such as generating and relay nodes, which are independently functional and need provisions from a generating node via paths of intra-edges, respectively [9]. Nevertheless, for simplicity, this work follows the assumption in [10] that each node in a constituent network is directly connected to a single conceptual generating node that has 100% reliability (homogeneous constituent graphs). Moreover, it is assumed that each supporting node provides a unit amount of support that is enough for a supported node to be operational (identical dependencies), following the same model in [10].

The definition of the survivability focuses on whether or not at least a small portion of a network is alive after failures, so the range or size of the functional network after failures is out of the scope of this paper. Therefore, in minimal cases, the surviving network may consist of only two functional nodes that support each other.

B. Requirement Specification

One aspect contrasting our work to other works is propounding a method to improve the survivability of existing interdependent networks by changing some topological structures, whereas other works assume the case of rebuilding the entire network topology. Because interdependent networks are likely to appear in cyber-physical systems, such as smart infrastructure networks, it seems difficult to redesign all the dependencies. Thus, our strategy of restructuring a portion of networks would have an advantage in practical applications.

However, some constraints are required during our restructuring process due to the existing systems: continuous network availability and supporting node capability. Because the existing network must remain available even during the relocations of dependency relations, it is necessary to avoid the loss of all supporting nodes for any node at any stage of the restructuring. This implies two rules (sufficient and necessary condition) for the live restructuring.

1) The existing cycles need to remain the same after the restructuring.
2) Every node maintains at least one incoming dependency arc at any stage of the restructuring.

Rule 1 is straightforward because some node can lose their supporting node if a cycle is removed from the network. Moreover, such a removal can induce cascading failure. Rule 2 is a formal definition for a node to be functional at any stage of the topology modification.

In addition to guaranteeing the continuous availability, a number of provisions by each supporting node should remain the same after the restructuring in order to consider the capability of each node. The capability could be, for example, the limit on electricity generation, calculation performance, or the number of ports available. In formal representation, $\deg_{\text{out}}(v)_G = \deg_{\text{out}}(v)_{G'}$ for all $v \in V$, where G is an original graph, and G' is the graph obtained by the dependency restructuring.

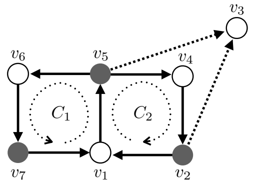

Fig. 5. Original graph G with Marginal Arcs (v_2, v_3) and (v_5, v_3).

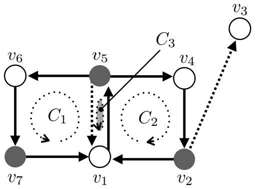

Fig. 6. Modified graph G' with a new arc (v_5, v_1).

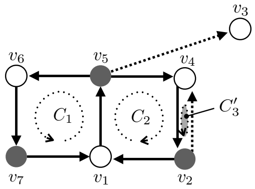

Fig. 7. Modified graph G'' with a new arc (v_2, v_4).

C. Restructuring of Dependencies

In order to follow the constraints, it is necessary to classify the dependency arcs into either changeable or fixed arcs. Rule 1 in Section IV-B regulates the relocation of the dependency arcs in any directed cycles. Thus, let the arcs that are not in any cycles in a given directed graph $G = (V, A)$ be called *Marginal Arcs* (MAs). Formally, the set $M \subsetneq A$ of MAs is defined as

$$M := \{(u, v) \mid (u, v) \notin A(C)\ \forall C \in \mathcal{C}(G)\}. \qquad (2)$$

Lemma 1. A removal of any marginal arc never decreases the survivability of interdependent networks: $|H(G)| \leq |H(\overline{G})|$, where G is a given graph, and $\overline{G}$ is the graph obtained by the removal.

Proof sketch. Let M be a set of marginal arcs. From the definition of MAs (Eq. (2)), the removal of MAs does not destroy or connect any existing cycles in $G = (V, A)$. Therefore, $|H(G)| = |H(\overline{G})|$, where $\overline{G} = (V, A \setminus M)$. □

Moreover, appropriate relocations of the removed MAs could improve the survivability of interdependent networks, assuring operability during the relocation process and provisioning capability of each node. Let us analyze the effect of dependency relocations using simple examples in Fig. 5-7. The given graph G in Fig. 5 has two marginal arcs: $M = \{(v_2, v_3), (v_5, v_3)\}$. In order to maintain at least one supporting node for v_3, one of the MAs has to remain the same, and the other can be relocated. Fig. 6 shows the case of relocating (v_5, v_3) to (v_5, v_1); on the other hand, Fig. 7 indicates the case of relocation of (v_2, v_3) to (v_2, v_4). Even though one new cycle (C_3 and C_3' respectively) is formed by the relocation, the modified graphs G' and G'' have different survivability: $|H(G')| = 1$ $(= H(G))$, and $|H(G'')| = 2$. This is because the cycles in G' are not disjoint with each other: $V(C_1) \cap V(C_2) \cap V(C_3') \neq \varnothing$; in contrast, $V(C_1) \cap V(C_2) \cap V(C_3'') = \varnothing$ in G''. Therefore, it could be said that the appropriate relocation for survivability improvement is forming disjoint cycles.

D. ΔH Problem

This section formulates the ΔH problem, which aims for enhancement of the survivability of a given interdependent network by restructuring dependency relationships, considering the continuous availability and supporting capability of each node.

Problem 1 (ΔH Problem)**.** For a given $G = (V, A)$ and a set of MAs $M \subset A$, maximize $\Delta H := |H(G')| - |H(G)|$, where $G' = (V, A')$ is obtained by the relocation of destinations of the arcs in M, satisfying $\deg_{\text{out}}(v)_G = \deg_{\text{out}}(v)_{G'}$ and $\deg_{\text{in}}(v) \geq 1$ for all $v \in V$.

The ΔH problem contains a subproblem: for a given constant L and $G'' = (V, A'')$ where A'' is decided by a given MA assignment, is ΔH larger than L? This subproblem is NP-complete because of the NP-completeness of the hitting set problem in bipartite graphs [11].

V. Heuristic for ΔH Problem

This section proposes a heuristic for the ΔH problem, which consists of three algorithms: Find-MAs, ΔH, and Minimal-add algorithms. The Find-MAs algorithm enumerates all the arcs that match the definition of MAs (Eq. (2)). With the set of MAs, the ΔH algorithm decides appropriate relocations of each dependency arcs in the set, considering disjointness of newly formed cycles, so that it can improve the network survivability. The Minimal-add algorithm also determines new destinations of MAs that cannot be relocated by the ΔH algorithm due to the absence of locations realizing the disjointness of cycles.

A. Find-MAs Algorithm

The Find-MAs algorithm first distinguishes MAs M, which are candidate arcs for relocations, and the arcs in directed cycles in a given graph $G = (V, A)$, employing Johnson's algorithm, which enumerates all elementary cycles in a directed graph [12]. It is enough for distinguishing MAs to obtain elementary directed cycles because any non-elementary cycle can be divided into multiple elementary cycles. After the enumeration of cycles in G by Johnson's algorithm, the set of MAs is obtained by $M \leftarrow A \setminus \bigcup_{C \in \mathcal{C}(G)} A(C)$.

B. ΔH Algorithm

With the set of MAs obtained by Johnson's algorithm, the ΔH algorithm (shown in Algorithm 1) relocates the destinations of MAs, considering disjointness of newly created cycles. (See Section IV-C.) For each MA (v, w), our algorithm first checks whether or not the relocation of this MA does not cause the loss of supports for the current destination (line 3).

Algorithm 1 ΔH-algorithm(G, l)

```
Input: interdependent network (directed graph) $G = (V, A)$,
    maximum hop $l \in \mathbb{N}$ (odd)
 1: $M \leftarrow$ find-MAs$(G)$        # $M \subset A$
 2: for each $(v, w) \in M$ do
 3:   if $\deg_{\text{in}}(w) \geq 1$ after $A \setminus \{(v, w)\}$ then
 4:     while True do
 5:       pick $C \in \mathcal{C}(v)$ (randomly)
 6:       for $i \leftarrow l$; $i > 0$; $i \leftarrow i - 2$ do
 7:         pick $u \in V(C) : \overline{d_C}(v, u) = i$
 8:         if $u \notin U$ then
 9:           $A \leftarrow A \setminus (v, w) \cup (v, u)$
10:           $U \leftarrow U \cup \{n \mid \overline{d_C}(v, n) \leq i\}$
11:           break to next arc in $M$ (line 2)
12:         end if
13:       end for
14:     end while
15:     Minimal-add$(G$, $(v, w))$
16:   end if
17: end for
```

If w still has some supporting node after the removal of (v, w), the next step is determining a new destination for $(v, \cdot)$. Our algorithm randomly selects one of the cycles that contains the source v denoted by $C \in \mathcal{C}(v)$ (line 5). There can exist some possible candidate nodes for a new destination in the cycle C. Thus, the new destination is decided by the size of a newly formed cycle, which is a result of the relocation (line 6, 7). To represent the size of the newly formed cycle, the distance from a node v to a node u in an (existing) cycle C in the counter direction is denoted as $\overline{d_C}(u, v)$ in our pseudo code. When the maximum hop is designated by l, the algorithm tries to make a new cycle with size $l + 1$. If it fails to form the cycle, it decreases the size by 2, which is the closest location of a same type node in C. Let us think about an example using a given graph G shown in Fig. 3 and the restructured graph in Fig. 4. Since the removal of (v_1, v_9) does not make v_9 lose all the incoming dependency arcs for it, our algorithm tries to relocate the destination of this arc to one of the nodes in the cycle C_1, which are v_2, v_6, v_8. For instance, $|V(C_3)| = 3 + 1$, choosing v_6 by $l = 3$; contrarily, $|V(C_3')| = 2$ if selecting v_8 by $l = 1$.

After selecting a destination candidate u in line 7, our algorithm checks if u is already used to create a new cycle (line 8). This is confirmed by a set of nodes U storing all the nodes that are in newly formed cycles: $\{n \mid \overline{d_C}(v, n) \leq i\}$ (line 10). For instance in Fig. 4, $U \leftarrow U \cup \{v_1, v_6, v_7, v_8\}$. As will be understood, when another MA tries to form a new cycle using one of these nodes in U, the new cycle and C_3 share some node, which means that those cycles are not disjoint with each other. Also, the arc set A is updated when the new destination is finally fixed (line 9). If there exists no possible destination for an MA that satisfies all the conditions, the MA is delegated to the Minimal-add algorithm (line 15).

C. Minimal-add Algorithm

The Minimal-add algorithm (shown in Algorithm 2) deals with the arcs for which the ΔH algorithm cannot find any destination. The edges satisfy either of the following cases: 1) The node v does not belong to any cycles: $\mathcal{C}(v) = \varnothing$, or 2) All the nodes in the cycles of $\mathcal{C}(v)$ are already used to compose new cycles by other MAs.

D. Complexity Analysis

The complexity of our heuristic is sensitive to the number of cycles in an input interdependent network. It is know that Johnson's algorithm finds all elementary cycles within $O((|V| + |E|)(|\mathcal{C}(G)| + 1))$. The ΔH-algorithm determines a new destination after $\frac{l}{2} \times \mathcal{C}(G)$ iterations for each MA, in the worst case. When only one cycle whose size is 2 exists in the input and the other nodes are supported by the cycle, the size of the set M becomes $|E| - 2$. It is obvious that the complexity of the Minimal-add algorithm is $O(1)$, so the worst case analysis takes the case where all MAs are reallocated by the ΔH-algorithm. Thus, $O((|V| + |E|)(|\mathcal{C}(G)| + 1)) + O((|E| - 2)(\lceil \frac{l}{2} \rceil \times |\mathcal{C}(G)|))$. Assuming the maximum hop l is small enough to be considered as a constant, the overall complexity of our heuristic becomes $O((|V| + |E|)|\mathcal{C}(G)|)$. Note that the assumption on l is valid with our strategy, which tries to increase disjoint directed cycles in a given graph.

VI. Simulation

A. Network Topology

The performance of the proposed algorithm is analyzed in random directed bipartite graphs that contain at least one directed cycles. Assuming the situation in which a current interdependent network is normally working, each node is either a member of some cycle or reachable from a node in a cycle through some directed path in an input graph. Because our algorithm only concerns the dependency arcs, any interdependent network is represented as a directed bipartite graph whose arcs connect a pair of different types of nodes.

Each random bipartite graph is generated by specifying the following parameters: $|V_i|$, l, $\max_{v \in V} \deg_{\text{in}}(v)$ and $\min_{v \in V} \deg_{\text{in}}(v)$. In order to observe the performance in different conditions, experiments are conducted in interdependent networks whose constituent graphs have identical number of nodes: $|V_1| = |V_2|$. Also, some variations in relative differences between the size of each graph are also taken into account: $|V_1| = \frac{|V_2|}{q}$ $(q \in \mathbb{Z}_{>0})$. The degree of each node is determined based on the uniform distribution between the given maximum and minimum incoming degree.

B. Evaluation

The survivability of the given, restructured and randomly reassigned interdependent networks are illustrated in our results. In addition to the survivability of the restructured graphs found by our algorithm, the results show the survivability of networks whose MAs are randomly relocated with the uniform distribution over all the nodes in the other constituent graph from the constituent graph that includes the source of an MA.

However, calculating the size of the cycle hitting set is know as NP-complete even in bipartite graphs. Our evaluation is conducted using a well-known approximation algorithm whose approximation factor is $\ln |V| + 1$ [13].

Furthermore, the density of a given graph $G = (V, A)$ defined by $\frac{|A|}{\prod_i |V_i|}$ is used to examine the relationship between the survivability improvement, and the given maximum and minimum degrees.

Algorithm 2 Minimal-add(G, (v, w))

```
Input: interdependent network (directed graph) $G = (V, A)$,
    an arc $(v, w)$
pick $(u, v) \in A_{\text{in}}(v)$ (randomly)
$A \leftarrow A \setminus (v, w) \cup (v, u)$
```

C. Results

Fig. 8 and 9 illustrate the survivability of the given and restructured graphs with identical and halved size constituent graphs, respectively. In both cases, our method demonstrates greater improvements of the survivability compared to the random assignment. The survivability of the original graphs $|H(G)|$ maintains a similar value regardless of the size of graphs, though the survivability of the graphs restructured by our method $|H(G')|$ steeply increases along with the size of the graph. Since, in the original graph, arcs are randomly added, it could be difficult to form larger directed cycles in the original graphs G. Therefore, it is reasonable that the number of disjoint cycles indicates the tendency to stay within a similar range of values. On the other hand, there would exist more MAs in larger graphs, because these graphs have more arcs that are not in directed cycles. This results in dramatic enhancement of the survivability in larger graphs. The difference caused by the given maximum hop l for our algorithm remains small over all sizes of a graph.

Fig. 10 indicates the relationship between the density of graphs and $\Delta H := |H(G)| - |H(G')|$, the amount of survivability improvement by our method and the random reassignment. In networks with lower density, our method succeeds in increasing the survivability. An observed general trend of our method is the gradual decrease in ΔH in accordance with the density. This trend seems to be induced by the fact that the graphs with more arcs have a higher possibility of composing cycles even in the original topology. This implies that graphs with higher density have fewer MAs that can form new disjoint cycles. On the other hand, the random reassignment does not demonstrate its effectiveness for the improvement in graphs with any density, which is the same result from Fig. 8 and 9. Moreover, the random assignment sometimes decreases the survivability ($\Delta H < 0$). It is conceivable that the reassignment connects two (or more) existing disjoint cycles and make it possible to decompose all these cycles by the removal of a node. This result implies that imprudent restructuring of the dependencies would cause more fragility of interdependent networks.

VII. Discussions

One interesting fact is that the results are not influenced much by the maximum hop l. In [14], it is discussed, as a motivating example, that multiple smaller directed cycles tend to increase the robustness because of their independence. However, in terms of total failures, small changes of l are not likely to impact the survivability unless the changes violate the disjointness of newly formed cycles.

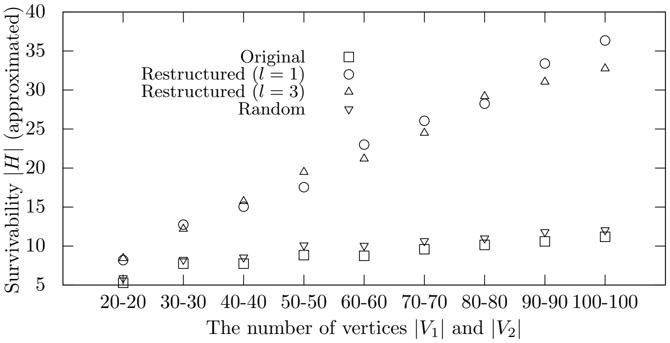

Fig. 8. Survivability of Interdependent Networks Before/After the improvement under $|V_1| = |V_2|$, $\max_{v \in V} \deg_{\text{in}}(v) = 4$, and $\min_{v \in V} \deg_{\text{in}}(v) = 2$, and $l = 1, 3$.

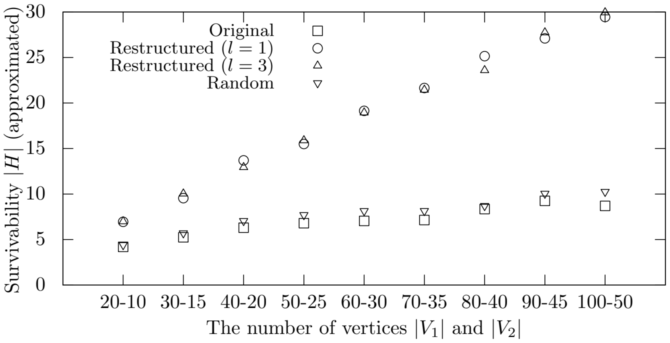

Fig. 9. Survivability of Interdependent Networks Before/After the improvement under $|V_1| = \frac{|V_2|}{2}$, $\max_{v \in V} \deg_{\text{in}}(v) = 4$, $\min_{v \in V} \deg_{\text{in}}(v) = 2$, and $l = 1, 3$.

Even though our work does not aim to reduce the range of a cascading failure as mentioned in Section IV-A, it is also observed that the increase in the average number of failed nodes after a single node failure is suppressed approximately within 0.15 nodes with any parameter settings. For instance, in interdependent networks with $|V_1| = |V_2| = 50$ in Fig. 8, the average number of failed nodes approximately increases from 0.348 to 0.475 by our restructuring. Thus, it could be said that our method does not much deteriorate robustness of the networks with respect to the scale of single node failures.

Additionally, our method seems simple enough to be implemented in the clustered cyber-physical system model discussed in [8]. This model exploits node clustering where nodes within a certain cluster can have dependency only with the nodes in the specific clusters in order to capture the physical limitations such as distance and administrative issues of infrastructure networks. When multiple companies collaboratively conduct operations of an entire system, the domain of which each company takes care should be independent from the others. Extracting a subgraph corresponding to the operation area of each company, our method can find MAs for each domain while it could reduce chances of more relocations.

VIII. Conclusion

This paper addresses the design problem of survivable interdependent networks under some constraints relating to the existence of legacy systems during restructuring. Based on the definition of the survivability proposed in a related work, it is claimed that the increase of disjoint cycles could enhance the survivability. The proposed heuristic tries to compose new disjoint cycles by distinguishing the dependencies whose relocations do not influence ongoing operations of existing systems. Our simulations indicate that the proposed heuristic succeeds in increasing the survivability especially in networks with fewer dependencies.

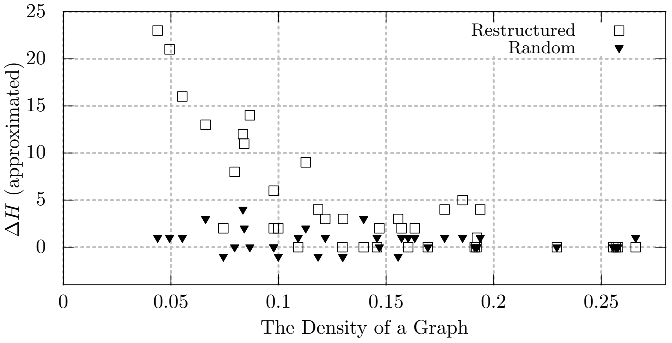

Fig. 10. The Relationship between Graph density and ΔH.

References

[1] M. Ouyang, "Review on modeling and simulation of interdependent critical infrastructure systems," *Reliability Engineering & System Safety*, vol. 121, no. Supplement C, pp. 43 – 60, 2014.

[2] D. H. Shin, D. Qian, and J. Zhang, "Cascading effects in interdependent networks," *IEEE Network*, vol. 28, pp. 82–87, July 2014.

[3] A. Berizzi, "The Italian 2003 blackout," in *IEEE Power Eng. Soc. General Meeting, 2004.*, pp. 1673–1679 Vol.2, June 2004.

[4] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, "Catastrophic cascade of failures in interdependent networks," *Nature*, vol. 464, pp. 1025–1028, Apr 2010.

[5] S. Tauch, W. Liu, and R. Pears, "Measuring cascade effects in interdependent networks by using effective graph resistance," in *2015 IEEE Conf. on Comput. Commun. (INFOCOM) WKSHPS*, pp. 683–688, April 2015.

[6] A. Sen, A. Mazumder, J. Banerjee, A. Das, and R. Compton, "Identification of k most vulnerable nodes in multi-layered network using a new model of interdependency," in *2014 IEEE Conf. on Comput. Commun. (INFOCOM) WKSHPS*, pp. 831–836, April 2014.

[7] Y. Zhao and C. Qiao, "Enhancing the robustness of interdependent cyber-physical systems by designing the interdependency relationship," in *2017 IEEE Int. Conf. on Commun. (ICC)*, pp. 1–6, May 2017.

[8] M. Rahnamay-Naeini, "Designing cascade-resilient interdependent networks by optimum allocation of interdependencies," in *2016 Int. Conf. on Comput., Netw. and Commun. (ICNC)*, pp. 1–7, Feb 2016.

[9] A. Sturaro, S. Silvestri, M. Conti, and S. K. Das, "Towards a realistic model for failure propagation in interdependent networks," in *2016 Int. Conf. on Comput., Netw. and Commun. (ICNC)*, pp. 1–7, Feb 2016.

[10] M. Parandehgheibi and E. Modiano, "Robustness of interdependent networks: The case of communication networks and the power grid," in *2013 IEEE Global Commun. Conf. (GLOBECOM)*, pp. 2164–2169, Dec 2013.

[11] M. Yannakakis, "Node-deletion problems on bipartite graphs," *SIAM Journal on Comput.*, vol. 10, no. 2, pp. 310–327, 1981.

[12] D. B. Johnson, "Finding all the elementary circuits of a directed graph," *SIAM Journal on Comput.*, vol. 4, no. 1, pp. 77–84, 1975.

[13] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[14] Y. Zhao and C. Qiao, "Enhancing the robustness of interdependent cyber-physical systems by designing the interdependency relationship," in *2017 IEEE Int. Conf. on Commun. (ICC)*, pp. 1–6, May 2017.