RNEDE: Resilient Network Design Environment

Venkat Venkatasubramanian[a], Tanu Malik[b], Arun Giridhar[a],
Kris Villez[a], Raghvendra Prasad[c], and Aviral Shukla[a]
[a]School of Chemical Engineering, [b]Cyber Center
and [c]Computer Science
Purdue University
West Lafayette, IN 47907
Contact Author: venkat@ecn.purdue.edu

Craig Rieger,
Keith Daum and Miles McQueen
Instrumentation, Control
and Intelligent Systems
Idaho National Labs
Idaho Falls, ID 83415
Email: craig.rieger@inl.gov

***Abstract*—Modern living is more and more dependent on the intricate web of critical infrastructure systems. The failure or damage of such systems can cause huge disruptions. Traditional design of this web of critical infrastructure systems was based on the principles of functionality and reliability. However, it is increasingly being realized that such design objectives are not sufficient. Threats, disruptions and faults often compromise the network, taking away the benefits of an efficient and reliable design. Thus, traditional network design parameters must be combined with self-healing mechanisms to obtain a resilient design of the network.**

In this paper, we present RNEDE**a resilient network design environment that that not only optimizes the network for performance but tolerates fluctuations in its structure that result from external threats and disruptions. The environment evaluates a set of remedial actions to bring a compromised network to an optimal level of functionality. The environment includes a visualizer that enables the network administrator to be aware of the current state of the network and the suggested remedial actions at all times.**

I. Introduction

The nation's critical infrastructure is increasingly characterized by large networks. These include the electrical power grids, road systems, airline systems, chemical production systems, and communication networks such as the Internet. A critical property of such networks is their topological structure [1]. The topology governs several crucial properties of the network such as the efficiency of the network and aggregation of the state of the individual components of the network.

To determine a topology, most applications seek a design that achieves some functional criteria often expressed as a maximization and/or minimization of a given objective function. Thus a road network system will aim to find a topological structure that minimizes the average transit time and a communication network will seek for a topology that minimizes latency within a given infrastructure. However, such an optimized network design has limited applicability in the real-world where the network will be subject to a variety of disruptions, faults and threats. In such a scenario, the design of an optimal topology for a network must be accompanied by remedial actions that make it resilient to a variety of threats and disruptions. Such remedial actions must be provided to system administrators monitoring the state of the network, necessitating the need for a resilient network design environment.

An environment for resilient design of a network may either suggest remedial actions to bring the compromised topology to an optimal level of functionality or may suggest that the current topological structure remains compromised. The environment helps the network administrator to answer critical questions, such as:

- *Which remedial action must be taken?*
- *Where in the current network remedial actions should be taken?*
- *Whether remedial actions are even worth taking?*

The design of such an environment can, however, be a challenging task. Disruptions in the network, more often than not are unpredictable in nature, *i.e.,* it is difficult to ascertain what kind of disruption may arise and thus which component will fail next. Even if the disruption is known, the space of remedial actions is often very large and the system needs to choose the most cost efficient remedial action. Further, it is important that the remedial action leads to a network structure that is optimal in that it maximizes/minimizes the objective function. It may be equally possible that no remedial action is taken.

This paper presents RNEDE our on-going project at Purdue University to create an environment for resilient network design. The environment acts as a guide for the network administrator to replay various threat scenarios and thus design and maintain resources for remedial measures in a proactive manner. The environment consists of an an optimizer and a decision controller, which when subject to a series of disruptions determines a topology that achieves the functionality of the network and determine cost-efficient remedial measures. The combined process of optimization and decision making is iterative in that a different network emerges as the system undergoes a sequence of threats. To facilitate ease of use, the framework is coupled with a visualizer that allows the network administrator to continuously visualize the topology as it undergoes iteration.

The remainder of the paper is organized as follows: In Section II, we present recent work on design of large complex networks. We demonstrate the use of RNEDE in a real-life application scenario in Section III. Section IV presents the overall RNEDE architecture and the objective that it achieves. In Section V we describe in detail various components of RNEDE. Finally, we conclude and describe future work in Section VI.

II. Related Work

Design and control theories of complex resilient networked systems is a relatively new discipline of inquiry. Indeed, network theory has so far largely been limited to the topological analysis of existing or emergent complex networks, such as discovering the power law degree distribution [2] and the small-world phenomena [3]. The inverse operation, laws and principles that govern the design and operation of complex networks has only been explored recently.

In [4] the authors search for the fundamental principles that govern self-organization in a complex network. They allow a network of nodes to evolve, over several generations, to form a "fit topology" that satisfies certain survival objectives. The "fitness" of a network topology is defined as a function of efficiency, robustness, cost and environmental constraints. By varying these parameters, they get a set of topologies that are immediately optimal in different scenarios. Recently, scientists discovered a biologically-inspired model for self-organization of complex networks by borrowing simple properties from a slime molds behavior on a uniform mesh of oat flakes [5]. However, such approaches are limited to static self-organization. In communication networks, dynamic organization of the network under changing communication patterns has also received significant attention [6]. In this paper we explore an orthogonal aspect of dynamic self-organization, *i.e.,* understand how a self-organization of the topological structure in a network can aid in overcoming various threats and disruptions.

To enable the generation and evaluation of design and control algorithms for network design, benchmark tools are required. In particular, for designing resilient complex networks, a dynamic simulator of networked systems is desired. Such a simulator will encompass the diverse nature of complex networks, allow for a choice of generic optimization methods, and also provide a framework for resilient decision making. Popular existing software packages such as GloMoSim [7], ns [8] , NetSim [9], OMNeT++ [10], OPNET [11] and QualNet [12] do not provide capabilities for resilient design of complex networks. Further, we found that these simulators to be rather inflexible to achieve all the above desired features.

III. Motivating Application

Consider a large chemical company that maintains a sensor network over its entire facility. The network allows the company to monitor the state of the various variables of its plant, such as temperature and pressure of incoming and outgoing streams. The company has invested in an efficient design of the sensor network that allows it to collect the state of the plant variables quickly and to the nearest approximation. The organization achieves its objective of an efficient network by minimizing the overall diameter of the sensor network.

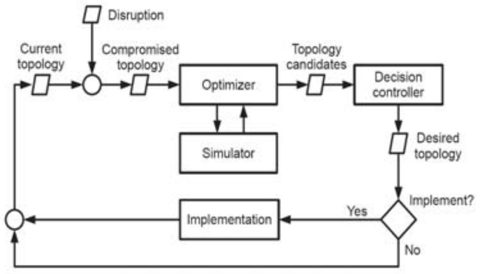

Fig. 1. RNEDE Architecture

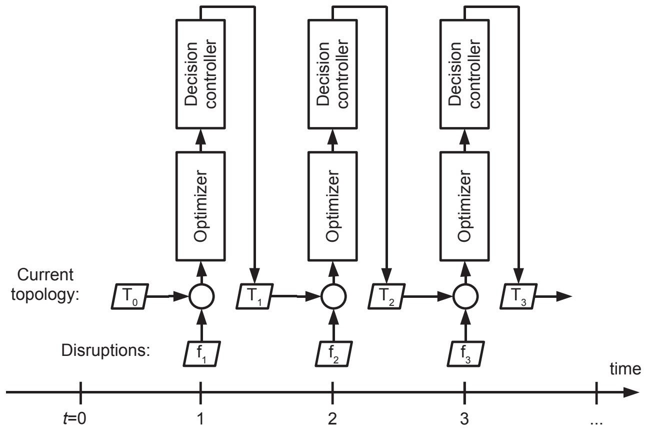

Fig. 2. RNEDE in action

However, achieving an efficient topology is constrained by the battery limitation of the sensors in that no sensor can communicate with more than k other sensors.

Unfortunately due to unpredictable environments, the sensors fail at any given instant of time thus affecting efficient information gathering on the network. The failure can be due to a software bug in the sensor leading to failure of the entire sensor or due to link loss as a result of network partitions. Sensor failures deteriorate the ability of the administrators to assess the health status of the plant. The company wants to design a resilient network in which remedial actions can be quickly identified and incorporated.

IV. RNEDE: Resilient Network Design Environment

Given a topology for a complex network and a sequence of incoming disruptions (resulting in a fault), the RNEDE determines which remedial actions to take such that the network remains resilient to structural changes in the topology (due to a given disruption). In RNEDE we assume that detection of a fault, as a result of the disruption, is both automatic and perfect. Thus whenever a node or an edge goes down we are immediately informed of its state. We also assume that diagnosis, *i*.e., determining the root cause of the structural losses does not alter the choice of a remedial action.

In RNEDE a remedial action corresponds to determining the location and number of new nodes and/or edges that can be placed back on the network to make it resilient to the disruption. The problem of determining the number and location of the remedial action is solved in two parts described as follows:

- In the first part, RNEDE determines the set of optimal remedial action that can be taken, *i.e.,* how many nodes/edges to add and where to add them. This determination is based on several factors such as the level to which the complex network has been compromised, and the constraints associated in implementing the remedial actions on the network.
- In the second part, RNEDE decides whether it should choose to implement any of the suggested remedial actions. This decision-making is crucial as a remedial action taken currently may not be valid in the future. This part of RNEDE is based on the sequence of incoming disruptions and the cost associated with the remedial actions.

We now describe the problem mathematically. Let the complex network be represented by a topology $T = (V, E)$, with V being the set of verticies and E being the set of edges. The topology specification T satisfies a set of constraints $C = c_1, \ldots, c_n$. The function $S : T \rightarrow \mathcal{R}^+$ determines the cost of maintaining a topology satisfying a set of constraints.

Let f_i be a disruption that arises in the network which "compromises" the topology of the network. Let T' be the compromised topology. T' may or may not satisfy the set of constraints C. Let $\mathcal{F} : T \rightarrow \mathcal{R}^+$ be a monotonic function that given the original topology and the compromised topology, quantitatively measures the amount of the compromise. Let the system's knowledge base consist of a set of remedial actions, $\mathcal{A} = a_1, \ldots, a_N$ and a cost function $Q : \mathcal{A} \rightarrow \mathcal{R}^+$. The two objectives in RNEDE then are to:

1) Obtain that set of remedial actions such that when applied to T' result in a remedied topology T'' such that the compromise is minimized, i.e., $F(T'', T) \leq \epsilon$, and
2) If $\sigma = f_1, \ldots, f_n$ is the sequence of disruptions that arise in the network and the knowledge about each disruption arrives sequentially at the system, then minimize the cost of maintaining the compromised topology and the cost of making the change, *i.e.,* $min \sum_{\sigma}(\sum_{a_i} Q(a_i) + \alpha S(T'))$. α is the weighing parameter that equates the two costs.

For our motivating application, the topology satisfies a degree constraint, *i.e.,*$|\{e | e \in E, (V, E) \in T\}| \leq k \ \forall V \in T$ for a given constant k. The function $\mathcal{F}$ is equal to the diameter of the topology. The cost function is application dependent and can be represented in any of the network quantities such as bandwidth or latency.

In RNEDE the first part of the problem is solved by modeling it as an optimization problem, and the second part of the problem is solved by modeling it as a decision-control problem. Figure 1 shows the architecture of RNEDE with the two vital components: the `Optimizer` and the `Decision-controller`. The `Optimizer` is aided with a `Simulator` to assess the level to which the complex network has been compromised. The framework is aided with a `Visualizer`, which allows the system administrator managing the network to diagrammatically see which changes in topology are being suggested. RNEDE evaluates a disruption one at a time and suggests a remedial action. This suggestion is based on the already seen threats and assumes minimal knowledge about the future sequence of disruptions that may arise. Figure 2 shows a conceptual time sequence as may arise through use of RNEDE. In the next section, we describe each component of RNEDE.

V. RNEDE Components

We now describe each individual component of RNEDE.

A. Optimizer and Simulator

The `Optimizer` minimizes the difference in the value of the objective function, *i.e.,* $\mathcal{F}$ on the original topology and the compromised topology. It uses a set of remedial actions to see which ones will minimize the difference. The objective function is deterministic as it only measures structural properties of the topology, such as connectedness, centrality, graph diameter. Thus the `Optimizer` tasks can be formulated as a mathematical program. It then guarantees that the remedied network obtained is indeed globally optimal, or within a known margin of the global optimum for many different performance measures. Note that if the objective function is a statistical performance measure, such as expected value of network lag, expected throughput of the network, 95-percentile cost of operating the network, mathematical programming cannot obtain an optimal or close to an optimal value. In this case, stochastic methods such as genetic algorithms, simulated annealing are better suited than mathematical programming.

The `Optimizer` is aided with a `Simulator` which when given a network specification, *i.e.,* its structure and parameters describing the weights on the nodes and edges, calculates the value of the objective function, such as diameter in our example. Such performance measures may be computed deterministically from the network parameters, or could be the result of a simulation, such as the expected value of a variable obtained from Monte Carlo simulation.

To demonstrate the working of the `Optimizer` we present the mathematical program that describes the scenario:

$$\min dia(G') \tag{1}$$

subject to

$$G''_{i,j} \geq G'_{i,j} \tag{2}$$

$$\sum_j G''_{i,j} \leq k \ \forall i \tag{3}$$

$$G'_{i,j}, G''_{i,j} \in \{0, 1\} \tag{4}$$

When present with this program, the `Optimizer` finds where should we add edges in the network to offset the loss of a node, such that the new network is still minimum diameter and satisfies all the original constraints, such as degree limits. The new network G' with added edges is subject to the same constraints as the original network. A further constraint specifies no deletion of edges that are already present.

B. Decision-Controller

Given the compromised topology T' and a remedied topology T'' (in general there may be several alternative topologies), the `Decision-controller` dtermines if it is beneficial to transition to T'' or remain in T'. The decision is based on the incoming sequence of disruptions, the cost associated with remaining in the current topology and the cost of transitioning between T and T''. The inherent model of the `Decision-Controller` can be summarized as that of a rent-vs-buy model: staying in the current topology, *i.e.,* T' corresponds to renting and moving to another topology *i.e.,* T'' corresponds to buying. In [6], we have presented several algorithms that can be developed on this model. These include both greedy heuristic and online-learning algorithms. The `Decision-Controller` can choose any of the algorithms based on the application and the knowledge about the failure model.

C. Visualizer

Graph visualization represents an important computational tool in analysis of network topologies. Topologies representing critical infrastructure applications belong the class of complex networks that are sparsely connected, large, strongly inhomogeneous, and their structure has constraints of small-worldness [3] and scale-free organization [2]. For such topologies, random graph placement is not efficient. Graph visualization algorithms based on physical models have shown to perform efficiently on such topologies [13]. RNEDE's `Visualizer` incorporates one such physics-based algorithm popularly known as the spring algorithm [14]. In this algorithm each node of the topology is considered a similarly charged particle and each edge as a spring. As same charges repel, the system tries to place the nodes as far away as possible from each other. Since some of the nodes are connected to each other they cannot be moved beyond a distance as the edges connecting them have an elastic limit, which is proportional to the edge weights.

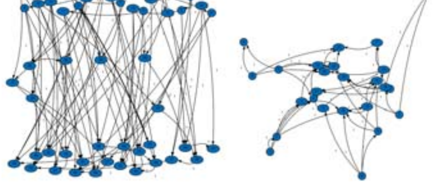

Fig. 3. Random and Spring Visualization

Figure V-C shows how the spring algorithm tries to move away overlapping edges in comparison to a random placement (left picture). Overall the algorithm tries to show the inherent dependencies among the nodes based on the edges and the corresponding edge weights. The computational complexity of the algorithm is $O(V(V+E))$

D. Implementation

RNEDE is currently being implemented in Python. It is integrated with a mathematical programming solver, such as GAMS [15]. The system measures a host of structural properties of the complex network, such as diameter, robustness, centrality measures, clustering coefficient. Any property or a combination of them can be chosen as the objective function of the `Optimizer`. In addition, constraints on the network can be specified in the program. It incorporates an event-based model in which the system listens for any incoming disruption. The tool is extensible in that new structural properties can be added. RNEDE is being developed in a modular fashion such that it can be integrated with any other mathematical program solver.

VI. Conclusion and Future Work

The RNEDE software allows to evaluate a variety of network designs and remedial actions in view of achieving resilience in networked systems. So far the environment includes a centralized and deterministic solutions to network monitoring and control. However, often for several critical infrastructures such as rail networks and road networks, centralized and deterministic approaches are inviable in practice. We plan to extend RNEDE such that it can incorporate fast approximation methods for network awareness and include distributed algorithms for network awareness and control. We also plan to extend it to a wide range of applications such as plant-wide control systems for chemical processing, product supply networks, and the electric power grid.

References

[1] V. Venkatasubramanian, "A theory of design of complex teleological systems: Unifying the darwinian and boltzmannian perspectives," *Complexity*, vol. 12(3), pp. 14–21, 2007.

[2] A.-L. Barabsi and R. Rka Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509 – 512, 1999.

[3] J. Travers and S. Milgram, "An experimental study of the small world problem," *Sociometry*, vol. 32, no. 4, pp. 425–443, 1969.

[4] V. Venkatasubramanian, S. Katare, P. Patkar, and F.-P. Mu, "Spontaneous emergence of complex optimal networks through evolutionary adaptation," *Computers and Chemical Engineering*, vol. 28, pp. 1789–1798, 2004.

[5] A. Tero, S. Takagi, T. Saigusa, K. Ito, D. P. Bebber, M. D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki, "Rules for biologically inspired adaptive network design," *Science*, vol. 327, pp. 439 – 442, 2010.

[6] T. Malik, R. Prasad, S. Patil, A. Chaudhary, and V. V., "Providing scalable data services in ubiquitous networks," 2010.

[7] "The GlomoSim," http://pcl.cs.ucla.edu/projects/glomosim/.

[8] "The Network Simulator (NS)," http://nsnam.isi.edu/nsnam.

[9] "The NetSim," ttp://www.tetcos.com/software.html.

[10] "The Omnet," http://www.omnetpp.org/.

[11] "The Opnet," http://www.opnet.com/.

[12] "The QualNet," http://www.scalable-networks.com/products/qualnet/.

[13] M. Šuvakov, "Physics Based Algorithms for Sparse Graph Visualization," *Computational Science–ICCS 2008*, pp. 593–600, 2008.

[14] P. Eades, "A heuristic for graph drawing," *Congressus numerantium*, vol. 42, no. 149160, pp. 194–202, 1984.

[15] A. Brook, D. Kendrick, and A. Meeraus, "GAMS, a user's guide," *ACM SIGNUM Newsletter*, vol. 23, no. 3-4, p. 11, 1988.