An Analysis of Cyclical Interdependencies in Critical Infrastructures

Nils Kalstad Svendsen[1] and Stephen D. Wolthusen[1,2]

[1] Norwegian Information Security Laboratory, Gjøvik University College, P.O. Box 191, N-2802 Gjøvik, Norway
[2] Information Security Group, Department of Mathematics, Royal Holloway, University of London, Egham Hill, Egham TW20 0EX, UK

Abstract In this paper we discuss the properties and algorithmic methods for the identification and classification of cyclical interdependencies in critical infrastructures based on a multigraph model of infrastructure elements with a view to analyze the behavior of interconnected infrastructures under attack. The underlying graph model accommodates distinct types of infrastructures including unbuffered classes such as telecommunications and buffered structures such as oil and gas pipelines. For interdependency analyzes particularly between different infrastructure types, cycles multiple crossing infrastructure sector boundaries are still relatively poorly understood, and their dynamic properties and impact on the availability and survivability of the overall infrastructure is of considerable interest. We therefore propose a number of algorithms for characterizing such cyclical interdependencies and to identify key characteristics of the cycles such as the strength of the dependency or possible feedback loops and nested cycles which can be of particular interest in the development of mitigation mechanisms.

Keywords: Multigraph models, interdependency analysis, multiflow models

1 Introduction

One of the key characteristics of critical infrastructures is the level of interconnectedness and hence interdependency required for fully functional operation. At the level of individual infrastructure sectors (e.g. telecommunications, water supply, financial services, or the electric power grid) or at least for individual network operators, models exist which allow both monitoring and predictive analysis. While these models may explicitly or implicitly incorporate individual dependencies on other infrastructures, this is typically not done in a systematic fashion which would allow the identification and characterization of interdependency cycles spanning multiple infrastructure sectors and infrastructure operators. However, particularly when assessing the potential impact of targeted attacks and the robustness of infrastructure against such attacks, these are vital characteristics which are captured only inadequately by statistical reliability

models as the latter generally assume independent random variables with well-characterized probability density functions for modeling infrastructure component failures. While linear dependencies are straightforward to identify, cyclical interdependencies leading to feedback cycles are less obvious and require analytical or simulative tools for their identification and evaluation. The description and analysis of such dependency cycles is therefore of considerable interest for gaining an understanding of the robustness and particularly dynamic characteristics of critical infrastructures under attack at larger national and international levels. Based on domain-specific metrics of the strength of interdependency such an analysis building on a sufficiently detailed model based on directed multigraphs can — beyond what can be learned from the identification of strongly connected components as reported in earlier research [1–3] — identify cycles of dependencies of a certain strength as well as the most significant dependencies within such cycles. However, it is another characteristic that vertices and in some cases edges are shared between multiple cycles which can also intersect, with implications for nested feedback cycles when analyzing the dynamic effects of such interdependencies. Several properties of interdependency cycles are therefore of particular interest. These include algorithms for the identification of cycles as well as the discovery of topological structures and other static properties but also include dynamic properties such as the duration and other characteristics of feedback loops propagating through the individual and interconnected cycles where multiflow models offer an elegant formalism for answering some algorithmic questions which may be posed in this context. The remainder of this paper is therefore structured as follows: Section 2 briefly sketches the multigraph model underlying the work reported here, while section 3 discusses the properties of cyclical interdependencies incorporating multiple types of infrastructures. Section 4 then derives formal descriptions of such cycles using both graph statistics and flow formalisms. Both of these models are equilibrium-based and provide only limited insight into the processes leading to such equilibria, however. Section 6 briefly reviews selected related work before section 7 provides conclusions on our results and an outlook on ongoing and future work.

2 Multigraph Model

This section summarizes the essential parts of the multigraph model of critical infrastructure previously introduced by the authors [1–3]. In the model interactions among infrastructure components and infrastructure users are modeled in the form of directed multigraphs, which can be further augmented by response functions defining interactions between components. The vertices $\mathcal{V} = \{v_1, \ldots, v_k\}$ are interpreted as producers and consumers of m different types of services, named dependency types. Transfer of services takes place along the edges connecting the nodes in the network. Each edge can transport or transfer one dependency type d_j chosen from the set $\mathcal{D} = \{d_1, \ldots, d_m\}$.

In the general case it is assumed that all nodes v_a have a buffer of volume V_a^j (indicating a scalar resource; this may represent both physical and logical

resources and, moreover, may be subject to further constraints such as integral values) for each dependency type d_j. Assuming that the amount of dependency type d_j in node v_a can be quantized as N_a^j. For each node we can then define a capacity limit $N_{\mathrm{Max}}(v_a, d_j)$ in terms of the amount of resource d_j that can be stored in the node. The dependency types are classified as ephemeral ($V_a^j = 0$ for all nodes v_a, and it follows that $N_{\mathrm{Max}}(v_a, d_j) = 0$), storable and incompressible ($N_{\mathrm{Max}}(v_a, d_j) = \rho V_a$, where ρ is the density of the resource), or storable and compressible ($N_{\mathrm{Max}}(v_a, d_j) = P_{\mathrm{Max}}(v_a, d_j)V_a$, where $P_{\mathrm{Max}}(v_a, d_j)$ is the maximum pressure supported in the storage of resource d_j in the node v_a). Further refinements such as multiple storage stages (e.g. requiring staging of resources from long-term storage to operational status) and logistical aspects are not covered at the abstraction level of the model described here. Non-fungible resources must be modeled explicitly in the form of constraints on edges or dependency subtypes. Pairwise dependencies between nodes are represented with directed edges, where the head node is dependent on the tail node. The edges of a given infrastructure are defined by a subset $\mathcal{E}$ of $\boldsymbol{\mathcal{E}} = \{e_1^1, e_2^1, \ldots, e_{n_1}^1, e_1^2, , \ldots, e_{n_m}^m\}$, where $n_1, \ldots, n_m$ are the numbers of dependencies of type $d_1, \ldots, d_m$, and e_i^j is the edge number i of dependency type j in the network. A further precision of given dependency, or edge, between two nodes v_a and v_b is given by the less compact notation $e_i^j(v_a, v_b)$. In addition to the type, two predicates $C_{\mathrm{Max}}(e_i^j(v_a, v_b)) \in \mathbb{N}_0$ and $C_{\mathrm{Min}}(e_i^j(v_a, v_b)) \in \mathbb{N}_0$ are defined for each edge. These values represent the maximum capacity of the edge $e_i^j(v_a, v_b)$ and the lower threshold for flow through the edge. Hence, two $g \times m$ matrices, where $g = |\mathcal{E}|$ and m is the number of dependency types, C_{Max} and C_{Min} are sufficient to summarize this information.

Let $r_a^j(t)$ be the amount of a resource of dependency type j produced in node v_a at time t. $D(t)$ is defined to be a $k \times m$ matrix over $\mathbb{Z}$ describing the amount of resources of dependency type j available at the node v_a at time t. It follows that the initial state of D is given by $D_{aj}(0) = r_a^j(0)$, and for every edge in $\mathcal{E}$

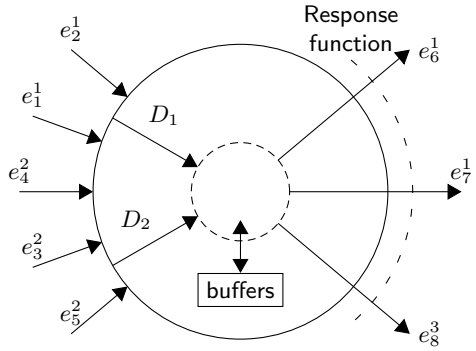

Figure 1. The parameters that define the functionality of a node, and its outputs

we can define a response function $R_i^j(v_a, v_b)$:

$$D_{aj} \times V_a^j \times N_a^j \times N_{\text{Max}}(v_a, j) \times C_{\text{Max}} \times C_{\text{Min}} \to \mathbb{N}_0 \tag{1}$$

that determines the i-th flow of type j between the nodes v_a and v_b (illustrated by fig. 1). The function $R_i^j(v_a, v_b)$ w.l.o.g. is defined as a linear function, and may contain some prioritizing scheme over i and v_b. By constraining the response function to a linear function and discrete values for both time steps and resources, linear programming approaches can be employed for optimization of the relevant parameters; interior point methods for this type of problem such as [4, 5] can achieve computational complexity on the order of $O(n^{3.5})$, making the analysis of large graphs feasible.

Given the responses at time t, the amount of resource j available in any node v_a at time $t + 1$ is given by

$$D_{aj}(t+1) = r_a^j(t) + N_a^j(t) + \sum_{i,s|e_i^j(v_s,v_a)\in\mathcal{E}} R_i^j(v_s, v_a, t). \tag{2}$$

A node v_a is said to be functional at time t if it receives or generates the resources needed to satisfy its internal needs, that is $D_{aj}(t) > 0$ for all dependency types j which are such that $e_i^j(v_b, v_a) \in \mathcal{E}$, where $b \in \{1, \ldots, a-1, a+1, \ldots k\}$. If this is the case for only some of the dependency types the node is said to be partially functional, and finally of no requirement are satisfied the node is said to be dysfunctional. For further argumentation on the motivation for the model, the granularity of the model, and example networks and scenarios we refer to [2]. For further modeling of networks carrying ephemeral and storeable resources, and the reliability of the network components we refer to [3].

3 Mixed Type Cycles

In [2] we demonstrate the effect of cascading failures through mixed types infrastructure networks. The design of network topologies is traditionally done with great care in critical infrastructures, following appropriate standards and regulations. With an appropriate approach in the design phase undesirable configurations within an infrastructure may be avoided. Our work focuses on approaches and analysis which can be performed such networks are interconnected to those involving other dependency types. This section therefore presents selected methods for detecting and classifying cycles across critical infrastructures.

3.1 Definition of Mixed Type Cycles

In general a cycle is a walk $W = v_{x_1} e_{x_1}^{d_1} v_{x_2} e_{x_2}^{d_2} \ldots e_{x_{n-1}}^{d_{n-1}} v_{x_n}$, through a subset of $\mathcal{V}$, which is such that v_{x_i} and v_{x_j} are pairwise distinct for $1 \leq i < j < n$, and $v_{x_1} = v_{x_n}$. If $d_i \neq d_j$ for some i and j we say that the cycle is a mixed type cycle, meaning that there are different dependency types linking the nodes together.

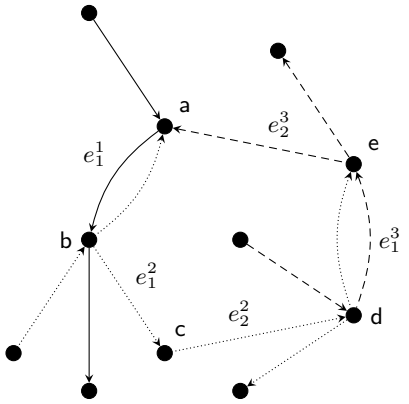

Figure 2. A mixed type cycle through three infrastructures

A simplified example of such a configuration can be seen in figure 2, where continuous, short, and long-dashed edges represents different dependency types. We easily see that the figure contains several mixed type cycles, among these the cycle $ae_1^1be_1^2ce_2^2de_1^3ee_2^3$. However, we cannot say anything about how these cycles interact without making further assumptions regarding the properties of the different dependency types and vertex behavior. Another issue worth exploring is the situation of node c. The owner of node c may think that the functionality of this node has only one external dependency, the one coming from node b. From the figure we see that the functionality of b is not at all straightforward. The following provides several mechanisms for exploring dependency properties, particularly the role of cyclic dependencies in the functionality of systems.

3.2 Detection of Mixed Type Cycles

Detection of mixed type cycles consists of two steps. First, all cycles of the typeless network are detected. The typeless network is the mapping of a network where each edge is associated with a dependency type to a network where an edge defines a dependency between the tail and the head node. Detection of cycles in networks is done by applying a classical depth first search on the network, where edges classified as back edges identifies the existence of a cycle [6]. The run time of this approach is $\Theta(|\mathcal{V}|+|\mathcal{E}|)$, where $|\mathcal{V}|$ and $|\mathcal{E}|$ are the cardinalities of the respective sets. The second part is to determine the consecutive dependency types of the cycle edges. If all edges of a cycle are of the same dependency type the cycle is not of mixed type, thus it is discarded. If there is at least two dependency types included in the cycle, it is a mixed type cycle.

4 Classification of Mixed Type Cycles

This section explores approaches to determine the importance of mixed type cycles. We propose two approaches to explore these features in large interconnected

infrastructures. One approach is based on the statistical properties of the nodes of a cycle and their surroundings, the other is based on flow considerations in the network, bottlenecks and min-max cuts

4.1 Statistical approach

A statistical approach to classify mixed type cycles is appropriate in two different situations. It is particularly useful for gaining a quick overview of a given configuration where only limited computational resources are available in relation to the size of the network to be investigated. The second situation is for large infrastructures where all the information from the model described in section 2 is not available. In this case the statistical model represents the achievable limits for analysis.

The traditional approach to network statistics (refer to e.g. [7] for an overview of network statistics) is to focus on either local or global statistics. For the purposes of the application area considered in this paper, mixed type cycles can be considered as a regional or mezzanine level of network statistics, describing the statistical properties of a subset of the nodes of a network, i.e. the nodes included in one or several cycles. A network statistic should describe essential properties of the network, differentiate between certain classes of networks and be useful in algorithms and applications [7].

Cycle length The length of a cycle is a very basic characteristic. Short cycle length might indicate that the cycle likely covers a small number of dependency types. In this way the cycle is more likely to be detected in naive approaches and the importance of the cycle might be more predictable. It is important to note that a cycle containing many nodes may well contain just a few dependency types. Thus one should also rank the cycles depending on the number of dependency types they contain as well as consider the feedback duration (minimum and average duration) based on the respective response functions cycles thus identified. We make the assumption that the number of cycles is bound by the number of nodes in the network. This assumption is realistic in most application scenarios, as too many cycles, especially in an infrastructure transporting physical commodities, are inefficient. The length of the cycle is bounded by the number of nodes in the network so finding the cycle length given the cycles is roughly of complexity $O(|\mathcal{V}|^2)$.

Average redundant in-degree of nodes in the cycle High average in-degree of the nodes in a cycle can, depending on constraints, indicate a high level of redundancy. One must obviously distinguish between the different dependency types in this analysis, referring to the typed dependency graph (e.g. considering the case that a node may go down if cooling dependencies are no longer satisfied even with multiple redundant electrical power feeds). The in degree of a node is bounded by the number of edges in the network, and is calculated once by going through the adjacency list or matrix (depending on the selected representation in of the implementation) of the graph. Given that the number of nodes in a cycle is $O(|\mathcal{V}|)$ the complexity of this statistic is $O(|\mathcal{V}||\mathcal{E}|)$.

Strength of interconnecting edge The strength of the interconnecting edge represents similar information to the average redundant in-degree of the nodes in the cycle of a given type, but focuses in particular on the nodes of the cycle where the input is of one dependency type and the output is of an other dependency type. This can e.g. be owing to such a shift often indicating a transition of infrastructure owner. The identification of such edges can be done by going through the adjacency list or matrix and check the dependency types required for the response function of the edge with the dependency types produced by the head node. In the presented model this can be done by a table lookup, and the complexity is $O(|\mathcal{V}||\mathcal{E}|)$. The in degree is found in the classical way.

Importance of interconnecting edge The relative importance of an interconnecting edge is determined by second-order dependencies on the edge. It is therefore required to perform a search of dependencies at a depth of one from the edge under consideration. The interconnecting edges are identified as for the strength of the interconnecting edges. For this metric the interest lies in the neighborhood of the head node of the interconnecting edge. A first indication is found by counting the number of response functions related to the head node that demands the dependency type provided by the interconnecting edge, again an approach based on table lookups and counting is suggested. Moreover, one can explore the size of the spanning tree of the interconnecting node, in order get an indication of the importance of the node in the dependency network.

Overlapping cycles Long cycles in multigraphs are likely to link the nodes of the network closer together. Further there is no reason why overlapping cycles may exist. This is likely to emphasize this effect and cause more chaotic chain reactions in case of failures. For a node or facility in a network it is not necessarily bad to receive parts of the resources from cycles, but when these cyclic infrastructures are interwoven and interdependent one should consider establishing alternative supplies. Thus algorithms to detect such configurations are important. Overlapping cycles may have one or several common points, or paths shared between cycles. Once the list of cycles is established, common points can be found by comparing the entries.

4.2 Flow-Based Modeling

In scenarios where more network information and time for model initiation is available, network flow models can provide insight beyond the information given by the statistical approach on the interconnected networks in question. Given that several dependency types flow through the networks, multicommodity flows or multiflows is a natural theoretical framework to map the presented model onto for further investigations (see e.g. [5]). Multiflows are traditionally used to model communication or transportation networks where several messages or goods must be transmitted, all at the same time over the same network. In general polyhedral and polynomial-time methods from classical (1-commodity) flows and paths, such as max-flow min-cut, do not extend to multiflows and

paths [5]. But, given particular properties of the networks in question, efficient solutions can be found in some cases. In this section we show how our model can be adapted to the multiflow framework, and explore what opportunities this gives for further studies.

Definition of the Adapted Multiflow Problem Given two directed graphs, a supply digraph $D = (V, A)$ and a demand digraph $H = (T, R)$, where V is a finite set (vertices), $T \subseteq V$, and A and R are families of ordered pairs respectively from V and T (edges), Schrijver [5] defines a multiflow as a function f on R where f_r is an $s{-}t$ flow in D for each $r = (s, t) \in R$. In the multiflow context, each pair in R is called a net, and each vertex covered by R is called a terminal.

The model presented in section 2 does not explicitly classify sources and sinks, but these can be deduced from the properties of the edges at any given time. Sources are nodes where $D_j(t) = 0$ and $r^j(t) > 0$, while sinks are nodes where $D_j(t) > 0$ and $r^j(t) = 0$. Further, if each of the m dependency types in our model is to represent one commodity flow in the multiflow network, this results in m super-sources s_j, each linked to every source of dependency type d_j and m super-sinks t_j connected to every sink of dependency type d_j. Given this modification, we now have that $|R| = m$, where m is the number of dependency types, and the flow network is called an m-commodity flow, and our dependency types can also be named commodities. The value of f is the function $\phi : R \to \mathbb{R}_+$ where ϕ_r is the value of f_r. For each edge we have previously defined a max flow, or maximum capacity function, $c_{Max} : A \to \mathbb{N}$, where $C_{Max}(e_i^j(v_a, v_b))$ is the value of c. We say that a multiflow f is subject to c if

$$\sum_{r \in R} f_r(e_i^j(v_a, v_b)) \leq c(e_i^j(v_a, v_b))$$

for each edge $e_i^j(v_a, v_b)$. The multiflow problem over our model is then given a supply digraph $D = (V, A)$, a demand digraph $H = (T, R)$, a capacity function c_{Max}, and a demand function $d = R \to \mathbb{R}_+$ at time t to find a multiflow subject to d, what is called a feasible multiflow. Related to this problem is the maximum-value multiflow problem, where the aim is to maximize d.

The two models are now equivalent up to the point of time dependency. Our model allows most of the features to vary over time, while as the multiflow framework assumes that edge capacity and node behavior is static. An important question in the following section is therefore how, or rather if, the behavior of a dynamic model (as well as the system being modeled) converges towards the idealized properties of a static model.

Applicable Properties and Algorithms on Multiflows and Related Problems The motivation for connecting our model to the multiflow model and its related problems and algorithms is to identify algorithms of polynomial time complexity that can be applied for network analysis. These cases are not numerous, but the few that exists are interesting for the scenarios the presented model is faced with.

If each flow f_r is required to be integral as stipulated in section 2 or rational, the multiflow problem described in the previous section is called respectively the integer and fractional multiflow problem. The fractional multiflow problem can easily be described as one of solving a system of linear inequalities in the variables $f_i(e_i^j(v_a, v_b)$ for $i = 1, \ldots k$ for all edges in $\mathcal{E}$, and a static solution to the multiflow problem can be found in polynomial time with any polynomial-time linear programming algorithm [5].

The disjoint paths problems is another class of problems that has an immediately intuitive application to the model discussed in this paper. Assuming all capacities and demands set to a value of 1, the integer multiflow problem is equal to the (k) arc- or edge-disjoint problem, i.e. given a directed graph $D = (v, A)$ and pairs $(s_1, t_1), \ldots, (s_k, t_k)$ of vertices of G, to find arc- (or edge-) disjoint paths $P_1, \ldots, P_k$, where P_i is a $s_i - t_i$ path $(i = 1, \ldots, k)$. In the terminology of critical infrastructure models, this is to find redundant paths or connections for the different flows. Similarly one can define the vertex disjoint problem [5]. The complexity of the vertex k disjoint path problem over planar directed graphs is polynomial, while it is unknown for the arc-disjoint path problem. This provides an efficient mechanism for checking whether a flow believed to be redundant is indeed redundant.

5 Analytical Approach

Based on the model introduced in section 2 and the statistics and algorithms presented in section 3, this section outlines algorithms to analyze the influence of mixed types cycles on a pre-defined subgraph from the perspective of an infrastructure or sub-network owner. Let $N = (\mathcal{V}', \mathcal{E}')$ be a subgraph of the multigraph $G = (\mathcal{V}, \mathcal{E})$. We assume that $|\mathcal{V}| \geq 1$, and that $|\mathcal{E}| \geq 0$, meaning that N can be a single node, a number of independent nodes, or a connected subgraph. For an infrastructure owner there are two scenarios including cyclical interdependencies that are of interest; cycles within the controlled network and cycles which are partially under control and partially traversing infrastructure controlled by other operators. Our focus is on the latter, and in the following we outline an approach for operators to detect critical configurations given that all operators of the network are willing to share network information.

5.1 Detection of intersecting cycles

For every node in G the approach for detection of cycles described in section 3.2 is used to detect mixed cycles. The cycles can be classified into three groups: Mixed cycles included in N, partially included in N, and those not included in N. This can be done using a string string matching algorithm such as the Knuth-Morris-Pratt algorithm of complexity $O(m + n)$, where m and n is the length of the strings to be matched [6]. Assuming that the longest cycle has $|\mathcal{V}| + |\mathcal{E}|$ elements and that $|C|$ is the number of cycles the detection has complexity $\mathcal{O}(|C|^2(|\mathcal{V}| + |\mathcal{E}|))$. We see that the complexity of the algorithm is highly dependent on the

number of cycles in the graph, and as this is an infrastructure dependent property giving this estimate in terms of e.g. $\mathcal{E}$ and $\mathcal{V}$ is unlikely to result in appropriate bounds.

5.2 Determination of cycle criticality

The characteristic of dependency cycle is neutral. In previous sections we have listed some properties of the overall stability of a cycle, e.g. the average redundant in-degree of the nodes of a cycle. As mentioned, a mixed type cycle may well cross subgraphs of the network of multiple ownership. Each of these owners will typically be more interested in how dependent the subnetwork is on the functionality of the cycle, and in some cases also how dependent the cycle is on the sub-network. Here we sketch an algorithm for a automatic, or semi-automatic, classification of the influence of a cycle which is partially included in the subnetwork N on N itself. We define an entry point, v_{in}, of a cycle to be the first vertex located inside our network N, that is $v_{x_i} \in N$ such that $v_{x_{i-1}} \notin N$, an its corresponding type be the dependency type binding the two nodes together. Further we define the corresponding exit point, v_{out}, to be the first vertex of the cyclic path located outside N, that is $v_{x_{i-1}} \in N$ and $v_{x_i} \notin N$ with a corresponding type defined as for the entry point. Further we let C be a table which for each cycle contains four-tuples of the form $(v_{\text{in}}, d_{\text{in}}, v_{\text{out}}, d_{\text{out}})$, enabling cycles to traverse N more than once. This definition of v_{in} and v_{out} is compatible with vertex and and edge coalescion, not based on connectivity properties as described in [8] but on ownership, which can be used for high-level network analysis. Algorithm 1 suggests an approach to derive some descriptive statistics of cycles that nodes of N are included in, and highlight important interdependencies. The algorithm applies the functions $S_{d_j}(v)$ (strength of incoming edge of dependency type d_j), $I(v)$ (importance of the functionality of vertex v), and $A(C)$ (average redundant in-degree of nodes in the cycle) and the complexity for each analysed cycle is $\mathcal{O}((|\mathcal{V}||\mathcal{E}|)^4)$. Based on this the infrastructure owner can use e.g. a breadth first search on the coalesced graph to identify alternative supplies of d_j to v_{in}.

6 Related Work

The need for models of critical infrastructures usable for both planning and operational purposes has led to a number of approaches; some of the more general

Algorithm 1 Detection of vulnerable nodes in cyclic interdependencies

```
G, N ∈ G, C, A
for all (v_in, d_in, v_out, d_out) ∈ C do
  if I(v_in) high, S_{d_in}(v_in) low, A(C) low then
    return v_in is in an vulnerable cycle. Consider redundant sources of dependency
    type d_in.
  end if
end for
```

approaches are reviewed by the present authors in [2] while an additional review of recent research in the CI(I)P area in Europe can be found in [9].

Graphs models represent a natural approach for dependency analysis and attack mechanisms and have been used at a number of different scales from individual attack models based on restricted graph classes [10] and abstract static hypergraphs [11] to work on graph properties [12–14]. The work reported in this paper attempts to bridge a gap in both the research taxonomy as proposed by Bologna *et al.* and also in the graph modeling in particular by investigating intermediate or regional-scale networks which, through careful conditioning of the model excerpt, still allows quantitative modeling and simulation; details of which underlying models can be found in [1–3].

7 Conclusions

Based on a multigraph model incorporating extensions to characterize the properties of selected types of infrastructures such as the electric power grid and oil and gas pipielines representing storable and non-storable as well as fungible and non-fungible resources, we have reported on mechanisms for characterizing cyclical interdependencies of multiple infrastructure types and the effects that such dependencies can have on the overall robustness of an infrastructure network. Our previous research has identified a number of configurations and scenarios in which feedback cycles can arise that are not always trivial or obvious to predict and may incorporate significant delays before taking effect [1–3]. By using both graph statistics and multiflow algorithms to characterize said cycles, it is possible to gain a more comprehensive understanding of the feedback cycles inherent in such configurations. However, it must be noted that the vast majority of research questions arising from said configurations are $\mathcal{NP}$-hard and can therefore often only be investigated using heuristic techniques or by limiting the subject of investigation to graphs of limited diameter and complexity.

Ongoing and future research will focus further on characterizing the risks and threats to the infrastructure network at both local and regional scales emanating from targeted attacks including the effects and attack efficacy which can be obtained by attackers from multiple coordinated events. While previous research has indicated that such attacks can be quite successful, particularly in networks with scale-free properties [2,3,15,16], there has been only limited research on the nexus between geospatial proximity and fine-grained time-based effects on interconnections and interdependencies of multiple infrastructure types [8]; this area is the subject of ongoing investigation by the present authors. In addition to the analytical and algorithmic approaches, we are also continuing to use simulations based on the model reported in this and earlier research both to validate the model itself and to show the usefulness of the results of applying the methodology. Given the large parameter space required even in well-characterized infrastructure networks, this is likely to permit the identification and exploration of further properties of the interdependence model.

References

1. Svendsen, N.K., Wolthusen, S.D.: Multigraph Dependency Models for Heterogeneous Infrastructures. In: First Annual IFIP Working Group 11.10 International Conference on Critical Infrastructure Protection, Hanover, NH, USA, IFIP, Springer-Verlag (March 2007) 117–130
2. Svendsen, N.K., Wolthusen, S.D.: Connectivity models of interdependency in mixed-type critical infrastructure networks. Information Security Technical Report **12**(1) (March 2007) 44–55
3. Svendsen, N.K., Wolthusen, S.D.: Analysis and Statistical Properties of Critical Infrastructure Interdependency Multiflow Models. In: Proceedings from the Seventh Annual IEEE SMC Information Assurance Workshop, United States Military Academy, West Point, NY, USA, IEEE Press (June 2007) 247–254
4. Karmarkar, N.: A New Polynomial Time Algorithm for Linear Programming. Combinatorica **4**(4) (1984) 373–395
5. Schrijver, A.: Combinatorial Optimization. Springer-Verlag, Berlin, Germany (2003) Three volumes.
6. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorihms. First edn. The MIT Electrical Engineering and Computer Science Series. The MIT Press (1990)
7. Bandes, U., Erlebach, T., eds.: Network Analysis, Methodological Foundations. First edn. Volume 3418 of LNCS. Springer-Verlag, Berlin Heidelberg (2005)
8. Wolthusen, S.: Modeling Critical Infrastructure Requirements. In: Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop, United States Military Academy, West Point, NY, USA, IEEE Press (June 2004) 258–265
9. Bologna, S., Di Costanzo, G., Luiijf, E., Setola, R.: An Overview of R&D Activities in Europe on Critical Information Infrastructure Protection (CIIP). In Lopez, J., ed.: Proceedings of the First International Workshop on Critical Information Infrastructures Security (CRITIS 2006). Volume 4347 of Lecture Notes in Computer Science., Heidelberg, Germany, Springer-Verlag (August 2006) 91–102
10. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In Won, D., Kim, S., eds.: Proceedings of Information Security and Cryptology (ICISC 2005). Volume 3935 of Lecture Notes in Computer Science., Heidelberg, Germany, Springer-Verlag (December 2005) 186–198
11. Baiardi, F., Suin, S., Telmon, C., Pioli, M.: Assessing the Risk of an Information Infrastructure Through Security Dependencies. In Lopez, J., ed.: Proceedings of the First International Workshop on Critical Information Infrastructures Security (CRITIS 2006). Volume 4347 of Lecture Notes in Computer Science., Heidelberg, Germany, Springer-Verlag (August 2006) 42–54
12. Callaway, D.S., Newman, M.E.J., Strogatz, S.H., Watts, D.J.: Network Robustness and Fragility: Percolation on Random Graphs. Physical Review Letters **85**(25) (2000) 5468–5471
13. Cohen, R., Erez, K., ben-Avraham, D., Havlin, S.: Resilience of the Internet to Random Breakdowns. Physical Review Letters **85**(21) (2000) 4626–4628
14. Cohen, R., Erez, K., ben-Avraham, D., Havlin, S.: Breakdown of the Internet under Intentional Attack. Physical Review Letters **86**(16) (2001) 3682–3685
15. Dorogovtsev, S.N., Mendes, J.F.F.: Effect of the accelerating growth of communications networks on their structure. Physical Review E **63** (2001) 025101
16. Casselman, W.: Networks. Notices of the American Mathematical Society **51**(4) (2004) 392–393