Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Pattern Recognition of Socio-technical Network Vulnerabilities

Modeling and preliminary results

Nicolas Léchevin
Anne-Laure Jousselme
Patrick Maupin
DRDC Valcartier

Defence R&D Canada – Valcartier

Technical Report
DRDC Valcartier TR 2013-409
December 2013

Canada

Pattern Recognition of Socio-technical Network Vulnerabilities

Modeling and preliminary results

Nicolas Léchevin
Anne-Laure Jousselme
Patrick Maupin
DRDC Valcartier

Defence R&D Canada – Valcartier

Technical Report
DRDC Valcartier TR 2013-409
December 2013

IMPORTANT INFORMATIVE STATEMENTS

This work is part of WBEs 15af07.

Abstract

When faced with potentially disruptive events, the state of a network may unexpectedly evolve to regions of the state space where safe operating conditions are no longer ensured. It is thus highly desirable to relate the network characteristics and operating conditions to its vulnerabilities, if any, in order to mitigate risk expressed as a function of network inoperability and loss of quality of service. A pattern recognition approach is adopted to relate the structural features of the network to the loss of operating nodes and edges. Two types of networks are considered for analysis and simulation in this document. A network characterized by flow conservation and capacity constraints is adapted from a fuse model, which may lead, in some instances, to cascading events. A tactical swarm of robots is deployed either to achieve terrain surveillance coverage or to maintain client connectivity so that every client can communicate in remote area. In both cases, the swarm of robots should maintain its connectivity at each time instant. The swarm deployment adapts to the loss of a robot caused by such factors as hardware/software failure, enemy action, or the presence of malware. The motion strategy prioritizes the client coverage, which may entail possible losses of connectivity. Given the motion strategy at hand, the swarm presents vulnerabilities related to the loss of some nodes. The classifier, instrumental in performing pattern recognition, is trained from a sample of networks obtained by some probabilistic generator. The classifier is shown to model, and to some extent, predict quickly the vulnerabilities of a class of networks as a function of their structural properties.

Résumé

La présence d'événements perturbant le fonctionnement des réseaux peut entraîner, sous certaines conditions, des dysfonctionnements importants. Il est donc souhaitable de cerner les conditions de fonctionnement et les caractéristiques pertinentes du réseau afin de modéliser ses vulnérabilités et d'en atténuer les risques encourus exprimant, entre autres, les baisses du niveau de qualité de service du réseau. Les techniques de reconnaissance de forme sont appliquées avec comme hypothèse de travail, une corrélation importante entre caractéristiques structurelles et vulnérabilité du réseau. Deux types de réseaux présentant des phénomènes d'avalanche sont proposés pour tester l'approche. Un réseau respectant le principe de conservation de l'énergie et intégrant des contraintes sur la capacité maximale de transport des liens (inspiré d'un réseau de fusibles) permet de générer, dans certain cas, des phénomènes de cascade. Un réseau tactique de robots est déployé dans un environnement dynamique afin de répondre au problème posé par deux scénarios possibles : (i) la couverture d'un terrain pour y accomplir des tâches de surveillance, et (ii) le maintien de la connectivité d'un réseau de clients voulant communiquer à partir de régions éloignées. Le déploiement des robots est capable de s'adapter à d'éventuelles pertes, notamment causés par des bris, une attaque ennemie ou par la présence de programmes malveillants. La stratégie de déplacement priorise la couverture des clients et maintient, lorsque cela est possible, la connectivité du réseau. La perte d'un ou plusieurs robots peut constituer une vulnérabilité car le redéploiement du réseau peut engendrer de nouvelles pertes de liens et des bris de connectivité. Le classifieur est entraîné à partir d'un échantillon de réseaux obtenu à l'aide d'un générateur aléatoire de réseaux. Il est montré que ce classifieur représente assez bien le lien

entre les caractéristiques structurelles de la classe de réseaux étudiée et ses vulnérabilités potentielles, permettant ainsi une prédiction rapide des vulnérabilités d'une classe de réseaux.

Executive summary

Pattern Recognition of Socio-technical Network Vulnerabilities: Modeling and preliminary results

Nicolas Léchevin; Anne-Laure Jousselme; Patrick Maupin; DRDC Valcartier TR 2013-409; Defence R&D Canada – Valcartier; December 2013.

Introduction or background: Risk mitigation of large networks is carried out by means of digital, time-domain simulations that try to predict the outcome of a time-varying, event-driven system as a function of current operating condition, a partially unknown environmental context, a set of possible stressors including exogenous threats and policies. Simulations based on the application of first principles and engineering protocols may yield accurate results but are time consuming when dealing with large amounts of data, which are typical of complex networks. It is believed that a pattern recognition approach would speed up the vulnerability analysis of such networks. Two types of networks, which have the potential to experience cascading events in response to some triggering events, are considered to validate the proposed approach. One network is based on the fuse model where subsets of edges and nodes are sequentially removed whenever the corresponding subsets of capacity constraints are exceeded, following a change in the flow of energy in response to the initial loss of a node or an edge. The other network involves a tactical swarm of robots whose main objective is either to perform terrain surveillance coverage or to maintain communication connectivity among of a set of clients equipped with handheld devices and evolving in remote areas. These two objectives are achieved by implementing an adaptive motion strategy. The removal of robots, caused by enemy action, the unexpected presence of malware, or hardware/software failure, may also entail a sequential loss of robot connectivity.

Results: A set of triggering events is applied to the networks and results, after simulations, in a data set used to train a classifier. This classifier is built on the assumption that the structural properties of a network are central to its vulnerability analysis. Considering a class of networks obtained by applying a probabilistic generator, a classifier relating the network class structural features to vulnerability labels is then derived. The classifier can be simplified by extracting the most relevant features and improved by combining several classification approaches. As expected, the classifiers derived are able to provide fast results with a classification rate of up to 70%, meaning that up to this fraction of vulnerabilities can be identified for a prescribed class of networks.

Significance: Although classifier training may require a lot of computational power, especially for large networks, the training phase is achieved prior to online use of the classifier. Once a classifier is derived, obtaining the label vector of vulnerability and thus identifying parts of the network that are vulnerable is a very fast process. The classifier can be used for any networks of a same class defined by a prescribed set of parameters, that is, generated using the same generator. However, the robustness of the proposed classification methods with respect to parametric uncertainties of the network generator is not shown.

Future plans: Other classification techniques involving advanced combination functions, and the use of non-structural features such as dynamical-system-based features and signal-based features

should be exploited to improve the classification rate. Pattern recognition has been applied to synthetic data. The approach should be tested and evaluated with actual data and improved accordingly.

Sommaire

Pattern Recognition of Socio-technical Network Vulnerabilities: Modeling and preliminary results

Nicolas Léchevin; Anne-Laure Jousselme; Patrick Maupin ; DRDC Valcartier
TR 2013-409 ; R & D pour la défense Canada – Valcartier; décembre 2013.

Introduction ou contexte : La simulation numérique du comportement de grands réseaux est souvent utilisée pour analyser et atténuer le risque encouru par de telles infrastructures. Celles-ci répondent à des actions de commande par l'intermédiaire d'une dynamique le plus souvent instationnaire, événementielle, potentiellement perturbées par des menaces, et évoluant dans des environnements incertains. Les modèles, qui sont obtenus à partir de lois fondamentales et de protocoles d'ingénierie, fournissent des résultats assez précis mais au prix d'une charge calculatoire d'autant plus élevée que les réseaux à analyser sont de grande taille. La reconnaissance de forme est l'approche privilégiée dans ce projet en raison de la rapidité d'exécution d'un classifieur une fois entraîné pour une classe de réseaux donnée. Les modèles de deux types de réseaux sont proposés dans le but de tester une famille de classifieurs utilisés pour identifier les vulnérabilités potentielles de ces réseaux. Ces deux types de réseaux peuvent, sous certaines conditions, présenter des phénomènes d'avalanche. En effet, la désactivation d'un nœud ou d'un lien d'un réseau de transport peut engendrer une séquence de bris si les capacités de transport des liens sont violées suite au recalcul du point d'équilibre du réseau. Un réseau de robots assurant la surveillance d'un environnement dynamique ou assurant le maintien de la connectivité d'un réseau de clients voulant communiquer à partir d'une zone éloignée peut également subir une série de perte de connectivité de certains de ses robots, suite à la perte initiale d'un robot causée par l'apparition d'une panne, la présence d'un maliciel ou d'une action ennemie.

Résultats : Un ensemble de classifieurs est entraîné à partir de données obtenues des simulations de deux classes de réseaux suite à un événement perturbateur initial. Les relations entre caractéristiques structurelles d'une classe de réseaux générés à partir d'un mécanisme probabiliste et de leur vulnérabilité constituent l'hypothèse de base à partir de laquelle ces classifieurs sont engendrés. Le classifieur peut être simplifié en sélectionnant les caractéristiques structurelles les plus pertinentes. Une fois l'apprentissage réalisé, le classifieur peut fournir une réponse sur la vulnérabilité d'un réseau d'une classe définie par un ensemble de paramètres donnés avec un taux de classification pouvant atteindre 70%.

Importance : Bien que la phase d'apprentissage puisse s'avérer très exigeante en besoin calculatoire, l'identification des vulnérabilités d'un réseau par classifieur est très rapide. Un seul classifieur peut fonctionner pour tous les réseaux d'une même classe définie par un générateur. La robustesse des classifieurs par rapport aux variations paramétriques du générateur n'est cependant pas démontrée.

Perspectives : Des caractéristiques non structurelles, telles que celles liées aux propriétés dynamiques du réseau et aux signaux mesurables, ainsi que des classifieurs obtenus à partir de fonctions de combinaison plus sophistiquées que celles utilisées dans le présent travail, devraient

être exploitées afin d'améliorer l'identification des vulnérabilités. L'évaluation de l'approche proposée doit être validée avec des données réelles et améliorée en conséquence.

Table of contents

List of figures

List of tables

1 Introduction

1.1 Context

As suggested by the multidimensional description of civilian infrastructures[1] proposed by Rinaldi *et al.* (2001) and extending Perrow's taxonomy (Perrow, 1984), numerous factors may detrimentally affect the course of action of very large interconnected infrastructures, which often correspond to complex networks. This multidimensional description puts forward various factors that substantially render more complex the analysis of infrastructures involved in modern societies. This description involves (*i*) various states of operation such as maintenance operations and the state of service delivery (disrupted or stressed), (*ii*) organizational, social, operational, temporal and spatial scale factors, (*iii*) several types of interdependencies, namely physical, social (psychological), cybernetical, geographic or logical, (*iv*) several types of dependencies, which are related to such properties as resource storability (the resource can be accumulated at several places of the infrastructure), and resource compressibility, referring to the maximum pressure supported by the storage of resources (Svendsen and Wolthusen, 2007), (*v*) several types and degrees of coupling (tight or loose, coupling order referring to event causality) and response behaviour (linear interactions, complex interactions), and (*vi*) a variety of environments such as business and economic opportunities, public or private investment, legal and regulatory concerns, and safety and security issues. It should be noted that parallels between federal and Department of National Defense (DND) perspectives on the protection of infrastructures can be established (Bozek, 2002)[2].

Possible occurrence of malfunctions may impact linked infrastructures by virtue of interdependencies; that is dependencies possibly characterized by feedback loops or complex adaptive systems (Wildberger, 1998) with, to some extent, self-healing behaviour. Couplings may in turn give rise to major failures, which are classified as cascading failures, escalating failures and common cause failures (Rinaldi *et al.*, 2001). Since an infrastructure is a network of manmade systems (President's Commission, 1997), it is important to delineate conditions and patterns that have the potential to lead to a crisis.

Figure 1 displays such descending scenarios, which, although derived from lessons learned in the electric power industry (Heydt et al., 2001), may to some degree be applicable to other infrastructures. The steps of the scenario are classified in order of severity, along with aggravating factors (AF) that increase the likelihood of the system transitioning to the next step

[1] Following the report in (President's Commission, 1997) an infrastructure is "a network of manmade systems and processes that function collaboratively and synergistically to produce and distribute a continuous flow of goods and services." Following the US Critical Infrastructure Assurance Office (CIAO, 2003), infrastructures are defined as "the framework of interdependent networks and systems comprising identifiable industries, institutions (including people and procedures), and distribution capabilities that provide a reliable flow of products and services essential to the defense and economic security of the United States, the smooth functioning of governments at all levels, and society as a whole."

[2] The continuous flow of goods and services through the military supply chain from domestic bases and industrial units to foreign bases and operational theatres can be paralleled to some extent with global industries, although battle command involves infrastructures that are unique to DND (Léchevin and Maupin, 2011).

(Heydt et al., 2001; Léchevin and Maupin, 2009) from stressed state to a crisis state. As suggested in *Figure 1*, the network state may evolve toward unsafe regions (characterized by system instability, physical failure, disconnection, outage) of the state space as a result of the action of exogenous and endogenous disturbances, or of inappropriate system operation. In normal operating conditions, the system is in secure[3] state, meaning that the system operates safely[4] and that threats are detected and properly handled. A system in secure state may evolve to a system in stressed state when, for instance, its facilities tend to operate close to their respective limits[5]; therefore, reserve resources, if any, are minimal. Aggravating factors include lack of information and erroneous real-time assessment of the system state.

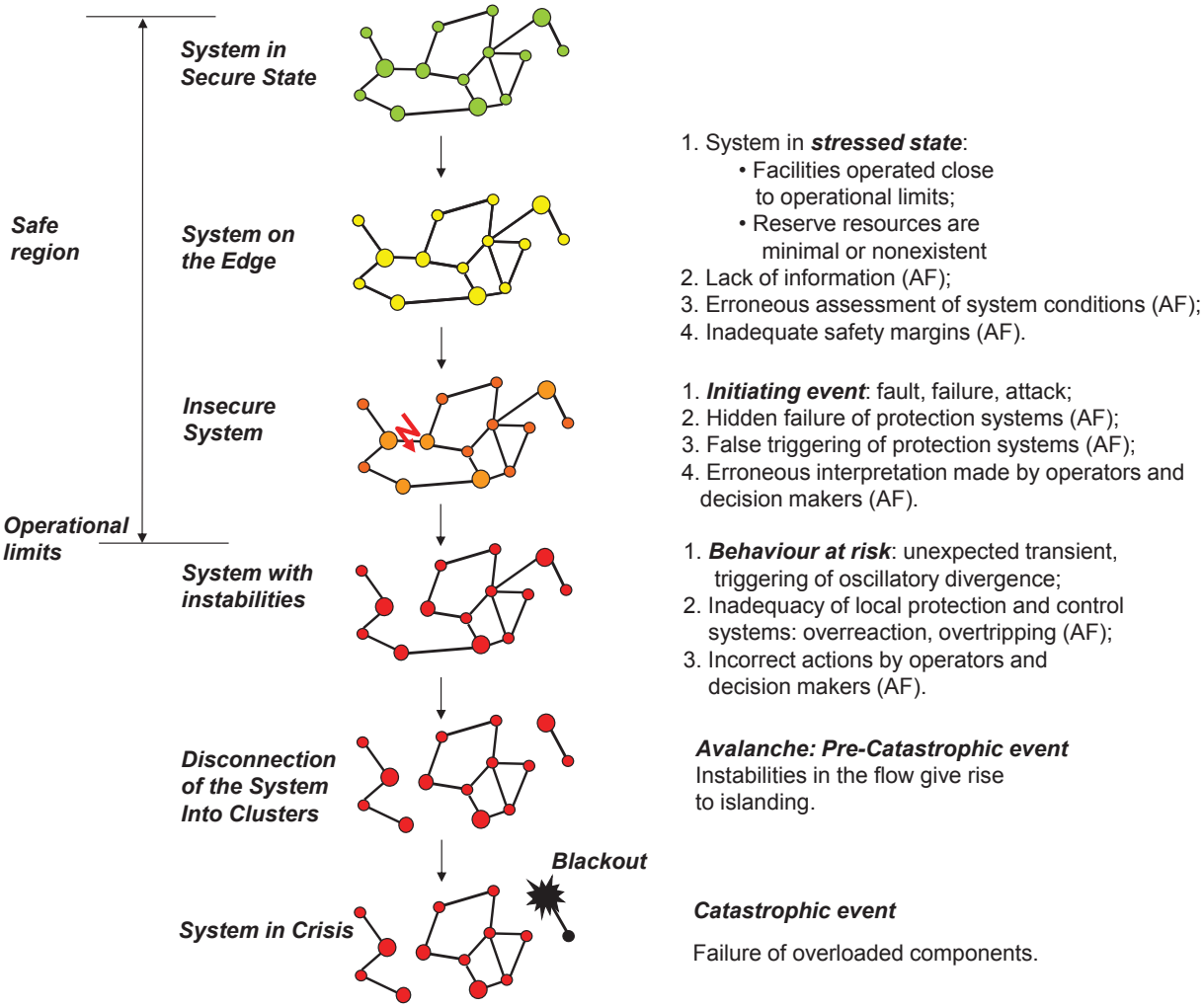

Figure 1: Mechanisms that lead to catastrophic events within infrastructures.

[3] Security refers to the prevention and detection of, and response to theft, sabotage of material, unauthorized access, illegal transfer or other source of malicious acts (IAEA, 2007).
[4] Safety refers to achieving proper operating conditions, preventing accidents and mitigating their impact on workers, the public and the environment (IAEA, 2007).
[5] Operational limits correspond to the parameter limits, the functional capability and the performance levels of equipment and personnel for safe operation of a facility (IAEA, 2007).

Indeed, vulnerabilities of a network are closely related to safety margins[6] defined with respect to operational limits. Exceeding operational limits may incur accident, thus possibly leading to the failure of a component or network and impacting workers, the public and the environment. The continued shrinking of these margins (system on the edge in *Figure 1*), owing to (*i*) economic and financial pressures, and to (*ii*) the lack of tools[7] for accurate, real-time evaluation of network state rather than relying on pre-calculated margins (Heydt et al., 2001), are some conditions that are the most conducive to the occurrence of behaviour at risk, and possibly to catastrophic events. And since various infrastructures tend to be less robust to disturbances in terms of safety margin, a triggering event (considered as an aggravating factor such as an attack, a fault, or a failure), although inoffensive in secure state, may push the infrastructure closer to an avalanche dynamic (behaviour at risk in *Figure 1*). Aggravating factors such as (*i*) the inadequacy of local protection and control systems in response to a global problem, false triggering and slow response times of those systems, and (*ii*) the inadequacy of human interpretations and decision making, lead to the tripping of additional facilities and finally to the catastrophic event shown in the lowest tier of *Figure 1*.

1.2 Prediction and Recognition of Vulnerabilities

To mitigate inoperability risks, a Prediction and REcognition of VUlnerabilities (PREVU) toolbox is being developed building on Pattern Recognition (PR) techniques (Léchevin *et al.*, 2011a). Building upon the objectives of PREVU and the Pattern Recognition (PR) formalization presented in Léchevin *et al.*, (2009, 2011a), we present in this report a classification-based vulnerability analysis focused on the exploitation of the structural features of networks. While other types of features[8] should also be exploited to analyze complex networks (Hines *et al.*, 2010), we aim at analyzing the effect of structural feature selection on the sensitivity of network vulnerability model to a class of triggering events corresponding to node and edge removal.

Two examples of network vulnerabilities are considered in this technical report. First, an avalanche mechanism corresponding to a variant of the fuse model (Hansen, 2005) is proposed. It is based on the energy conservation law across the network and on the fact that an edge is removed when the flow across it exceeds a prescribed value. While being a simplistic model when compared to an actual electric network, it is believed that this class of model can display dynamical features that are sufficiently informative to conduct and analyze pattern recognition experiments.

Second, vulnerabilities of a simplified model of a tactical, mobile, robot swarm deployed for surveillance coverage or for ensuring client connectivity are analyzed (Jousselme *et al.*, 2012a). For such a network, the individual robots are at risk of losing contact with the rest of the swarm owing to the local sparseness of the surveillance coverage or to clients' move, terrain complexity,

[6] The safety margin is defined as the difference between the operational limits (safe operating conditions) and the actual parameter value, functional capability or performance level of the system (infrastructure, personnel).

[7] Not being able to accurately estimate the network state may entail improper operation of the network, which in turn may steer the network state closer to the state-space boundary corresponding to its operational limit.

[8] For instance, it is shown in (Léchevin *et al.*, 2011b) how to derive dynamical system features for monitoring the stability of a class of cyber-physical systems in response to corrupted control systems.

and the accidental loss of a robot or a subset of robots due to a mechanical failure or enemy action. Identifying the swarm vulnerabilities expressed in terms of loss of communication connectivity may prevent detrimental events such as the inability for the swarm to maintain an acceptable level of target detection and identification.

The classifier proposed in this report outputs a label determining whether or not a local representation is deemed vulnerable. A local representation of a network corresponds to a subgraph of the network including edge, node, clique, cluster, community (Caldarelli, 2007). The training and classification phases discussed in this report are performed with the two aforementioned classes of network, namely, random geometric networks (fuse model) and with a class of mobile ad hoc communication network (mobile robot swarm). Random geometric graphs are characteristic of randomly deployed wireless sensor networks. As shown in this report, a classifier generated by PREVU can be interpreted as a predictive vulnerability model as it maps a set of appropriately selected structural features characterizing every network local representation to a set of corresponding vulnerability labels.

Experiments conducted to evaluate PREVU consist in training a set of classifiers and then evaluating classification rates. The two aforementioned avalanche mechanisms, that is, the modified fuse model and the robot swarm, both serve two purposes:

- generating training datasets, and
- generating datasets used to assess the performance of classifiers.

Eighteen simple classifiers are considered. Combined classifiers using six combination functions applied to the best six simple classifiers are also analyzed. The misclassification rate of a classifier is computed by comparing the actual vulnerability labels obtained from avalanche simulations with vulnerability label estimates resulting from the application of a classifier. The best classifiers (simple and combined) achieve classification success up to 70%, meaning that up to 70% of the vulnerabilities as a result of threat realizations (node removals) were correctly detected.

1.3 Outline of the report

The report is divided as follows. A brief literature review is presented in Chapter 2. It should be noted that further details and references can be found in Léchevin and Maupin (2009). Modeling of the two proposed networks and avalanche mechanisms is the subject of Chapters 3 and 4. Motivations for designing a vulnerability analysis tool are also discussed in these two chapters. The classifier-based model, which is central to PREVU, is described in Chapter 4, where notations, definitions, design methodology, and structural features of interest are provided. Chapter 6 presents key results of a first version of PREVU applied to the analysis of the vulnerabilities of the two networks presented in Chapters 3 and 4 (fuse model and robot swarm). Concluding remarks are provided in Chapter 7, including future work.

2 Avalanche modeling

Forecasting the pattern of an avalanche is not a trivial task. Various mechanisms leading to cascading events have been reported in Léchevin and Maupin (2009). Several approaches can be drawn.

The use of simplified models allows the application of techniques that pertain to statistical physics, although the accurate prediction of avalanches occurring in actual technical networks is unlikely, owing to unrealistic models. For instance, the fibre bundle and the fuse model are simplified abstractions of the mechanism of cascading failures in electrical circuits, or more generally in networks characterized by flow conservation such as Kirchhoff's laws. These models can also be used to analyze the occurrence of cracks in loaded materials represented by means of a network of fuses (Hansen, 2005). The distribution of cracks, as the result of edge removal, can be derived from statistical physics techniques. The extension of this approach to manmade networks does not seem feasible.

Approaches consisting in defining the load of a node by its betweenness centrality[9] or variants may also lead to cascading failures by comparing the load of each node resulting from the removal of a set of edge and nodes to a prescribed load limit (Motter and Lai, 2002; Wang and Rong, 2009). While providing interesting insights concerning the importance of structural features on the possible occurrence of avalanches, there is no evidence that the models proposed in (Motter and Lai, 2002; Wang and Rong, 2009) comply with the power flow that is specific to networks ruled by energy conservation, and thus by Kirchhoff's laws.

Branching processes have been leveraged, with some success, to approximate the pattern of cascading failures in response to the occurrence of potentially harmful events (Dobson *et al.*, 2010). Such probabilistic models give the number of line removals at each step of the iteration given the number of lines at the preceding step. A queue-model interpretation with a single server is also provided in (Dobson *et al.*, 2005).

On the other hand, approaches followed by industries for analyzing and predicting the behaviour of networks aim to comply with design constraints, technological and economic constraints, functions of network components, and various design trade-offs, which are notions that are understated by mainstream sciences of networks (Alderson and Doyle, 2010). For instance, the power system industry develops forecasting tools based on the integration of various techniques such as the simulation of dynamic model coupled to power flow model, dynamic decision event trees, and *N-k* contingency analysis (Heydt, 2001), each of which attempts to faithfully represent a facet of network behaviour.

Networks may experience outage and denial of service despite failure detection and recovery mechanisms. In particular, uncontrolled and unexpected propagation of an isolated problem beyond local proportions when the network operates close to its physical limits is likely to arise under non-optimal operating conditions, occurring for instance near maximum allowable capacity. In the next two chapters, two types of avalanche mechanisms are detailed for the

[9] Betweenness refers to a measure of the importance of a vertex or edge in a graph. The betweeness is obtained by counting the number of shortest paths that pass through a vertex (vertex-betweenness) or through an edge (edge-betweenness) to connect two parts of the graph (Freeman, 1979).

purpose of illustrating the proposed pattern-recognition-based vulnerability analysis. Both mechanisms express the interplay between

- the effect of the redistribution of some physical quantities through a network (e.g., power flow, distance between communicating robots) after a failure or a set of failures, and
- constraints, whereby link capacity must not be exceeded.

In the variant of the fuse model presented in Chapter 3, an edge between two nodes is removed whenever the flow through it exceeds a prescribed capacity. In this network, flows satisfy Kirchhoff's laws.

A communication link between two robots of the tactical mobile cloud presented in Chapter 4 is disabled whenever the distance separating the two robots is greater than a prescribed communication range. The distance between two robots is the output of a dynamical system consisting of robot dynamics, communication processes, and a motion strategy.

3 Fuse model variant

Despite its relative simplicity, a variant of the fuse model is used as one example of mechanism allowing to generate avalanches and thus the data sets that will be used to train the classifier. This model is used for several reasons.

First, it is flexible enough to generate data from virtually any types of graph structures, which will be very useful for

- feature extraction based on structural properties of the graph,
- classifier training, and for
- analyzing the impact of possible network representations on the accuracy of avalanche prediction.

Second, structural and flow parameters of the network, constraints, and the number and location of source and sink nodes can be selected arbitrarily.

Last, the classification techniques will not be dependent on manmade networks' specific technological considerations such as design constraints and protocols.

However, this model does not account for the wide variety of network components such as protection and control systems, whose behaviour may contribute to triggering and feeding cascading failures. That is the reason why it is intended to complete the current work with tests involving actual data such as those obtained with the tactical mobile cloud whose model is presented in the next section.

3.1 Principle

The proposed avalanche mechanism is equivalent to a sequence of steady states, resulting in a discrete-event system whose state jumps at each iteration step k, (i.e., at time instant t_k), from an equilibrium of the system to the next one, if any. The jump condition (JC) is related to the infringement of the flow constraint of every edge of the network and to the initial structural perturbation as described later on.

More precisely, the envelope of the time trajectories corresponding to the flows through network edges, occurring in nominal conditions, must remain within some prescribed safety domain, as illustrated in Figure 2. At t_1, one of those trajectories crosses the safety boundary outward, which entails the removal of the corresponding edge and in turn impacts the entire tube through the jump condition (JC1). Typically, JCi where $i=\{1,2,\ldots\}$ is a set of algebraic conditions that initializes the modified continuous-time, dynamical system that evolves over $[t_i, t_{i+1})$. One of the remaining trajectories is particularly affected and leaves the safety domain at t_2. A sequence of events occurring at t_1, t_2, t_3, and t_4 along with jump conditions JC1 to JC4 may lead to an avalanche, which involves the loss of a fraction of the network edges. (Léchevin and Maupin, 2011).

It should to be noted that the hybrid nature of the avalanche mechanism proposed in the next section is a simplified version of the avalanche dynamics shown in Figure 2 since the continuous-time dynamics leading to transient behaviours is disregarded.

Indeed, it has been shown in (Simonsen *et al.*, 2008) that accounting for the transient may reveal vulnerabilities unseen by it discrete-time counterpart. The discrete-event model is thus used as a first approximation of very large networks whose continuous-time transients are not considered by the classificatory problem. The model of avalanche is applied to various types of graphs, which are randomly generated.

Random geometric graph, as illustrated in Figure 4(a), small-world graphs, and graphs with exponential degree distribution and power-law distribution are graphs of interest since some of their structural properties have been shown to comply with those of several networks such as power networks (Bakke *et al.*, 2006), the Internet (Barabási, 2007), social network (Dorogovtsev and Mendes, 2004), and wireless communications networks (Penrose, 2003).

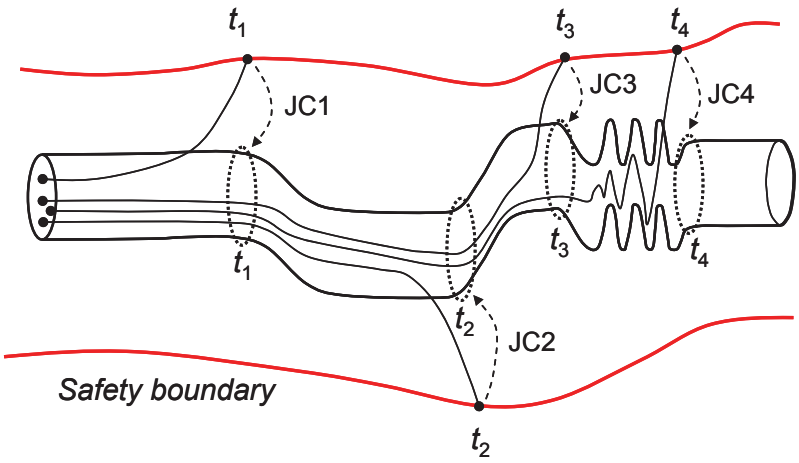

Figure 2: Onset of an avalanche in the hybrid time domain.

3.2 Model

Given a graph, its weighted Laplacian matrix, $L=D-A$, is computed. Matrices D and A stand for the degree matrix[10] and the adjacency matrix of the graph, respectively (Godsile and Royle, 2001). Each nonzero and nondiagonal entry (i, j) of A is equal to the weight characterizing the corresponding edge (i, j), which links nodes i and j. The ijth entry of A is zero when nodes i and j are not connected. Building upon analogies with electric circuits, the weight represents the conductance of the branch linking i to j, whereas each nonzero entry of a standard adjacency matrix is equal to one. The graph is assumed undirected. Applying Kirchhoff's laws, one obtains at time instant t_k

$$\begin{aligned} o_k &= L_k s_{V,k}, \\ s_{E_{ij},k} &= L_k(i,j)(s_{V_j,k} - s_{V_i,k}) \end{aligned} \tag{1}$$

[10] D is a diagonal matrix. The iith entry of D is the degree (or valency) of node i; that is, the number of edges incident to i.

where $s_{E,k}$ and $s_{V,k}$ stand for the state vectors characterizing the edge set E and the vertex set V of the graph at t_k, respectively. $s_{E_{ij},k}$ and $s_{V_i,k}$ stand for the state associated with edge (i,j) and node i, respectively. $L_k(i, j)$ denotes the (i,j) entry of the Laplacian matrix at t_k. Each entry of o_k is equal to zero when the corresponding node is neither a source nor a sink; otherwise, it is equal to an incoming or outcoming flow accordingly.

Consider, for instance, the three-node network shown in Figure 3 and characterized by mutual admittances y_{12}, y_{23}, and y_{31} and by self-admittances y_1, y_2, and y_3. Its weighted Laplacian matrix (or admittance matrix) and its state vectors are given by $s_{E,k} = [I_1, I_2, I_3]^T$, $s_{V,k} = [V_1, V_2, V_3]^T$, and

$$L_k = \begin{bmatrix} y_1 + y_{12} + y_{13} & -y_{12} & -y_{13} \\ -y_{12} & y_2 + y_{12} + y_{23} & -y_{23} \\ -y_{13} & -y_{23} & y_3 + y_{13} + y_{23} \end{bmatrix}.$$

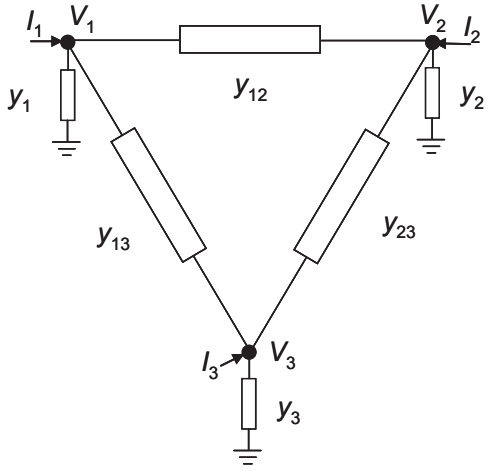

Figure 3: Example of a circuit with shunt impedances.

Inflows I_i are nonzero whenever the node is connected to a source or a sink node. Each pair of source E_{i+} and sink nodes E_{i-} is assumed characterized by flow variables $s_{E_{i+},k}$ and $s_{E_{i-},k}$, and the state difference $s_{V_{i+},k} - s_{V_{i-},k} = \delta s_{V_i,k}$, where $s_{V_{i+},k}$ and $s_{V_{i-},k}$ stand for the entries of $s_{V,k}$ that correspond to source V_{i+} and sink V_{i-}, respectively. The knowledge of the flow variables and state differences allows

- to simplify (1) by eliminating algebraic dependencies, which leads to

$$0 = \bar{L}_k \bar{s}_{V,k} + f(\delta s_{V_1,k}, ..., \delta s_{V_p,k}), \tag{2}$$

- and to compute state $s_{V,k}$ and then $s_{E,k}$.

$\bar{L}_k$, $\bar{s}_{V,k}$, and f in (2) stand for the reduced-order Laplacian matrix and vertex state, and for a function of source-sink state differences.

Unfortunately, the Laplacian matrix is singular when self-admittances are equal to zero. Moreover, computing the Laplacian matrix inverse, if any, can be computationally cumbersome

when dealing with very large networks. State $s_{V,k}$ can be approximated by leveraging the Jacobi relaxation method (Batroumi and Hansen, 1998), which is applied to (2) by adding a derivative term in (2)

$$c\frac{d\bar{s}_{V,k}(t)}{dt} = \bar{L}_k \bar{s}_{V,k}(t) + f(\delta s_{V_1,k}(t),...,\delta s_{V_p,k}(t)), \tag{3}$$

for all $t \in [t_k, t_{k+1})$ and $c>0$, which is selected such that

$$\lim_{t \to t_{k+1}} \bar{s}_{V,k}(t) = \bar{s}^*_{V,k}. \tag{4}$$

$\bar{s}^*_{V,k}$ denotes the network equilibrium resulting from the jump at t_k caused by the infringement of flow constraints. In steady state, system (3) is equivalent to (2) since $d\bar{s}^*_{V,k}/dt \equiv 0$. The iterative process corresponding to (3) can be expressed as

$$\bar{s}_{V,k,i+1} = \bar{s}_{V,k,i} + \varepsilon(\bar{L}_k \bar{s}_{V,k,i} + f(\delta s_{V_1,k,i},...,\delta s_{V_p,k,i})), \tag{5}$$

where $\varepsilon = T/c$. $\bar{s}_{V,k,i}$, $\delta s_{V_l,k,p}$, and T denote state variables $\bar{s}_{V,k}$ and $\delta s_{V_l,k,i}$, at the ith iteration of (5) over $[t_k, t_{k+1})$, and the time step, respectively.

Jump condition: The relaxation process in (5) is triggered at t_k when at least one flow constraint is no longer satisfied, leading to a jump entailed by setting several entries of the Laplacian matrix to zero, and is stopped when the relative error in $\bar{s}_{V,k,i}$ is less than or equal to some prescribed value.

Stopping rule: The avalanche, that is, the sequence of jumps is stopped either when the connection between sink and source nodes is lost or when every remaining flow constraint is satisfied.

3.3 Network vulnerability: motivation for developing an analysis tool

Figure 4 shows seven such transitions when one node, randomly selected, has been removed from the 1000-node, random geometric network. Two nodes i and j are connected if dist(i,j) is less than a prescribed value $r<1$. The weight of each edge is a decreasing function of the internode distance; that is, the (i,j) entry of L_k is equal to $-(1-\text{dist}(i,j))/R$, where R denotes an upper bound of all internode distances.

To obtain interesting network behaviour in response to structural disturbances, the flow constraint is set to 150% of the network equilibrium in nominal operating condition. Higher flow constraints would decrease the frequency of avalanche occurrence and even prevent the network from experiencing avalanches. Whenever the absolute value of the flow of an edge exceeds 1.5 times the corresponding flow equilibrium obtained before any edge or node removal, the edge is open for all subsequent t_k. The avalanche stops after the last transitions whereby one edge is lost. The

source and sink nodes of the network resulting from the avalanche are still connected owing to the fact that the equilibrium satisfied the flow constraints.

In Figure 4(a), the nodes of the random geometric graph are drawn from a uniform distribution over [0,1]×[0,1]. Each edges is established whenever the distance between the corresponding pair of nodes is less than or equal to r=0.047, giving rise to 495713 edges. In Figure 4(b), two nodes are disabled from the nominal network in Figure 4(a), entailing the succession of seven jumps each of which occurs from one equilibrium to another. Transitions (a)→(b), (b)→(c), (c)→(d), (d)→(e), (e)→(f), (f)→(g), and (g)→(h) entail the removal of 202, 49, 158, 47, 139, 123 and 20 edges, respectively.

The avalanche shown in Figure 5 differs quantitatively and qualitatively from that in Figure 4 although both avalanches are triggered from the same network. The removal of node 2 leads to a completely different avalanche than observed in Figure 4, where node 1 was removed. Transitions (a)→(b), (b)→(c), (c)→(d), (d)→(e), and (e)→(f) entail the removal of 2, 150, 6, 2, and 3 edges, respectively.

As suggested by the dynamics shown in Figure 4 and Figure 5, predicting the final state of the network following a possible avalanche, or estimating the cost of network malfunctioning is intricate since the removal of a limited set of edges and nodes is not necessarily localized near to a salient feature, if any, of the network. Loss of edges and nodes depends on the flow equilibrium established prior to a change of network topology. It thus motivates the recourse to network vulnerability analysis techniques that will exploit signals and systems properties of the network. This approach amounts to assessing the sensitivity of the flow equilibrium to structural changes characterizing the network. A node of the network is deemed vulnerable when the cost entailed by its removal and the possible sequence of cascading events is greater than a threshold expressing an upper bound of acceptable costs.

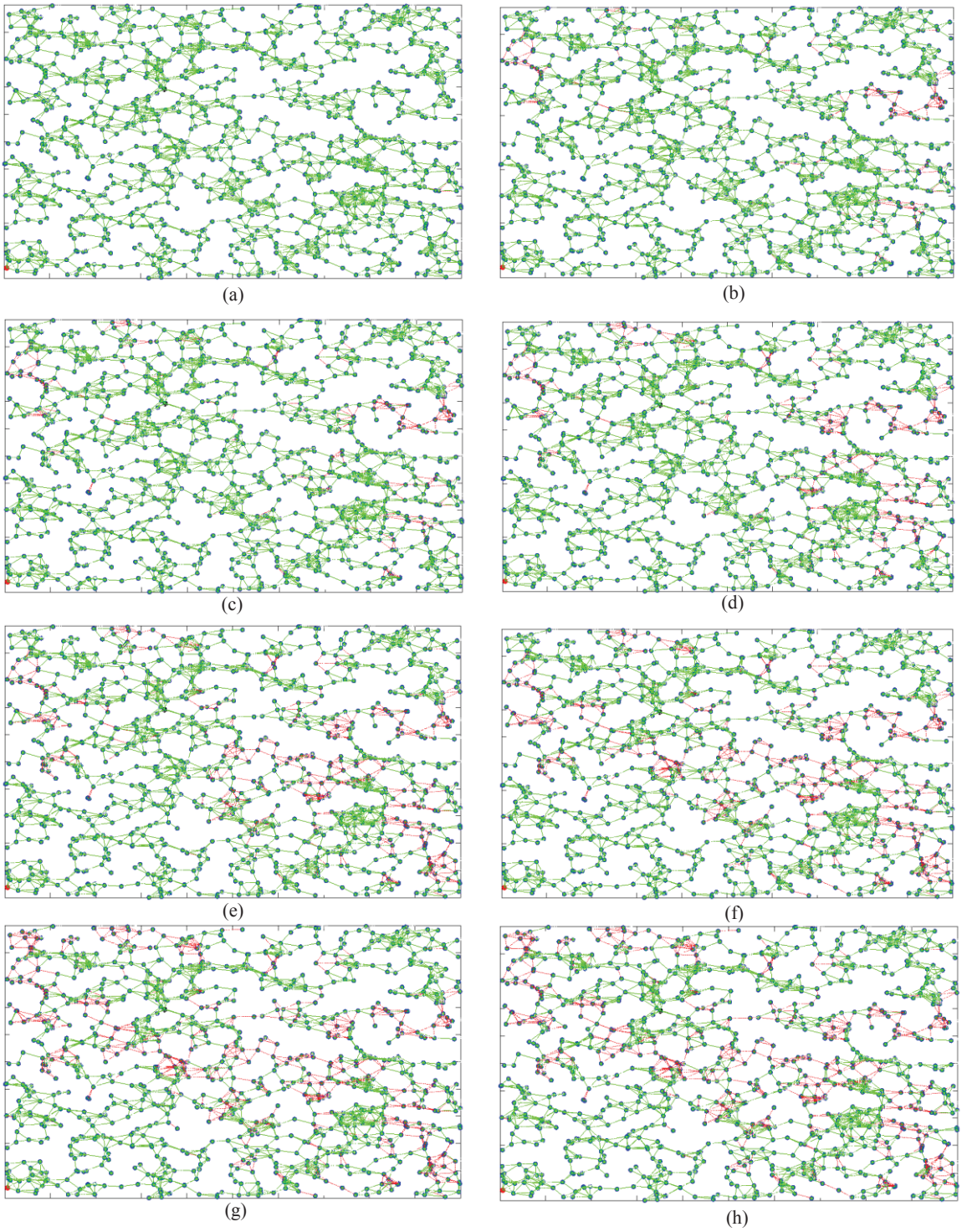

Figure 4: Sequence of networks (r=0.047) following the loss of one node (node 1).

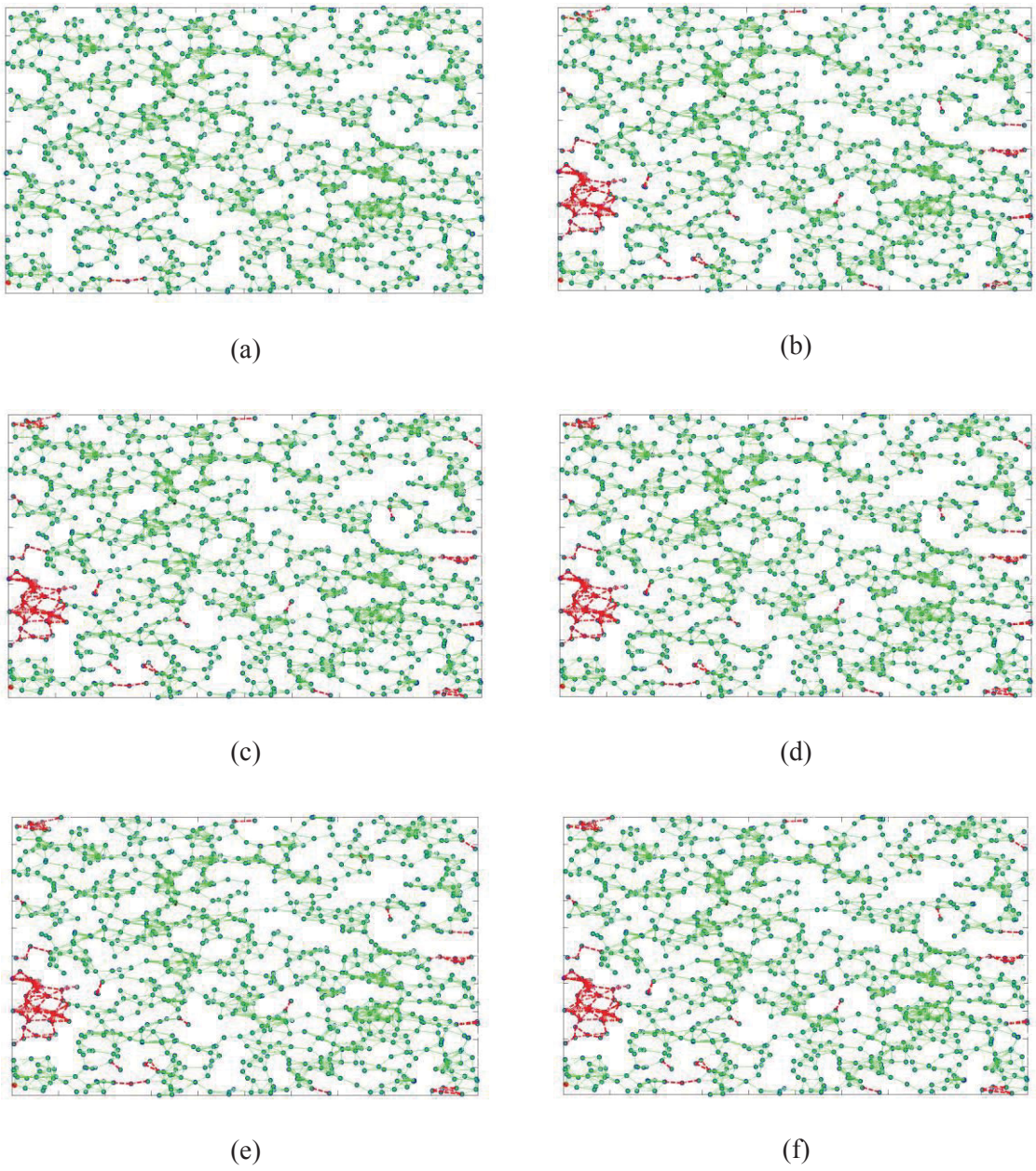

Figure 5: Sequence of networks (r=0.047) following the loss of one node (node 2).

This page intentionally left blank

4 Tactical mobile cloud

4.1 Definitions, assumption and objective formulation

The objective for the robot swarm presented in this chapter is to perform terrain surveillance coverage or to maintain client connectivity in as economical manner as possible, which implies successfully combining the swarm's goal of dispersing itself over a wide area while reducing unnecessary coverage duplicity at the local node level. One of the major threats lies in the swarm's individual robot's lack of self-awareness. PREVU applied to the robot swarm thus aims to provide and maintain a state of situation awareness for a group of agents, namely robots, observing a scene or maintaining the communication connectivity of a group of clients evolving in a remote area. In the context of the present surveillance swarm monitoring problem, the vulnerability pattern recognition entails swarm members automatically identifying situations that could pose a connectivity threat to the swarm.

The swarm model and motion strategies presented in this chapter follow from Jousselme *et al.* (2012a, 2012b), which result from collaboration with the US Military Academy of West Point.

The notation and basic definitions are now introduced. Let $R=\{r_1, \ldots, r_N\}$ be the set of robots and $C=\{c_1, \ldots, c_M\}$ the set of clients. The set C combined with their spatial location is a *configuration*. We denote ρ as a robot's unique communication range, which could be adjusted based on environmental demands. Next, $E=\{e_1, \ldots, e_N\}$ is the set of communication links between the robots and $G_c=(R, E)$ is the corresponding communication graph. Let N_c, be the set of node coordinates. Two robots r_i and r_j are separated by a distance of d_{ij} and are connected if their distance is less than ρ. In the remainder of this section, d_{ij} will denote the distance between two robots, two clients, or one robot and one client. We assume that

- indirect links are possible through intermediate nodes acting as relays, and
- at least one of the nodes is connected to an external communication node such as a satellite or UAV.

It is thus assumed that there exists a communication resource capability within the network to ensure that clients' messages are handled properly through the mobile cloud via an external wide range communication relay.

The initial state shown in Figure 6 corresponds to an equilibrium state in which the three surveillance objectives are satisfied. Robots are represented by red stars, while clients are blue circles. The communication connection between two robots is represented by black lines.

The objectives of the motion strategy consist of

- maintaining the network connectivity,
- maintaining the client's communication coverage, and
- maintaining an acceptable level of sensing coverage.

It should be noted that maintaining the network connectivity at each time instant is a strong constraint. This time constraint could be relaxed, to some extent, by considering a time sequence

of networks that are jointly connected, that is, the union of the sequence of networks results in a connected network. This relaxation is not implemented in the motion strategy in the next section.

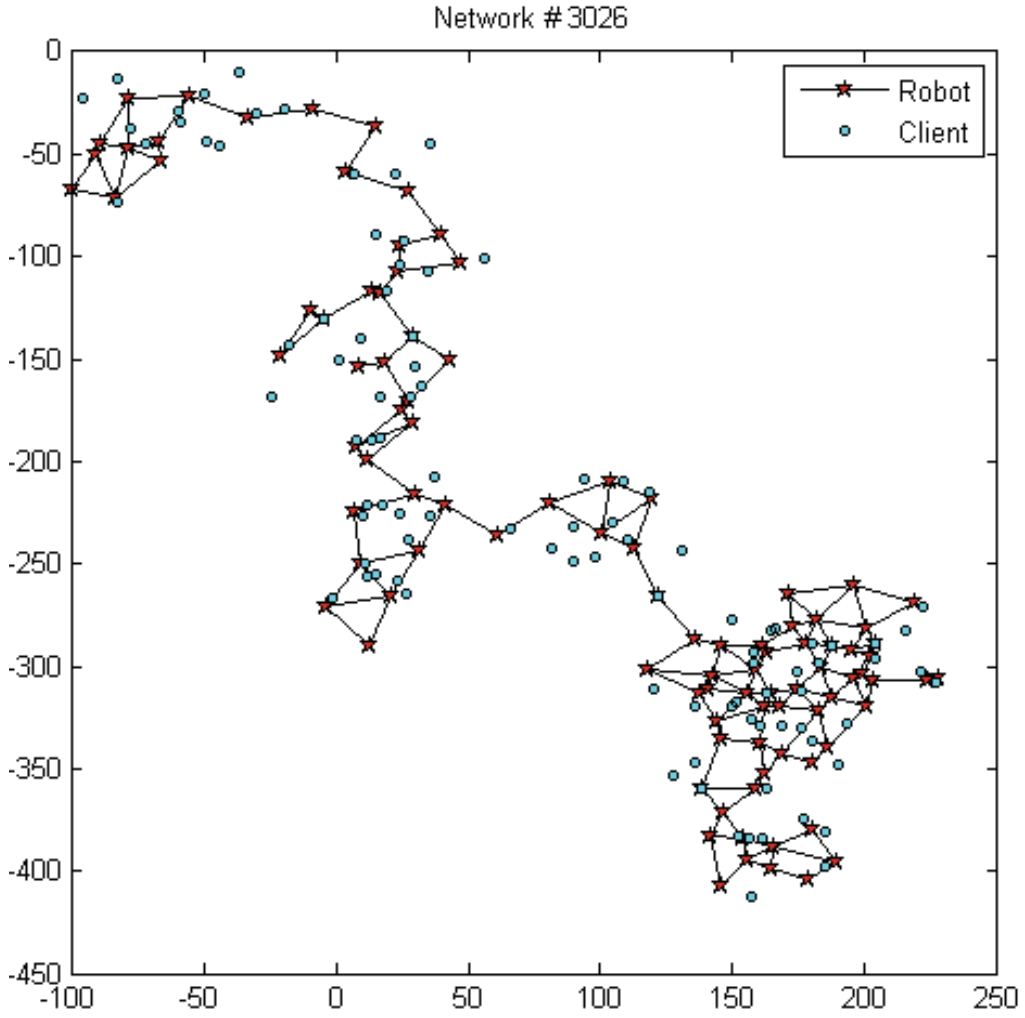

Figure 6. A tactical mobile cloud of 100 robots for communication coverage of 100 clients.

Given the stochastic nature of the clients' move as well as of the robots' performances, the equilibrium state where the last three objectives are satisfied may be weakened. Two major causes of such a weakness are:

- a loss of a node (robot's failure), and
- a loss of a link caused by a client's move, or an obstacle to the communication link.

The following assumptions are considered:

- the swarm of robots has a motion strategy to recover that state of equilibrium and ensure the clients' coverage, the network connectivity, and the network sensing coverage;
- both the original network structure and the motion strategy are robust enough to absorb small perturbations.

The network vulnerability assessment presented in Chapter 5 and applied to the robot swarm in Chapter 6 is used as an early warning system and is intended to modify the motion strategy, should the perturbation grow larger than what the network can absorb. Closing the loop on the motion strategy will be the topic of future work.

4.2 Motion strategies

Each client knows its own location via a location finding device such as a GPS. The continuous behavior of clients and robots are modeled as a sequence of instances. At each time instance, each robot evaluates the current position of its clients as well as the set of neighbouring robots. Based on this evaluation, each robot calculates the optimum position to provide coverage to clients as well as to maintain connectivity to at least one neighbouring robot. If both goals cannot be attained, then the priority is to maintain coverage for clients.

At the initial equilibrium state, the network optimally covers the set of clients (communication coverage), optimally covers the area under consideration (sensing coverage) and is fully connected (each client is able to communication with an external node relaying the communication). That means in particular that

- each client is within the communication range of at least one robot (clients' coverage), and
- each robot is within the communication range of at least one other robot (network connectivity).

The robot motion strategy is based on the work in (Bezzo and Fierro, 2011). Each robot in the swarm follows a spring-mass virtual physics. In particular, the motion of the i^{th} node in a swarm of N robots is as follows:

$$\ddot{X}_i = \left[\sum_{j \in S_i} k_{ij} (l_{ij} - l_{ij}^0) \hat{d}_{ij} \right] - \gamma_i \dot{X}_i, \tag{6}$$

with i=1,..,N_r and $i \neq j$.

X_i, $\dot{X}_i$, and $\ddot{X}_i$ are the robot's position, velocity, and acceleration respectively. S_{ij} is the set of robot neighbours for the ith robot while l_{ij} is the length of the virtual spring between ith and jth robot. The symbol l_{ij}^0 represents the spring relaxed length while $\hat{d}_{ij}$ is the unit vector indicating the direction of the spring force. Finally, the equation uses two constants, k_{ij} and γ_i. The former is the spring constant between robots i and j and the latter is the damping coefficient with a value assumed to be greater than zero.

The spring-mass model described above could create a mesh with limited expandability and thus restrict a swarm's ability to cover a region adequately. To overcome this challenge, the inter-node spring-mass links is implemented within the constraints of a Gabriel graph (Bullo et al., 2009).

In particular, a spring is formed between two robots i and j if and only if there is no k robot inside the circle with a diameter formed by $\overline{ij}$ (Bezzo and Fierro, 2011). More formally, consider φ that evaluates to 1 if there is a spring between robots i and j and 0 otherwise. Hence, we have the following:

$$\varphi_{ij} = \begin{cases} 1 \text{ if } \hat{xkj} \le \pi/2, \\ 0 \text{ if } \hat{xkj} > \pi/2, \end{cases} \tag{7}$$

with i, j, k=1,..,N_r and $i \neq j$ and $j \neq k$. $\widehat{xkj}$ is the interior angle formed by robots x, k and j.

Given the supporting equations above, the distributed algorithm that each robot in the swarm executes reads as follows.

```
for each robot_j neighbour of robot_i do
    while l_ij ≠ l_ij^0 do
          Compute the next move from (6) subject to (7)
          if robot_i detects a user then
          place a spring connection between robot i and the user using (7)
          end if
          return X_i, X_i^&, and X_i^&&
    end while
end for
```

4.3 Swarm coverage objective

4.3.1 Definition

The problem formulated in Section 4.1 involves the following two types of coverage:

- a sensing coverage, since the swarm is responsible for covering a given area and detect possible intruders, and
- a communication coverage, that can be split into:
  - *clients' coverage*, since the swarm of robots is responsible for providing communication coverage to the set of clients within the given area, and
  - *robots' coverage*, since the swarm is responsible for maintaining the network connectivity.

We consider the following general definition of coverage, which encompasses the three notions above. Let q be an emitter and p be a point of interest of a given area. The coverage provided by q at p is expressed as follows:

$$cov(p,q) = f(d_{pq}), \tag{8}$$

where d_{pq} is the distance separating p from q and f is a decreasing function. Both p and q are located in space and are represented by their spatial coordinates along x and y axes, *e.g.* (x_p,y_p).

4.3.2 Connectivity

As an instance of (8), the network connectivity is defined as follows. We say that two robots are connected if they are in their respective range of communication. We then define

$$\begin{aligned} con(r_i, r_j) &= 1 \quad \text{if } d_{ij} < \rho_c, \\ &= 0 \quad \text{otherwise,} \end{aligned} \tag{9}$$

where ρ_c is the communication range of the robots and d_{ij} is the distance separating r_i form r_j. f is thus a step function of the distance and a value of 1 means then that a link exists between the two robots. We say that the global network connectivity holds if for each pair of robots (r_i,r_j) there exists a path linking r_i to r_j:

$$con(R) = 1 \text{ if } \forall (r_i, r_j) \in R^2, \exists (r_1,\ldots,r_m) \in R^m; (r_i, r_1), (r_1, r_2)\ldots,(r_m, r_j) \in E. \tag{10}$$

Alternative definitions could easily replace these binary definitions and define different connectedness indices such as the algebraic connectivity (Godsile and Royle, 2001). Indeed, allowing real values for the connectivity would increase flexibility in the definition of vulnerable states and lead to different cost functions.

4.3.3 Client's communication coverage

For simplicity, a binary definition of coverage is adopted and consists of the following two states: covered or not covered. However, this definition can be easily extended to other models, such as probabilistic ones. We define coverage provided by robot r_i to client c_j as

$$\begin{aligned} cov(r_i, c_j) &= 1 \quad \text{if } d_{ij} < \rho_c, \\ &= 0 \quad \text{otherwise,} \end{aligned} \tag{11}$$

where ρ is the communications range of r_i and d_{ij} is the distance between r_i and a client. The set of clients covered by robot r_i is then given by

$$cov(r_i, C) = \sum_{j=1}^{M} cov(r_i, c_j). \tag{12}$$

Given the coverage definitions from the perspectives of both the client and the robot, Equation (13) describes the global coverage of the network relative to the set of clients:

$$cov(R,C) = \sum_{i=1}^{N} cov(r_i, C) = \sum_{j=1}^{M} cov(R, c_j) = \sum_{i=1}^{N} \sum_{j=1}^{M} cov(r_i, c_j). \tag{13}$$

4.3.4 Sensing coverage

The sensing coverage provided by a robot r_i at a given point of interest p is given by the following exponential model (Jousselme and Maupin, 2012b)

$$cov(r_i, p) = \begin{cases} 1 & \text{if } d < R, \\ \exp(-\lambda(d_{ip} - R)) & \text{else,} \end{cases} \tag{14}$$

where λ and β are real parameters, d_{ip} is the distance between robot r_i and the point of interest p, R is the sensor range. It should be noted that other models of sensing coverage could be used to build the swarm model. Figure 7 shows an example of this degradation model that will be used in the simulations. The degradation model is a function of the distance between robot r_i, represented by a red star in Figure 7, and the point being observed. Within a range of 16 meters, the sensing coverage is assumed to be perfect while decreasing to reach 0 at around 30 meters.

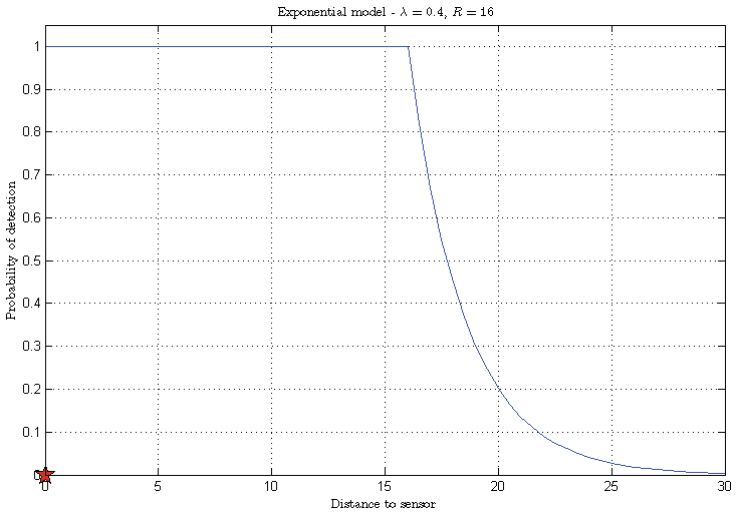

Figure 7. Exponential model of detection performance degradation.

Figure 8 shows a network of 100 robots and 572 possible positions of interest. Each circle represents the coverage at 16 meters according to the exponential model described above. Robots are represented by red stars, while positions of interest are black dots.

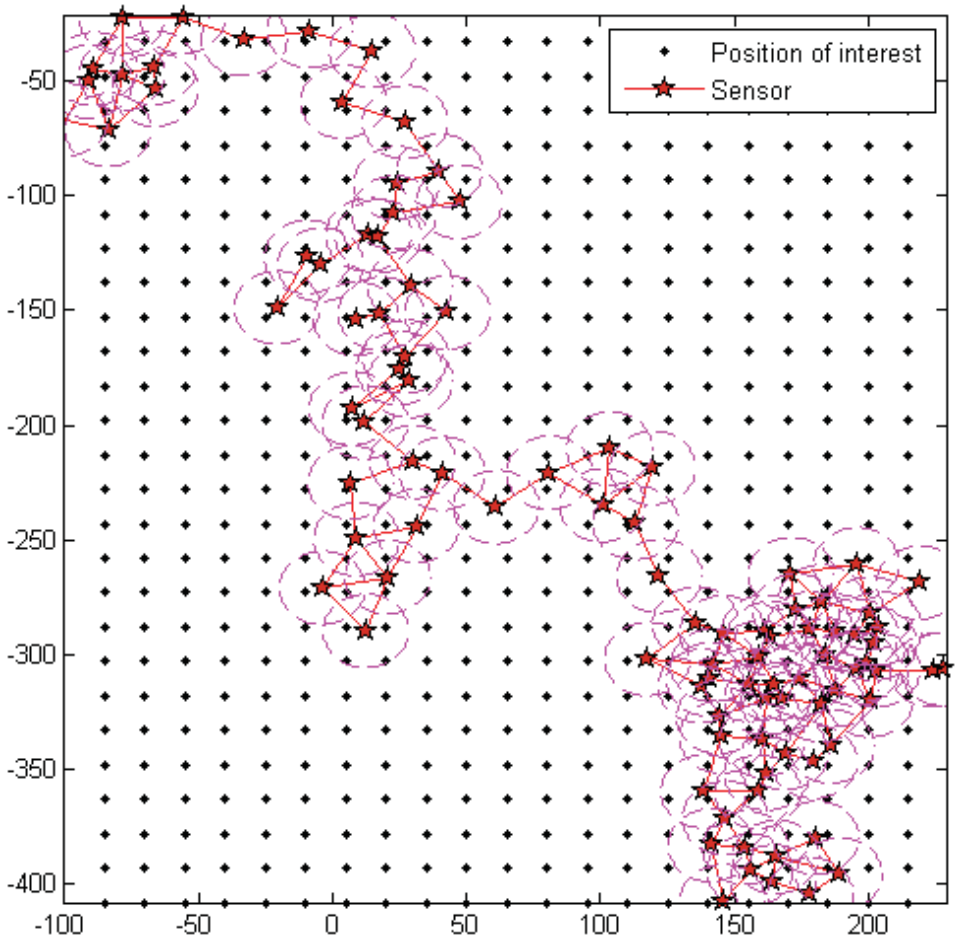

Figure 8. A tactical mobile cloud of 100 robots for sensing coverage of 572 positions of interest.

4.4 Network vulnerability: motivation for developing an analysis tool

A mobile network vulnerable state can be defined as an instance of the network state that may evolve in time until it affects the network functions and the completion of its goals. Endogenous and exogenous threats to the network include the robots' inability to proceed as intended, possibly due to hardware-software failures or malevolent acts, electronic warfare, obstacles, or unexpected client moves that cause some robots to move beyond their neighbours' communication range. A component of the graph, that is an edge, a node, or a subgraph, is classified as vulnerable when a graph connectedness-related cost associated to this component is above a prescribed threshold. A node is thus vulnerable if its loss (failure) leads to a break in the network connectivity.

The mechanism that may lead the network state to some undesired set of operating conditions, as a response to some threat or failure, is presented next, and will be instrumental in assigning labels for classification training (see Chapter 5).

Consider a sample set of possible client configurations C_0 (Figure 9(a)) and a corresponding robot deployment represented by graph $G_0=(R, E_0)$. Include also the set $N_{c,0}$ of node coordinates at time instant t_0 (encoded as attributes of the nodes). Various experiments are conducted by triggering the loss of a robot (Figure 9(b)). The occurrence at t_1 of this triggering event gives rise to an adaptive robot deployment (Figure 9 and Figure 10) as a result of the motion strategy presented in Section 4.2, whereby communication links can be either permanently lost (Figure 10) or re-established (Figure 9), depending on the relative distance to neighbouring robots.

From an initial optimal deployment in Figure 9(a), where blue dots and red stars represent the clients and the robots, respectively, the loss of two communication (Figure 9(b)) links is entailed by some triggering event. This event may result from the robot's inability to operate properly owing to software or hardware failures, robot destruction, or the presence of malware.

In Figure 10(a), robots are deployed adaptively to recover an optimal coverage of the area despite the loss of a robot (one link is recovered). In Figure 10(b), a new optimal robot deployment can be achieved without the missing robot; therefore, its loss is not classified as being a vulnerability to the initial robot network.

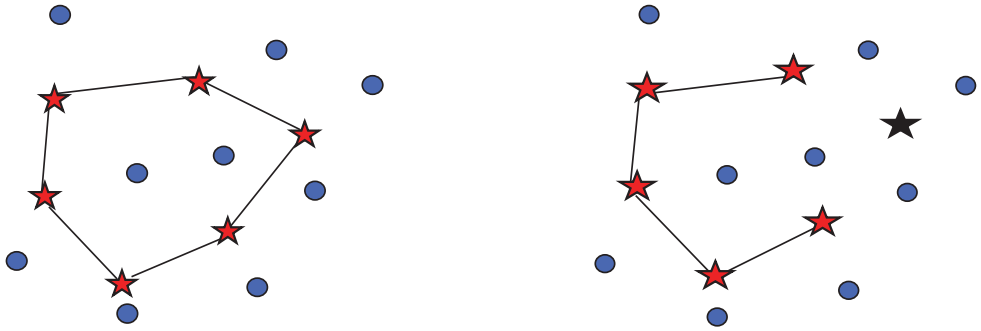

Figure 9. Initial state of a swarm of six robots.

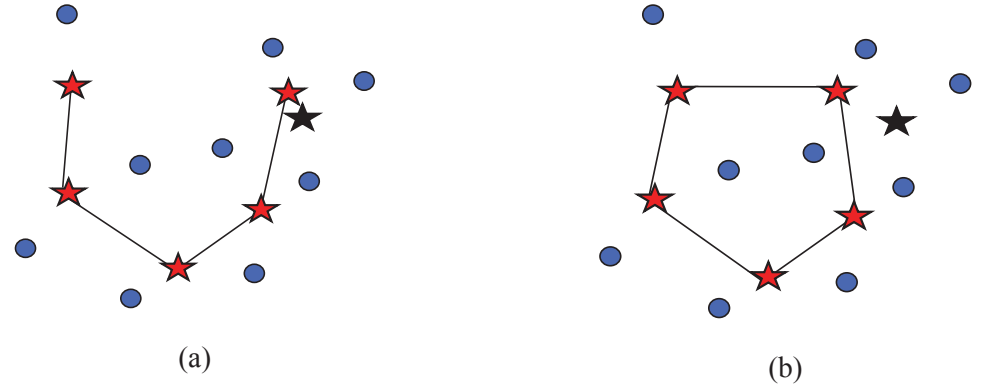

Figure 10. Robot loss is not classified as a vulnerability.

In Figure 11, the environment evolves in time and space leading to possible cascading events. Since the motion strategy prioritizes the coverage function, robots adaptive deployment results in a loss of connectivity (Figure 11(b)); that is, two operating robots are disconnected from the rest of the swarm.

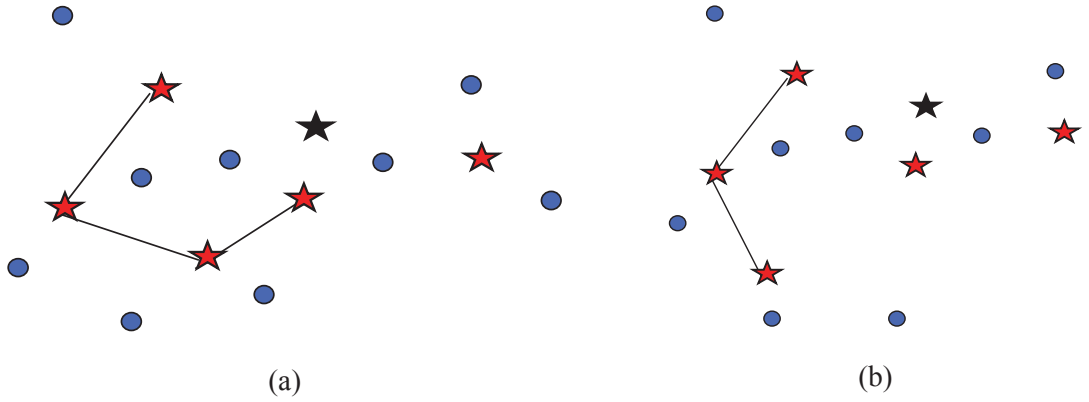

Figure 11. Robot loss is a vulnerability.

This hybrid dynamical system is thus characterized by switching time instants $\{t_1, t_2, \ldots, t_m\}$, where $t_{i+1} > t_i$. At t_i, the edge set jumps from E_i to E_{i+1}. An edge (i, j) is lost whenever the distance between robots r_i and r_j is greater than the communication range. It is assumed that the node set R remains invariant whether or not a robot is able to operate.

Stopping rule: The final time instant t_m is defined by the absence of any future triggering events such as a robot failure or a client move.

This page intentionally left blank

5 Structural-Feature-based modeling of networks for Vulnerability Pattern recognition

5.1 Preliminaries

Standard approaches used to identify the vulnerabilities of a dynamical system consist of performing a *N–k* contingency analysis based on the time-domain simulation of this system. *k* out of *N* components of a network are removed. The time evolution of the network state, which includes flows (edges) and potential functions (vertices), is obtained by simulating the network dynamics over a finite horizon, following the hybrid approach presented in Chapter 3. In so doing, one can verify whether the network state remains within its security domain. Simulations are started by exploiting the last available data that characterize the network and its environment. Starting from a set of initial conditions, $\binom{N}{k}$[11] simulations are required to achieve a *N–k* contingency analysis; therefore, the length of the prediction window as well as the value associated with *k* are determined to meet computational load constraints and decision-making constraints.[12]

The approach proposed in this chapter is based on pattern recognition. Based on data set obtained with such models as those presented in the last two chapters, a classifier is devised to build a representation that is particularly suited to vulnerability recognition problems. The classifier is thus not equivalent to simulating a network since it will not provide the time-evolution of the network state. The novelty of the approach consists in building a network model, namely, a classifier, that establishes a map between a set of network features and a set of vulnerability labels. The selection of the map and of the set of features seeks to minimize the error rate of a function that discriminate the vulnerable states from the non-vulnerable states, as shown in Section 5.4.

The PR approach is thus preferred over the model of network dynamics for the following reasons (Léchevin *et al.*, 2011):

- features are naturally geared to the modelling of classificatory problems;
- handling physics-based models of large interconnected networks as well as large measurement sets can be computationally prohibitive, thus requiring model and dataset reduction;

[11] The number of combinations obtained by selecting *k* unordered outcomes from *N* possible components of the network.

[12] The *N-k* contingency analysis amounts to solving a combinatorial optimization problem when a system dynamics is replaced by a set of algebraic equations obtained when the system is in steady state. Transients are thus disregarded even though they may behave abnormally. When contingencies result from the effect of exogenous threats such as a group of attackers, the identification of network vulnerabilities can be related to the defence planning of infrastructures, which can be expressed as a defender-attacker optimization problem (see Brown *et al.* (2005)).

- once the classifier is derived, fast and efficient recognition is expected when compared with the approach consisting in using network hybrid models.

However, adopting a PR approach may entail the following drawbacks:

- statistically significant training data is necessary to generate a classifier with satisfactory measures of performance;

- labelling is central to the vulnerability analysis since it defines how to attribute the vulnerability status to a local representation, whose state is characterized by the value of a score function. If the labeling process changes over time[13], a new classifier should be generated.

5.2 PREVU objectives

Given a knowledge base constituted of (*i*) past events that have affected a network or a class[14] of networks and of (*ii*) partial network information[15] obtained from measurement and observations, identify a set of vulnerable components (edges, nodes, or subgraphs) that is likely to affect the network, should an attacker exploit the knowledge of this set of components.

The identification of vulnerable network components consists of determining the identity and location of these components.

Vulnerability is to be related to risk assessment and thus to the cost associated with the loss of components or network inoperability. The definition of risk and determination of thresholds that expresses the level of acceptable risk should be considered as design parameters of PREVU.

5.3 Notation

Let $\mathcal{Y}=\{\mathrm{V}, \bar{V}\}, y\in\mathcal{Y}, \hat{y}\in\mathcal{Y}, y^x\in\mathcal{Y}^x, \mathbf{z}^r_{m+1}=(\mathbf{x}^r_m,\mathbf{y}^r)$, and $\psi(\mathbf{z}^r_{m+1},.)$ denote

- the set of possible labels V and $\bar{V}$, which stand for vulnerable and non-vulnerable, respectively,
- a class label,
- an estimated class label,
- a vector of x class label,

[13] The computation, perception, and acceptation of risk may change over time, which in turn impact the labelling process, for instance, by selecting new threshold values.

[14] A class of networks refers to the combination of (*i*) a class of graphs (e.g., random geometric, power law, small world) and (*ii*) a class of diffusion processes (e.g., cascading failures, behaviour, gossip, epidemics, computer viruses)

[15] Network states include edge and node attributes such flows and network topological properties such as various notions of centrality.

- a training data set (features and labels) of r local representations $S_i \in S(\omega)=2^{\mathcal{N}}$ of the network ω, and
- the trained classifier, respectively.

A local representation S_i (Figure 12) of a graph $\mathcal{G}=(\mathcal{N},\mathcal{E})$, where $\mathcal{N}$ and $\mathcal{E}$ denote the edge set and node set of $\mathcal{G}$, corresponds to any possible subgraphs including nodes, edges, clusters, cliques, and communities (Caldarelli, 2007). Indeed, the selection of the local representations constitutes a design parameter of the pattern recognition toolbox.

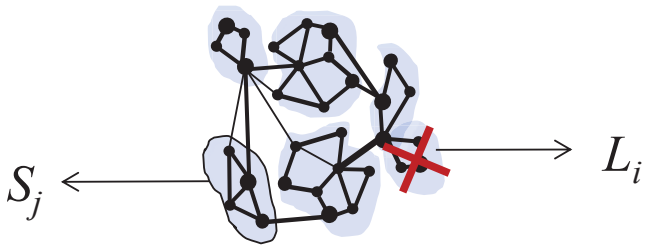

Figure 12. Local representations of a network.

The loss of S_i entails a cost L_i, expressing performance degradation, inoperability, or decrease of the quality of service (Figure 12). Following (Léchevin *et al.*, 2011a), a risk function, with L_i as argument, allows to define the vulnerability of a network component as follows. A set $\theta^*(\omega)$ of components of the network is classified as vulnerable when the protection effort that could be applied to this set minimizes the risk function given constraints involving cost functions. Alternatively, $\theta^*(\omega)$ could be determined by identifying, from the training data set z, the local representations S_i that have been lost following the occurrence, if any, of a major detrimental event (cascading failure, service disruption, etc).

$\mathbf{y}^r=[\,y^1 \ldots y^r]^T$ and $\mathbf{x}_m^r$ are the vector of labels and the table of m features representing the set $S^r(\omega)$ of r local representations of ω expressed as

$$\mathbf{x}_m^r = \overbrace{\begin{matrix} x_1^1 & x_2^1 & \ldots \\ x_1^2 & x_2^2 & \ldots \\ \mathrm{M} & \mathrm{M} & \mathrm{M} \\ x_1^r & x_2^r & \ldots \end{matrix}}^{\text{Local}} \overbrace{\begin{matrix} \ldots & x_{m-1}^1 & x_m^1 \\ \ldots & x_{m-1}^2 & x_m^2 \\ \mathrm{M} & \mathrm{M} & \mathrm{M} \\ \ldots & x_{m-1}^r & x_m^r \end{matrix}}^{\text{Global}}.$$

Features are divided into two classes, depending on whether local information about a node or global information about the network is required. Further details on the features used for the vulnerability analysis of networks are given in Section 5.6.

5.4 Basic principles of classification

A classifier is derived, at the training phase, from a labelled data set $\mathbf{z}_{m+1}^r=(\mathbf{x}_m^r,\mathbf{y}^r)$, where $\mathbf{x}_m^r$ and $\mathbf{y}^r$ stand for a r-by-m feature matrix, which characterizes the objects to classify, and the label

vector, respectively. The feature matrix consists of *r* instances of *m*-feature vectors. Two sets of features are shown in Figure 13 and clearly form two classification (or decision) regions separated by a classification (or decision) boundary (Duda et al. 2001).

A classification problem aims to find a mapping ψ from the feature space to the label set yielding a score for the respective class where an object might be. The mapping, illustrated by the decision boundary in Figure 13 and Figure 14, is not unique. In Figure 13, LDC stands for linear discriminant classifier whereas Parzen classifier belongs to the set of nonparametric classifiers. Numerous classifier design methods, each of which is based on a specific discriminant function, are proposed in the literature. These methods are based on probabilistic models or on approximations of discriminant functions (See Kuncheva (2004) and references therein for a taxonomy of classified method designs).

The classifiers derived is then tested, possibly refined and adapted online when used with actual data. A set of observational data leads to a corresponding set of scores each of which indicates a class to which the observation belongs to (Figure 14).

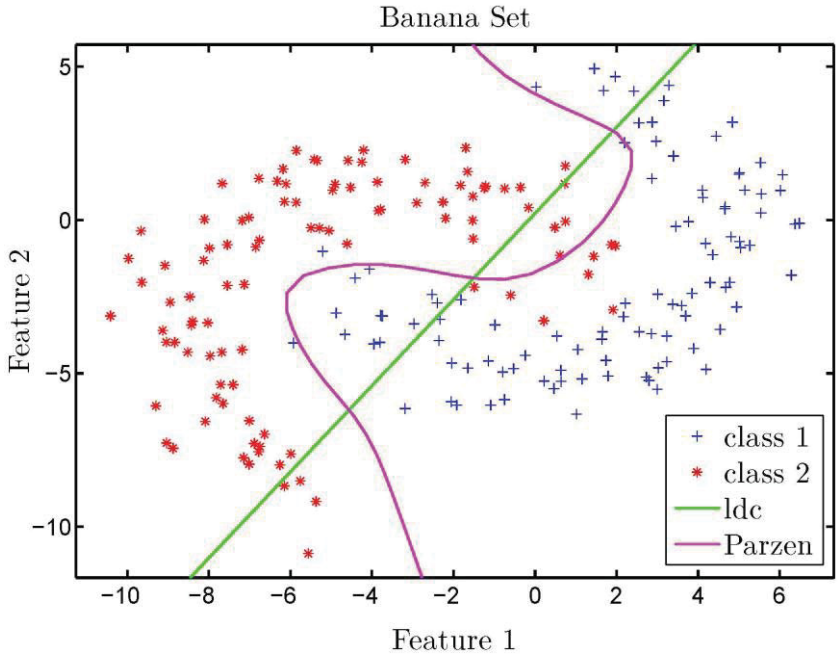

Figure 13. Building the decision boundary from a training data set (Jousselme and Maupin, 2012d).

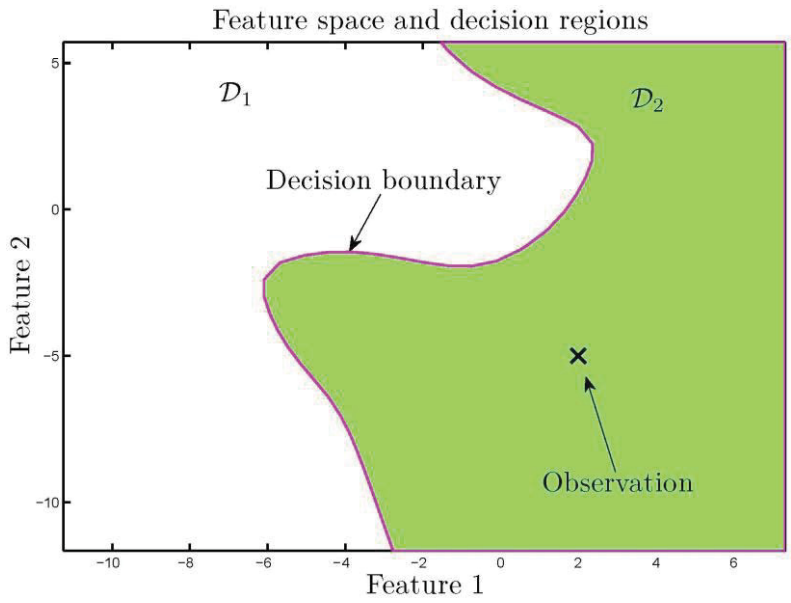

Figure 14. Assign a class in the feature space (Jousselme and Maupin, 2012d).

The classification performance is defined by means of various classification error function some of which express classification rate and misclassification rates, as shown in Table 1, where TP, FP, FN, and TN stand for true positive, false positive, false negative, and true negative respectively.

Table 1. Confusion matrix given a classifier ψ, a feature vector x, and a class label y.

	True label	
	y	$\neg y$
$\psi(x) = y$	TP	FP
$\psi(x) = \neg y$	FN	TN

From the confusion matrix and the number of misclassified samples that has been counted, various measures of performances[16] can be computed.

In a context of target detection, the receiver operating characteristic (ROC) curve shown in Figure 15 illustrates the trade-off that the optimal design of a classifier may take into account. A Pareto front between the probability of detection, which corresponds to true positive (TP) and the rate of false alarms (FP), delineates the domain that both measures of performance may take on. A measure of performance often computed from the ROC curve is the area under the curve.

[16] True positive rate, false positive rate, false negative rate, true negative rate, accuracy, precision, positive predictive value, negative predictive value, Kappa index (Jousselme and Maupin, 2012d).

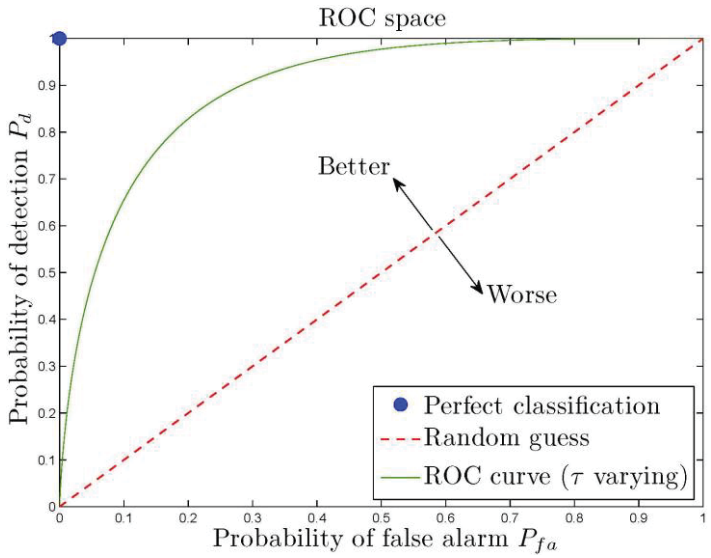

Figure 15. Classifier performance (Jousselme and Maupin, 2012d).

5.5 Classifier learning and testing

As shown in Figure 17, the pattern recognition system consists of the training module and the classification module (Jain *et al.*, 2010). Generating a classifier amounts to seek a mapping

$$\psi(\mathbf{z}^r_{m+1},.): \quad \mathcal{P}(S) \rightarrow \mathcal{Y}^n$$
$$\mathbf{x}^n_m \; \alpha \; \hat{\mathbf{y}}^n, \tag{15}$$

within a solution space, where in (15) $\mathcal{P}(S)$ and $\hat{\mathbf{y}}^n$ stand for the power set of S and the estimate of label set of network ω, respectively. The classifier model in (15) is generated by minimizing the estimate error $\hat{\mathbf{y}}^n - \mathbf{y}^n = \hat{\varepsilon}_r$ in Figure 16 and by defining decision boundaries within the feature space.

Error minimization thus involves a feedback loop from the training error $\hat{\mathbf{y}}^n - \mathbf{y}^n$ to the classifier mathematical representation, whose parameters are modified accordingly. Various simple and combined classifiers are tested in Chapter 6.

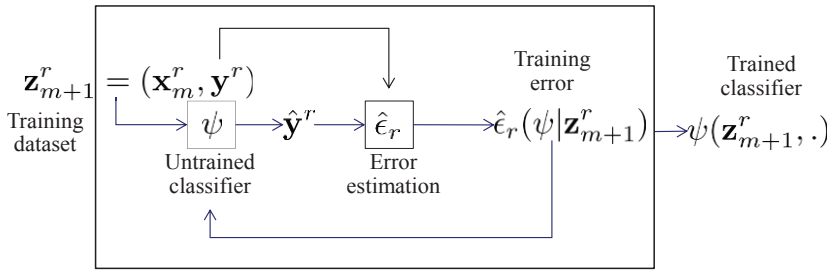

Figure 16. Training of the network vulnerability classifier (Jousselme and Maupin, 2012d).

It should be noted that, instrumental in providing the hard label vector $\hat{\mathbf{y}}^n$ [17], whether in the training or classifying phase shown in Figure 17, a score vector $g(\mathbf{z}^r_{m+1},.) : \mathbf{x}^n_m \alpha\ \bar{\mathbf{y}}^n$, considered as a soft label vector[18], is first computed. The score is then processed by a labelling function (e.g., the score is compared with a threshold), which yields the hard label vector $\hat{\mathbf{y}}^n$. The elements of $\hat{\mathbf{y}}^n$ represent the output of a mapping between the feature space shown in Figure 14 and the hard label set $\{V, \bar{V}\}$.

The feedback path in Figure 17 (dashed line) allows optimizing the training phase of a given classifier by selecting the most appropriate set of features and possibly by modifying the set of local representations of the networks. This last option is not presented in this report and is left for future investigations. When a classifier is trained, actual data feeds the classifier used for the purpose of vulnerability recognition, as shown in the lower part of Figure 17.

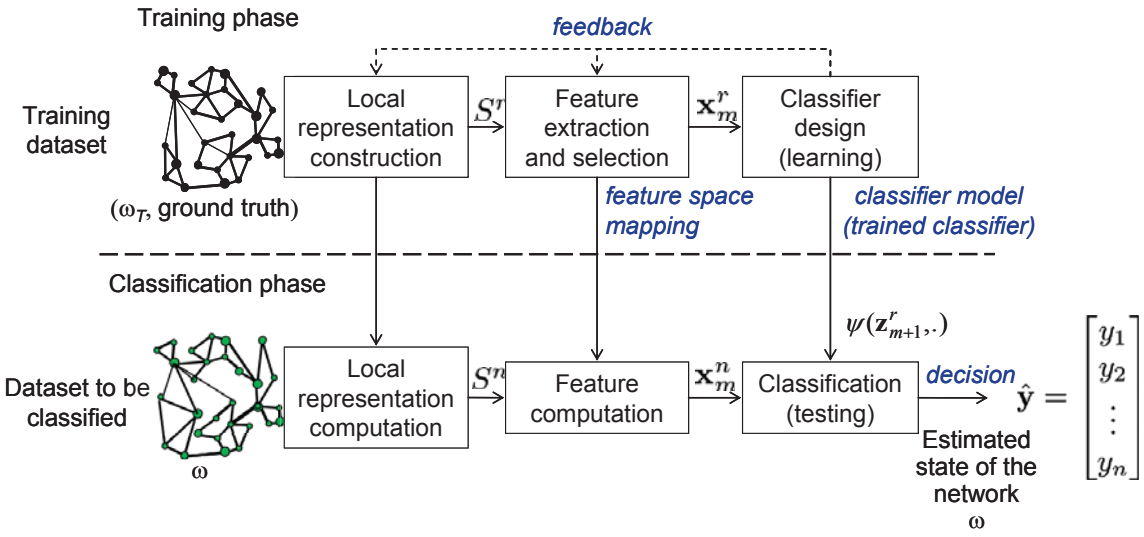

Figure 17: Vulnerability recognition systems (Adapted from Jain et al. (2010)).

The trained classifier can be interpreted as a vulnerability model of networks, whose sensitivity to triggering events is maximized. It should be noted that this model can be related to a Stackelberg-game-based model involving the definition of a risk function (Léchevin and Maupin, 2011c). Game-theoretic approach typically allows to identify vulnerabilities of moderately complex networks (Brown *et al.*, 2005).

When the classifier training is deemed satisfactory[19], the classification phase consists in applying the trained classifier $\psi(\mathbf{z}^r_{m+1},.)$ to an unlabelled dataset (i.e. only data set x^n_m is available) that has never been handled by the classifier at the training phase (Figure 18).

The set $\theta(\omega)$ of local representations $S_i \in S(\omega)$ that are considered as being vulnerable is obtained as follows :

---

[17] A hard label vector is a discrete-valued vector, whose entries belong to $\mathcal{Y}^n$.

[18] A score is the output of a discriminant function from $\mathcal{P}(S)$ to R or to the normalized interval [0, 1].

[19] Each selected measure of performance must satisfy a criterion of success; for instance, the area under the ROC curve must be greater than some prescribed threshold.

$$\theta(\omega) = \left\{ S_i \in S \,|\, \psi(\mathbf{z}^r_{m+1}, \mathbf{x}_j) = V \right\}, \tag{16}$$

where $\mathbf{x}_j = [x_1^j \ldots x_m^j]$.

In Figure 18, a network that does not belong to the training set feeds the classifier, which identifies its vulnerable subgraphs $S_i \in S(\omega)$. Following (16), $\theta(\omega)=\{S_1, S_2\}$.

A classification error estimate is computed based on the training phase.

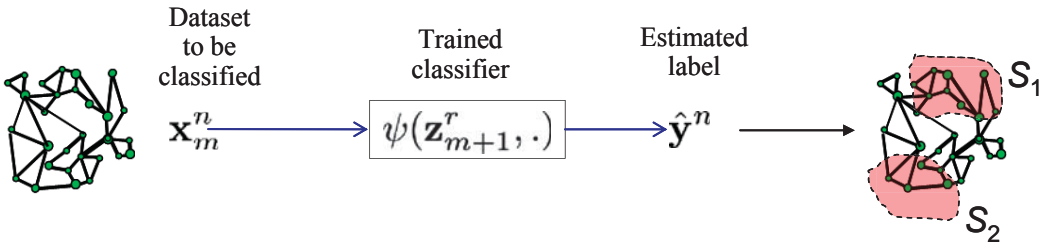

Figure 18: Classification phase.

It should be noted that the formulation of a multiclass problem may also be appropriate. The set of possible class labels is then defined by $\{V_1, V_2, ..., V_c; \overline{V}\}$, where V_i characterizes the type or degree of consequence associated with the vulnerability of S_i. Classes are defined by comparing each element of the output vector of the score function, $g(\mathbf{z}^r_{m+1}, .)$, not to a single threshold but to a set of ordered thresholds $\{s_1, s_2, \ldots, s_{c+1}\}$; that is, given $i \in \{1, \ldots, n\}$ and $j \in \{1, \ldots, c\}$, the ith element of the hard label estimate $\hat{\mathbf{y}}^n$ belongs to class V_j if the ith element of the score output $\overline{\mathbf{y}}^n$ (soft label) lies in $[s_j, s_{j+1}[$.

A combined classifier, using such aggregation functions as product, mean, median, minimum, maximum, and vote, can be optimized in the same way as a simple classifier is, by noticing that a combined classifier is built as follows. Let $\{\psi_1, \ldots, \psi_p\}$ be a set of p simple classifiers. Mean, average, median, minimum, and maximum functions, denoted A, are applied to the score function g_i, $i=\{1, \ldots, p\}$; that is, $A(g_1, \ldots, g_p)$: $(x_1, \ldots, x_p) \mapsto A((g_1(x_1), \ldots, (g_p(x_p))$. More precisely, the following definitions are used:

- Mean: $A(g_1, \ldots, g_p) = (g_1 + \ldots + g_p)/p$;
- Minimum: $A(g_1, \ldots, g_p) = \min(g_1, \ldots, g_p)$;
- Maximum: $A(g_1, \ldots, g_p) = \max(g_1, \ldots, g_p)$;
- Product: $A(g_1, \ldots, g_p) = g_1 \times \ldots \times g_p$;
- Median: $A(g_1, \ldots, g_p) = g_{(1+p)/2}$ if p is odd; $(g_{p/2} + g_{1+p/2})/2$, otherwise.

A vote function such as the majority vote (Kuncheva , (2004) uses the hard labels as input arguments. Letting l_i be the ith labelling function, then $A(l_1, \ldots, l_p)$: $(\overline{\mathbf{y}}^1, \ldots, \overline{\mathbf{y}}^p) \mapsto A((l_1(\overline{\mathbf{y}}^1), \ldots, (l_p(\overline{\mathbf{y}}^p))$, whereby the class entailing a majority of identical hard label values is selected.

The classifier design methodology aiming at generating the best classifier consists, as shown in

the experiments presented in Chapter 6, of evaluating the classification error of every simple classifier and combined simplifier. It should be noted that more sophisticated classification configuration will be investigated as future work to take into account modelling uncertainties and the automatic selection local representations (Jousselme and Maupin, 2012d). Possible extensions are discussed in Chapter 7.

5.6 Structural features

Four classes of features, based on structural, dynamical, functional and complexity properties of ω, were proposed in (Léchevin *et al.*, 2011a; Léchevin and Maupin, 2011c) to frame the vulnerability PR of networks. A tentative list, shown in Figure 19, consists of

- the structural features f_s of a labeled weighted graph (e.g., centrality, similarity, connectivity),
- the flow dynamics features f_d obtained by exploiting information on signals and systems (e.g., bifurcation analysis, efficiency measures (Latora and Marchiori, 2004; Qiang and Nagurney,2007)),
- features f_{cs} pertaining to complex system science and statistical physics (e.g., entropy, fractal dimension), and
- functional features f_f on key components of the network (e.g., based on expert's knowledge about the network).

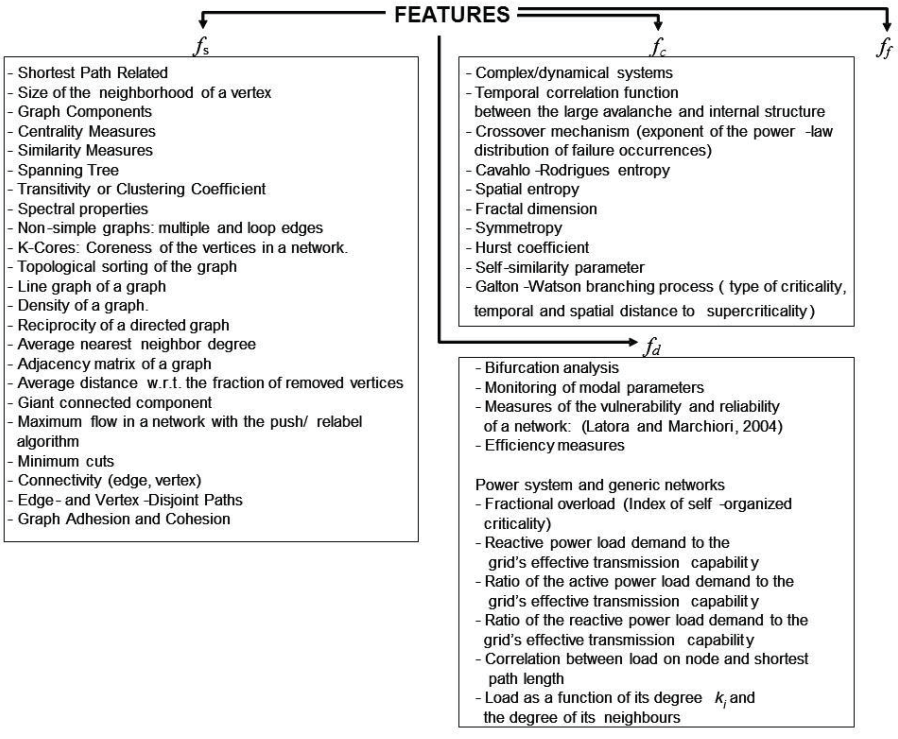

Figure 19: Categories of features.

In the remainder of the report, the pattern recognition of networks vulnerabilities in response to a class of threat is carried out by exploiting only the structural features. We restrict the set of features used by the classifier to a set of structural features for the following reasons.

- The structural features presented below characterise, to some extent, the topological properties of a network, independently of the type of infrastructure it represents, whereas flow dynamic features tend to depend on the physical properties of the system to be analyzed. Nevertheless, it should be noted that structural properties may also take into account time-invariant, physical attributes of nodes and edges such as transport capacity or throughput of edges.
- Efficient network analyzer toolboxes are widely available whereas features based on system dynamics properties and complex system metrics remain to be developed for the analysis of network vulnerabilities; therefore, it is natural to start the analysis of PREVU classifier-based approach using only structural features.
- Selecting structural features allows to assess the extent to which vulnerability analysis can be performed by comparing the classification rates obtained with various sets of features, with structural features serving as a reference.

However, the authors of this report are perfectly aware that improvements of classification performance are likely to be obtained by accounting for features that are not only structural.

The structural features are selected to characterize the local representation of a network, as suggested by the feature matrix $\mathbf{x}_m^r$ in Section 5.3. A structural feature that is specific to the whole network or to a type of local representation that is not part of the list of representations $\{S_1, S_2, \ldots\}$ cannot be directly exploited by the classifier since it cannot comply with the format of the feature matrix $\mathbf{x}_m^r$. Thus, the development of PREVU is started with structural properties that characterize only nodes and edges (no subgraphs are considered), which are the simplest local representations of a network. However, some of the proposed features could be applied to subgraphs such as clusters. Such an extension will be undertaken in future work. Features selected include such properties as degree distribution, various centrality measures, density measure, and clustering coefficient.

igraph (Csárdi and Nepusz (2006a, 2006b)) is an open source, free software[20], which proposes efficient routines for large networks analysis, generation and visualization. The following features are directly taken from igraph library and will thus be part of PREVU toolbox (see Csárdi and Nepusz (2006a, 2006b) and references therein for further details on network structural features):

- vertex betweenness: number of geodesics going through a vertex;
- degree: number of edges incident to a given node;
- closeness: number of steps required to access every other vertex from a given vertex;
- alpha centrality: solution of $x=\alpha A^T x+e$, where A is the adjacency matrix, e is a vector of exogenous sources, and α is the relative importance of the endogenous factors versus exogenous factors; (Csárdi and Nepusz, 2006a,b);
- eigenvector centrality: values corresponding to the components of the first eigenvector of

[20] igraph is developed as a Python extension module or as an R package.

the graph adjacency matrix;

- page rank (Brin and Page, 1998);
- average nearest neighbour degree (Barrat et al., 2004);
- graph strength: sum of the edge weights of the adjacent edges to a vertex;
- transitivity: ratio of the number of triangles[21] connected to the vertex to the number of triples[22] centered on the vertex, where the vertex is incident to both edges (Csárdi and Nepusz, 2006a,b); the transitivity is related to the local clustering coefficient (Wasserman and Fraust, 1994);
- Kleinberg's authority score: principal eigenvector of $A^{T}A$, where A is the graph's adjacency matrix;
- graph coreness, where the coreness of a vertex is k if it belongs to the k-core but not to the (k+1)-core, noting that the k-core of graph is a maximal subgraph in which each vertex has at least degree k; and
- average shortest path per node obtained from the length of all shortest paths between every pair of nodes of the network.

Considering the graph {{1,2,3,4,5,6,7}, {(1,2), (1,6), (2,3), (3,6), (3,5), (3,4), (4,5), (5,6), (6,7)}}, shown in Figure 20 (see its Laplacian matrix in Annex A.3), the structural features, which are computed for every node of the graph, result in a matrix whose rows, detailed below, are obtained using igraph (Csárd and Nepusz, 2006a, 2006b).

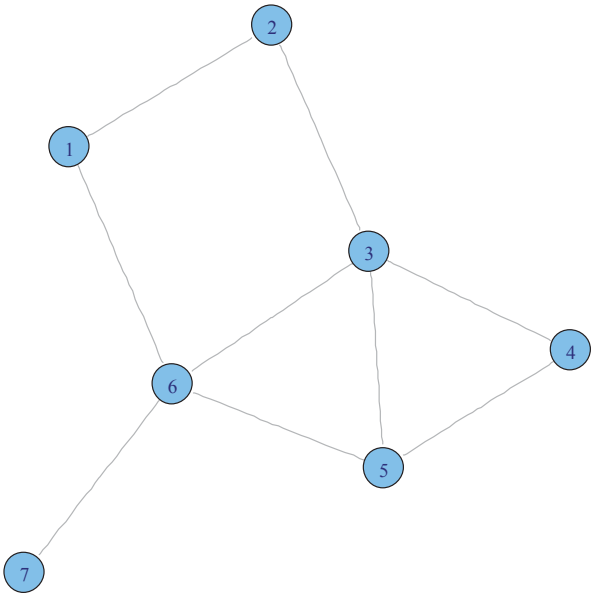

Figure 20: A five-node graph generated by igraph.

[21] A triangle is composed of three closed triplets, each of which is centered on a node of the triangle. A triplet is composed of three connected nodes. Connections involve either two edges or three edges. A three-edge triplet is termed closed triplet.
[22] A triple is a subgraph composed of three vertices and two edges.

1. Vertex betweenness: 1.0000, 0.8334, 4.6667, 0.000, 1.3334, 7.1667, 0.0000;

2. Node degree: 2, 2, 4, 2, 3, 4, 1;

3. Closeness: 0.0909, 0.0909, 0.1250, 0.0833, 0.1111, 0.1250, 0.0769;

4. Alpha centrality: 1, 2, 3, 4, 8, 13, 14;

5. Eigenvector centrality: 0.4842, 0.50670, 1.0000, 0.6390, 0.8709, 0.9107, 0.3110;

6. Page rank: 0.1171, 0.1157, 0.2096, 0.1112, 0.1597, 0.2187, 0.0679;

7. Average nearest neighbour degree: 3.0000, 3.0000, 2.7500, 3.5000, 3.3333, 2.5000, 4.0000;

8. Graph strength: 2, 2, 4, 2, 3, 4, 1;

9. Transitivity: 0.0000, 0.0000, 0.3334, 1.0000, 0.6667, 0.1667, NaN;

10. Graph coreness: 2, 2, 2, 2, 2, 2, 1;

11. Kleinberg's authority score: 0.4842, 0.5069, 1.0000, 0.6390, 0.8709, 0.9107, 0.3110;

12. Average shortest path per node: 1.6667, 1.0000, 1.5000, 1.1667, 1.1667, 1.8333;

Following the definition of transitivity, the transitivity of a node cannot be defined if this node is not centered on an open triple of nodes (three nodes connected). In Figure 20, node 7 is not incident to both edges; therefore, the transitivity is not defined for this node. Since transitivity is a positive scalar lying in [0, 1], setting by convention a value outside this interval when the transitivity of a node is not defined allows the classifier to disregard this value.

A submatrix of the feature matrix characterizing the 1000-node random network presented in Chapter 3 (fuse model) is given in Section 6.1.1.

5.7 System architecture

Every step of PREVU, from data generation to vulnerability analysis, can be performed seamlessly in Matlab (Mathworks, 2012), even if the graph analysis module necessitates using igraph (Csárdi and Nepusz, 2006a, 2006b), as illustrated in Figure 21. However, igraph can be started from Matlab, which can also retrieve the data resulting from igraph analysis (See Annex A for the igraph script used to analyze networks and the Matlab script that calls the igraph script). igraph is run as an R package[23]. R is a free software environment for statistical computing and graphics.

The main program, which is coded in Matlab, acts as a task sequencer from data generation to network analysis carried out by igraph, and to vulnerability labeling output by the classification module developed as a Matlab toolbox (Jousselme and Maupin, 2012d).

[23] http://www.r-project.org/ (Date visited: January 24, 2013)

The data generator provides a training dataset and a dataset used to test the classifier performance. These datasets embed the graph definition (list of nodes and edges, or a graph Laplacian matrix) and the responses[24] of the networks presented in Chapters 3 and 4 to the occurrence of disturbances (node removal). Random network generation and diffusion processes (avalanches, collective robot behaviour) are coded in Matlab. Other types of networks (power law, small word, actual data) and avalanche mechanisms (e.g. epidemics, gossip propagation) could be analyzed as long as the nomenclature of the graph data file (list of edges and list of labels), shown in Figure 21, is observed.

Data exchanges between Matlab and igraph involves two text files, namely, a graph data file and a graph feature file. The latter is the input of a Matlab pattern recognition toolbox (PR Lab System) whereby classifiers, whether simple or combined, feature selections and different metrics generates a label vector and various performance indicators. The label vector identifies the graph local representations that are possibly vulnerable.

A detailed description of the classifier generation system, which is implemented in Matlab and based on PRTools (2012), is given in Jousselme and Maupin (2012d). Up to eighteen simple classifiers are used for conducting the network vulnerability analysis experiment presented in the next chapter. Simple aggregation functions are also used to derive combined classifiers, as explained in Section 5.5 and Chapter 6. Further design options are left for future investigations. They include, as briefly presented in Chapter 7, feature filter, uncertainty modeller, and labeling functions. The search for classifiers in such a large decision space calls for optimization techniques particularly suited for high-dimensional problems.

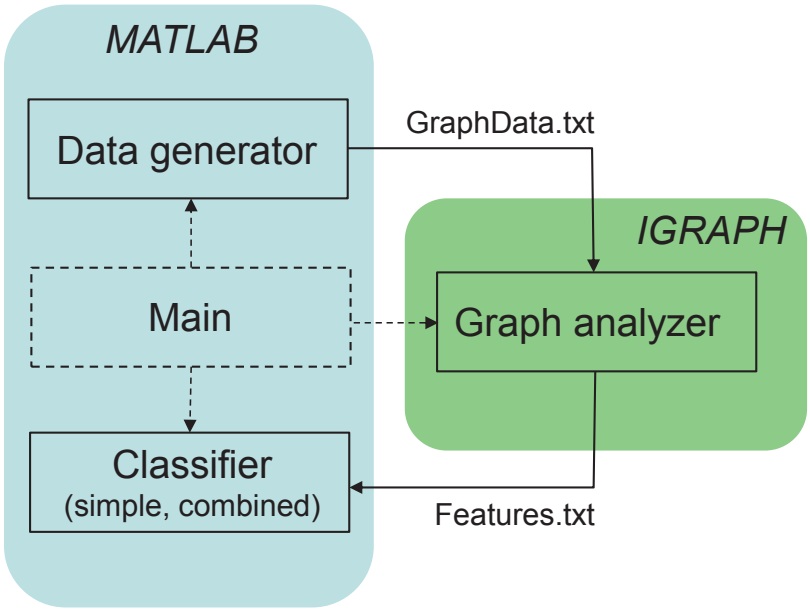

Figure 21: PREVU programming environment.

[24] Recall that the network response to a particular stressor is represented by an element of the hard label vector $\mathbf{y}^n$ that a classifier aims to estimate ($\hat{\mathbf{y}}^n$ in (15)) whenever the network operating condition differs from those characterizing the training dataset.

This page intentionally left blank

6 Pattern Recognition Experiments

6.1 Fuse model

6.1.1 Experiment description

A random network consisting of 1000 nodes, generated using the process presented in Chapter 3, is used to train and test the classifiers. To trigger potential avalanches, a single node of the network is removed. The edge removal mechanism, as given by equation (5), the jump condition and the stopping rule in Section 3.2, is then applied. The number of nodes and edges lost, which are instrumental in determining the label, are measured. This process is repeated for every single node removed from the network. A 1000×11 feature matrix and a 1000×2 label matrix (number of nodes and edges lost) are obtained for all $\mathscr{G}_{r,I}$ with $i \in \{1,\ldots,10\}$ and $r \in \mathscr{R}$, resulting in the labelled matrix of features shown in Figure 22, where feature and label values have to be multiplied by a factor of 10^3. From the soft labels provided in the matrix shown in Figure 22, hard labels (binary labels) are obtained by comparing the soft label with a threshold expressing a prescribed cost constraint.

The cost constraint expresses the maximum number of disabled edges and nodes that can be tolerated. This constraint is context specific and depends on the type of infrastructure. For instance, defining the quality of service delivered by an infrastructure includes a risk assessment whereby the acceptable/unacceptable domains of a risk graph (impact versus event probability) are determined (see, for instance, Hydro-Québec's risk graph in Trudel et al. (2005)). The boundaries of the acceptable/unacceptable domains thus constrain the cost[25] of a class of detrimental impacts to some value. This value serves as a threshold used to set the labels to one or zero. Since the fuse model does not represent any actual system such as a specific infrastructure or a network of infrastructures, the cost constraint has been set so that the occurrence of avalanches is not considered as being a rare event (i.e., an event with a probability less than 10^{-5}). In so doing, the frequency of avalanche occurrences allows to train and to evaluate the classifier with an acceptable level of confidence. It should be noted that, with actual networks, elaborate risk functions could be used in the decision-making process; see, for instance, McCalley et al. (1999) for risk functions derived to assess the security of bulk transmission networks.

Features											Soft labels	
0.5679	0.0050	0.0001	0.0012	0.0000	0.0000	0.0026	0.0002	0.0007	0.0040	0.0000	0.0310	0.0870
4.1842	0.0050	0.0001	0.0013	0.0000	0.0000	0.0028	0.0002	0.0006	0.0040	0.0000	0.0590	0.1390
1.1081	0.0050	0.0001	0.0012	-0.0000	0.0000	0.0049	0.0001	0.0005	0.0040	0.0000	0.0400	0.1170
1.2845	0.0130	0.0000	0.0011	0.0000	0.0000	0.0171	0.0001	0.0006	0.0060	-0.0000	0.0620	0.1940
0.9496	0.0080	0.0001	0.0015	-0.0000	0.0000	0.0054	0.0004	0.0007	0.0060	0.0000	0.0590	0.1380
1.2053	0.0060	0.0000	0.0011	0.0000	0.0000	0.0132	0.0000	0.0007	0.0040	-0.0000	0.0370	0.0920
9.7415	0.0130	0.0001	0.0013	-0.0000	0.0000	0.0130	0.0002	0.0007	0.0070	0.0000	0.0300	0.0460

Figure 22. Sample of the 1000×11 labelled matrix of features.

[25] The cost is often expressed as an economic cost.

Simple and combined classifiers, coded as Matlab scripts (see Figure 21) and available in PRtools (PRTools, 2012), are considered to map the feature domain to labels. Simple classifiers consist of (Duin et al., 2007):

- normal densities-based linear classifier (ldc);
- normal densities-based quadratic classifier (qdc);
- normal densities-based classifier with independent features (udc);
- nearest mean classifier (nmc);
- scaled nearest mean classifier (nmsc);
- minimum least square linear classifier (fisherc);
- linear perceptron-based classifier (perlc);
- quadratic classifier (quadr);
- linear classifier by Karhunen-Loeve expansion of the common covariance matrix (klldc);
- logistic linear classifier (loglc);
- An arbitrary classifier is run with polynomial features (polyc);
- Parzen classifier; and
- k-nearest neighbour classifier (k=1, 3, 4, 5, 7, 9).

The classifier combination functions, instrumental in generating combined classifiers, consist of:

- product combining classifier (product);
- averaging combining classifier (mean);
- median combining classifier (median);
- minimum combining classifier (minimum);
- maximum combining classifier (maximum); and
- vote combining classifier (vote).

Given p simple, trained classifiers, $\{\psi_1,\ldots, \psi_p\}$, a combined classifier results from the application of one of the six aforementioned combination function to $\{\psi_1,\ldots, \psi_p\}$, as explained in Section 5.5.

Following Figure 17 and Figure 21, a classifier, whether simple or combined, receives as input the network feature matrix illustrated in Figure 22 and outputs a label set, which can be used to assess the classifier performance. The training of a combined classifier is performed similarly to that of a simple classifier by minimizing its classification error.

To assess the performance of a classifier, whether it is simple or combined, an error estimate is computed. The error estimate is a deviation metric between actual vulnerability labels obtained from avalanche simulations and vulnerability label estimates resulting from the application of

each type of classifiers. The error estimate is obtained by means of cross validation[26] using three folds and ten repetitions.

6.1.2 Results

The best recognition rate (68%) is obtained with classifiers ldc, fisher, klldc and polyc (see the four dark blue bars in Figure 23). This rate means that the predicted vulnerability label vector corresponds, up to a proportion of 68%, to the actual vulnerability vector obtained by simulation; that is, up to 68% of the network local representations are identified with the correct labeling.

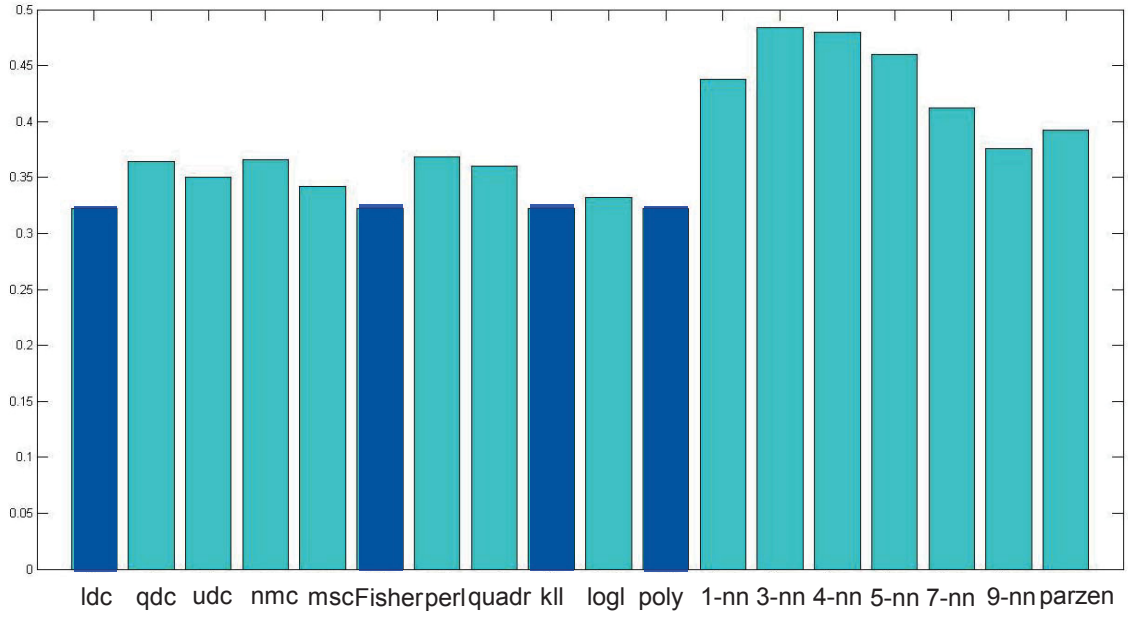

Figure 23. Misclassification rate computed for 18 classifiers. The best four classifiers are coloured in dark blue (ldc, fisher, klldc, polyc).

It is believed that the combination of several classifiers could result in better classification rates that those obtained with individual classifiers. The combination functions in Section 6.1.1 (see Section 5.5 for a formal description) are applied to the best six classifiers (*ldc, fisher, klldc, polyc, logl, msc*) in Figure 23.

The best recognition rate (71%) is obtained with the product function, as shown in Figure 24.

[26] Let Z be a dataset containing n-dimensional features describing N objects. Z is divided into K subsets of dimensions N/K. One subset is used to test the performance of the classifier trained on the union of the K-1 other subsets. This procedure is repeated K times; the test subset is changed every time (Kuncheva, 2004).

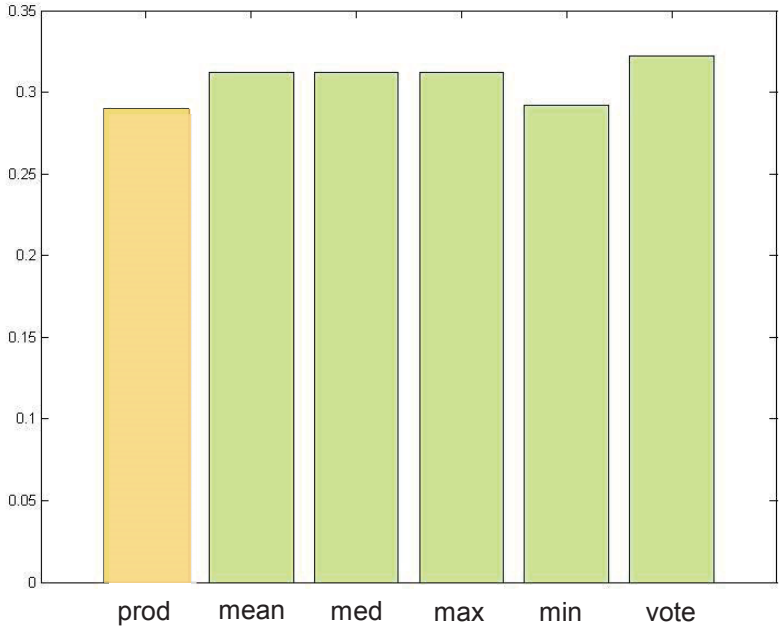

Figure 24. Misclassification rates obtained by combining the best six classifiers in (ldc, ficherc, klldc, polyc, loglc, msc).

The relevance of the structural features presented in Section 5.6 is obtained by computing the Fisher criterion (Duda et al., 2000). The best four features among those listed in Section 5.6 are, in order of decreasing Fisher criterion values, closeness, transitivity, graph strength, and vertex betweenness (Figure 25).

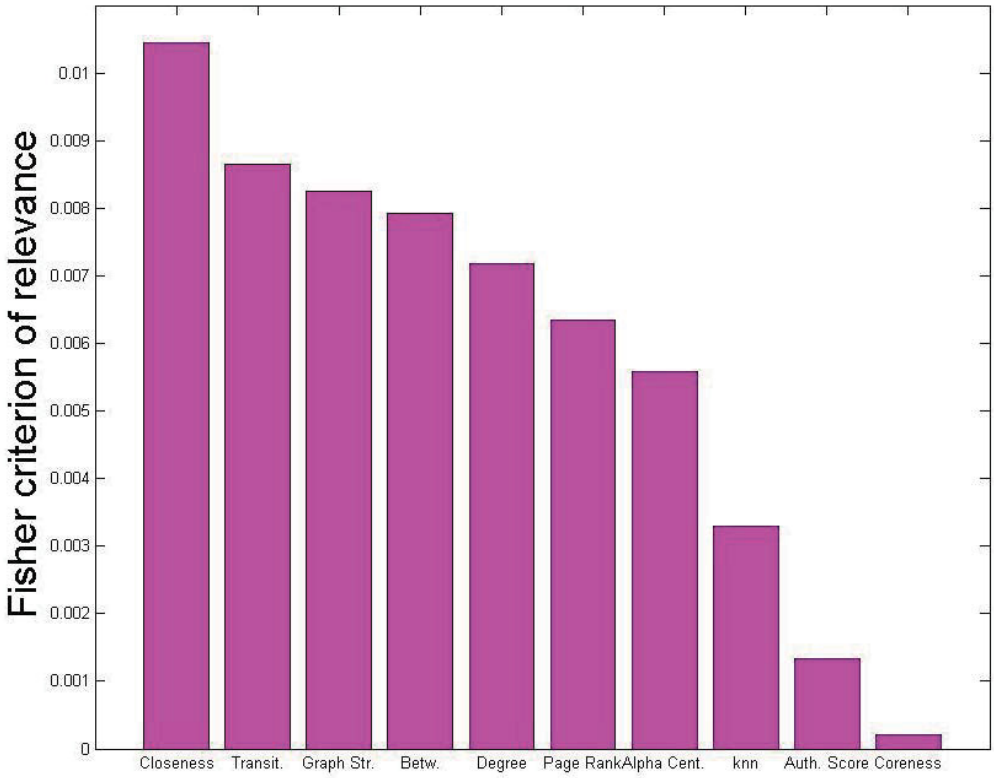

Figure 25. Relevance of structural features that are part of the classifier.

6.2 Tactical swarm

6.2.1 Experiment description

Given an initial clients configuration C_0 and a corresponding robot deployment G_0, (including the robots' coordinates $N_{c,0}$), the training data set z used to derive the classifier in (6), is obtained from the disturbance sample set $\mathcal{D}=\{D_1,\ldots,D_n\}$. D_i stands for a set of sequence of disturbances $\{\delta_{ij1}, \delta_{ij2},\ldots,\delta_{ijm}\}$. This sequence is in a one-to-one correspondence with the occurrence time set $\{t_1, t_2, \ldots, t_m\}$, $t_{i+1}>t_i$. δ_{ijk} represents a random realization of the j^{th} disturbance of the sequence applied to the network of robots at time t_k and for experiment i. The first disturbance δ_{ij0} corresponds to a robot failure and is followed by a series of new client configuration C_j, $\{\delta_{ij1},\ldots,\delta_{ijm}\}$.

It assumed that the first triggering event (first disturbance) δ_{ij1} is a robot failure occurring at t_1 and that all δ_{ij1} span R, the set of robots. The following disturbances are a series of new client configuration C_j, $\{\delta_{ij1},\ldots,\delta_{ijm}\}$ representing new client configurations arising from clients adjusting their positions. In order to evaluate the vulnerability of each robot, they will be removed successively. D_i is then defined as the following union $\cup_{j\in R}\{\delta_{ij1}, \delta_{ij2},\ldots, \delta_{ijm}\}$. D_i is

instrumental in defining experiment i. Indeed, sequence D_i generates the sequence of edge sets $\{\{E_{11},\ldots, E_{1m}\},\ldots,\{E_{|R|1},\ldots, E_{|R|m}\}\}$, where $|R|$ denotes the total number of robots.

The state of G_m is used to determine the cost $\mathscr{C}_{ij}(con_i,cov_i)$ associated with a disturbance sequence j of experiment i. This cost depends on the connectedness of the graph through a disconnectedness index, $con_i(G_m)$, derived from Equation (10), and the coverage $cov_i(G_m,,C_m)$ of the set of clients by the robot network at time t_m. In the binary case, when G_m is disconnected, con_i is equal to 1 and when G_m is connected, con_i is set to zero. The binary (see Equation (10) in Section 4.3.2) case is implemented.

Sequence j of experiment i is thus associated with the following mapping

$$\{d_{ij1},d_{ij2},\ldots,d_{ijm}\} \rightarrow \mathscr{C}_{ij} \rightarrow y_{ij}$$

Each experiment i is characterized by the set of labels $\{y_{i1},\ldots,y_{i|R|}\}$.

6.2.2 Results

A classifier is trained to recognize the vulnerable components of a network and thus predict its vulnerabilities, should a potential set of triggering events occur. Each node of a network is labeled as vulnerable or non-vulnerable. To obtain the data pertinent to the classifier training, various simulation runs are performed. Starting with a swarm configuration that initially corresponds to an equilibrium state, as shown in Figure 6, a node is labelled as vulnerable if its removal affects the network connectivity by virtue of the adaptation mechanism presented in Chapter 4. A swarm of 100 robots is considered, a number which is beyond current autonomous robotics practical applications. The simulation runs are initiated from a series of 100 swarms of robots, which are randomly deployed and designed to cover a set of 100 clients whose positions are also randomly generated. Figure 5 shows such a network.

Then, similarly as in Section 6.1.1, features are extracted for each node, leading to a matrix of features $\mathbf{x}_m^r$ as described in Section 5.6. Besides the structural features, a coverage feature, which is defined as the number of clients covered by each robot, is computed. Furthermore, a vector of labels is built where each component corresponds to a node of the network. The value of a component is determined by the connectivity index expressed in (10).

We obtained a recognition rate of 69,7% with a nearest mean classifier over a cross-validation error estimation with 5 folds and 10 repetitions. These preliminary results show that we are able to predict vulnerabilties within the porposed tactical swarms with a success rate less than 70%.

Selecting a limited number of features, indeed up to 11 features, may allow to implement the vulnerability analysis in real time by distributing the classification process over the group of robots. However, the real-time, distributed computation of several features such as the edge betweenness or the eigenvector centrality is not straightforward since such features require global knowledge of the network, which in turn may call for more memory and possibly more computation time as suggested by Table 2, where n, N, and M stand for the number of vertices to calculate, the number of vertices in the graph, and the number of edges in the graph, respectively.

Table 2: Some structural features and their computational complexity.

Features	**Space**	**Time**	**Required information**
Vertex betweenness	O(M)	O($M\,N$)	Global
Degree	O(n)	O(n)	Local
Eigenvector centrality	O(M)	O(M)	Global
Kleinberg's authority score	O(M)	O(M)	Global
Average shortest path per node	O(M^2)	O($M\,N \log N+M$)	Global

This page intentionally left blank

7 Conclusion

7.1 Summary of current capability

Establishing a mapping linking a limited set of structural features of a graph to a set of vulnerability labels is deemed central to building a model that is expected to yield fast results for the analysis and prediction of large network vulnerabilities. Based on a Matlab pattern recognition toolbox, this approach consists in training a classifier or a set of classifiers using datasets obtained from networks that may experience cascading failures when affected by the loss of links or nodes. Two networks are considered in this document, namely, a fuse model and a tactical robot cloud.

It has been shown that the proposed approach is able to identify the vulnerabilities of a network of a prescribed class[27] with a classification success up to 70%. Although this result may not be as accurate as that obtained with model-based simulations, the computation of a vulnerability label vector is very fast when the classifier is appropriately trained.

The classifier-based approach presented in this document is general enough to be applied to any types of networks. It is believed, however, that the efficiency of the approach may depend on the class of networks, avalanche mechanisms considered and of triggering event, explaining why the results reported in this document are preliminary.

7.2 Way ahead

The preliminary results presented in this document suggest further improvements and experiments.

First, possible ways to increase the classification rate include accounting for (*i*) extended classifier design options, for (*ii*) extended network features, and for (*iii*) more general network local representations.

Classifier could be improved by combining, as shown in Figure 26, various design options (Jousselme and Maupin, 2012d):

- feature filter *f*, which selects a subset of the initial feature set;
- uncertainty modeller *g*, which provides posterior probabilities, fuzzy sets and belief functions, to name a few;
- labelling functions *l*, which outputs labels based on scores, formal uncertainties, or hard labels.

[27] Recall that a class of networks refers to a specific type of graph whose node or edge attributes evolve with time following some dynamics: (*i*) a static random geometric graph serves as a layer in Chapter 3 for the proposed fuse model of avalanche, whereas (*ii*) the robot swarm in Chapter 4 is a mobile network whose dynamics results from the neighbour-based motion strategy implemented in every robot and from the location of each client.

These options along with the combination functions ρ presented in Chapters 5 and 6 can be considered as part of the decision variables of the classifier design optimization problem since the resulting classifier ψ can be expressed, in its most general form, as the following composition $\psi=\rho \circ l \circ h \circ g \circ f$, where g stands for classification mapping, whose input and output are a feature subset and a score vector, respectively. The application of genetic algorithm is considered to automate the search for a near-optimal solution to this classifier design (Jousselme and Maupin, 2012d).

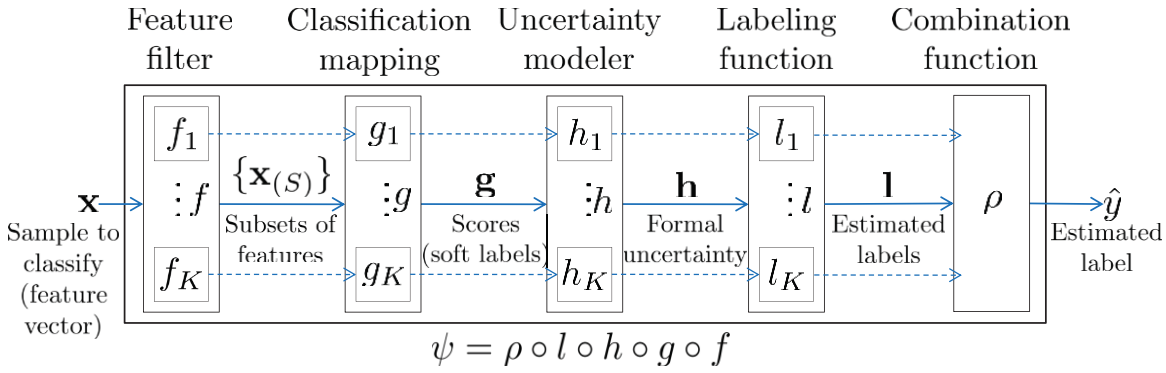

Figure 26. Multiple classifier system (Jousselme and Maupin, 2012d).

Design options related to network representations and properties consist of:

- optimizing PREVU classifier training by selecting a subset of local representations over the set of all local representations of a network, which include some of its subgraphs such as cliques, clusters, and communities; and
- integrating other types of features, especially those based on signal analysis using various transforms and on the dynamical content of a network.

Second, application of the proposed approach to other types of graphs and avalanche mechanisms should be undertaken to assess PREVU performances as a function of graph topological class and diffusion process class (e.g., cascading failures, epidemics). Conversely, it would be interesting to build a taxonomy of graph topology and avalanche mechanisms as a function of the classification efficiency.

Third, the class of initiating event considered in this report consists in removing a single node from the graph, therefore restricting the vulnerability analysis to the identification of a single point of failure. However, removing a limited subset of nodes may significantly increase the cost of network malfunctioning beyond linear extrapolations as a result of couplings inherent to networks. It is thus desirable to extend the vulnerability analysis of PREVU to triggering events that could directly affect subgraphs.

Last, it is expected in 2013 that PREVU toolbox be tested with actual data such as that obtained with the tactical mobile cloud as part of collaboration with West Point US military academy. Ultimately, it is expected that the classifier be used in closed loop with the motion strategy to improve the overall performance of the tactical mobile cloud. Furthermore, the robustness of PREVU to parametric uncertainties that may affect the network generation and the avalanche mechanism is of interest.

References

Alderson, D., and Doyle, J. (2010), Contrasting views of complexity and their implications for network-centric infrastructures, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 40, No. 4, pp. 839-852.

Bakke, J.O.H., Hansen, A., and Kertsész, J. (2006), Failures and avalanche in complex networks, *Europhysics Letters*, Vol. 76, No. 4, pp. 717-723.

A. Barrat, M. Barthélemy, R. Pastor-Satorras, and A. Vespignani (2004), The architecture of complex weighted networks, *Proceeding of the National Academy of Sciences of the United States of America*, vol. 101 no. 11 3747-3752, March 16
http://www.pnas.org/content/101/11/3747.full.pdf+html (Date visited: August 10, 2012)

Batroumi, G.G., and Hansen, A. (1998), Fourier acceleration of iterative processes disordered systems, *Journal of Statistical Physics,* Vol. 52, No 3-4, pp. 747-773.

Barabási, A.-L., Architecture of complexity (2007), *IEEE Control Systems Magazine*, Vol. 27, No. 4, pp. 33-42.

Bezzo, N and Fierro, R. (2011), Swarming of Mobile Router Networks, *2011 American Control Conference*, San Francisco, CA.

Bozek, T (2002), DoD critical infrastructure protection, In *Proceedings of the 18th NDIA Security Division Symposium and Exhibition*.

Brin, S., and Page, L. 1998, The Anatomy of a Large-Scale Hypertextual Web Search Engine, *Proceedings of the 7th World-Wide Web Conference*, Brisbane, Australia.

Brown, G.G, Carlyle, W.M., and Salmeron, J., and Wood, K. (2005), Analyzing the vulnerability of critical infrastructure to attack and planning defenses, *Tutorial in Operations Research*, Informs, pp. 102-123.

Bullo, F., Cortés, J., and Martinez, S. (2009), Distributed control of robotic network: a mathematical approach to motion coordination algorithms, Applied Mathematics Series, Princeton University Press, http://www.coordinationbook.info/pdfs/DCRN-BulloCortesMartinez-10mar09.pdf (Access date, 12 September 2012).

Caldarelli, G. (2007), Scale-free networks – Complex webs in natural, technological and social sciences, Oxford University Press.

CIAO (2003), Critical Infrastructure Assurance Office, Presidential Decision Directive 63, http://www.ciao.gov

Csárdi, G., Nepusz, T. (2006a), The igraph software package for complex network research, *InterJournal Complex Systems*, 1695.

Csárdim, G., and Nepusz T. (2006b), igraph Reference Manual, http:// igraph.sourceforge.net/doc/igraph-docs.pdf (Date visited: March 21, 2012)

Dobson, I., Carreras, B.A., and Newman, D.E. (2005), A loading-dependent model of probabilistic cascading failure, *Probability in Engineering and Informational Sciences*, Vol. 19, pp. 15-32.

Dobson, I., Kim, J., and Wierzbicki, K.R. (2010), Testing branching process estimators of cascading failure with data from a simulation of transmission line outages, *Risk Analysis*, Vol. 40, No. 3, pp. 650-662.

Dorogovtsev, S.N., and Mendes, J.F.F. (2004), Evolution of networks- From biological nets to the internet and WWW, Oxford University Press.

Duda, R.O., Hart, P.E., Stork, D.G. (2001), Pattern recognition, Toronto: Wiley.

Duin, R.P.W., Juszczak, P., Paclik, P., Pekalska, D., de Ridder, Tax, D.M.J, and Versakov, S. (2007), A Matlab toolbox for pattern recognition, Delft University of Technology.

Freeman, L.C. (1979), Centrality in social networks – Conceptual clarification, *Social networks*, pp. 215-239.

Godsile, C., and Royle, G. (2001), Algebraic graph theory, New York: Springer.

Hansen, A. (2005), Physics and fracture, *IEEE Computing in Science and Engineering*, Vol. 7, No. 5, pp. 90-95.

Heydt, G.T., Liu, C.C., Phadke, A.G., and Vittal, V. (2001), Solutions for the crisis in electric power supply, *IEEE Computer Application in Power*, Vol. 14, No. 3, pp. 22-30.

Hines, P., Cotilla-Sanchez, E., and Blumsack, S., (2010), Do topological models provide good information about vulnerability in electric power networks, *Chaos*, Vol. 20, No. 3.

IAEA, International Atomic Energy Agency, (2007), IAEA Safety Glossary, 2007 Edition, http://www-ns.iaea.org/standards/safety-glossary.htm (Date visited: March 23, 2010).

Jain, A.K., Robert, P.W., and Mao, J. (2010), Statistical Pattern Recognition: A Review, *IEEE Transactions on Pattern Recognition and Machine Intelligence*, Vol. 22, No. 1.

Jousselme, A.-L., Huggins, K., Léchevin, N., Maupin, P., and Larkin, D., (2012a), Vulnerability-aware architecture for a tactical, mobile cloud, *3rd Workshop on Complex Networks* (CompleNet), Melbourne, FL.

Jousselme, A.-L., Huggins, K., Maupin, P., Léchevin, N., and Larkin, D., (2012b), Surveillance coverage and vulnerability awareness concepts for tactical swarm, *IST-112 Symposium on IST-112/SET-183, Joint Symposium on Persistent Surveillance: Networks, Sensors, Architecture*, NATO, Québec, Canada.

Jousselme, A.-L., and Maupin, P. (2012c), General models for performance representation and surveillance coverage, Defence R&D Canada – Valcartier, Technical Note.

Jousselme, A.-L., Maupin, P. (2012d), Pattern Recognition System Configurations for decision under uncertainty – Some illustrative results, Technical Report DRDC- Valcartier.

Kuncheva L.I. (2004), Combining pattern classifiers – Methods and algorithms, Wiley-Interscience.

Latora, V., and Marchiori, M. (2004), "How he science of complex networks can help developing strategies against terrorisms," *Chaos, Solitons, and Fractals*, No. 20, pp. 69-75.

Léchevin, N., and Maupin, P. (2009), Vulnerabilities In Very Large Socio-Technical Networks – State of the art and proposed approach toward crisis prediction – Modeling and preliminary results, Technical Memorandum, DRDC Valcartier TM 2009-303.

Léchevin, N., and Maupin, P. (2011), Integration of early warning systems into protection planning for critical infrastructures, Technical Memorandum, DRDC Valcartier TM 2011-204.

Léchevin, N, Jousselme, A.L., and Maupin, P. (2011a), Pattern Recognition Framework for the Prediction of Network Vulnerabilities, *IEEE Network Science Workshop*, West Point, NY.

Léchevin, N., Rabbath, C.A., and Maupin, P. (2011b), Toward a stability monitoring system of an asset-communications network exposed to malicious attacks, *American Control Conference*, San Francisco.

Léchevin, N., and Maupin, P. (2011c), Integration of early warning systems to the protection planning of critical infrastructures, Technical Report, DRDC Valcartier TR 2011-204.

Mathworks (2012), http://www.mathworks.com/

McCalley, J.D., Vittal, V., and Abi-Samra, N., An overview of risk-based assessment, *Power Engineering Society Meeting*, Vol. 1, pp. 173-178, 1999.

Motter, A.E. and Lai, Y.-C. (2002), Cascade-based attacks on complex networks, *Physical Review E*, Vol. 66, No. 6, pp. 065102-1(4).

Perrow, C. (1984), Normal accidents: living with high-risk technologies, New York: Basic Books.

Penrose, M. (2003), Random geometric graphs, Oxford University Press.

President's Commission on Critical Infrastructure Protection (1997), Critical Foundations: Protecting America's Infrastructures, http://www.ciao.gov (Date visited: March 10, 2010)

PRTools (2012), http://www.prtools.org/, (Date visited: November 5, 2012).

Qiang, Q. and Nagurney, A. (2007), "A unified network performance measure with importance identification and the ranking of network components," *Optimization Letter*, Vol. 2, No. 1, pp. 127-142.

Rinaldi, S.M., Peerenboom, J.P., and Kelly, T.K. (2001), Identifying, understanding, and analyzing critical infrastructure interdependencies, *IEEE Control Systems Magazine*, Vol. 21, No. 6, pp. 11-25.

Simonsen, I., Buzna, L., Peters, K., Bornholdt S., and Helbibing, D., (2008), Transient dynamics increasing network vulnerability to cascading failures, *Physical Review Letters*, Vol. 100, No. 21, pp. 218701-1(4).

Svendsen, N.K., and Wolthusen, S.D. (2007), Analysis and statistical properties of critical infrastructures interdependency multiflow models, In *Proceedings of the 2007 IEEE Workshop on Information Assurance,* United States Military Academy, West Point, NY.

Trudel, G., Gingras, J.-P. and Pierre, J.-R. (2005), "Designing a reliable power system: Hydro-Québec'sintegrated approach," *Proceedings of the IEEE*, Vol. 93, No. 5, pp. 907-917.

Wang, J.-W., and Rong, L.L. (2009), Cascade-based attack vulnerability on the US power grid, *Safety Science*, Vol. 47, No. 10, pp. 1332-1336.

Wasserman, S., and Fraust, K. (1994), Social network analysis: Theory and applications, Cambridge: Cambridge University Press.

Wildberger, A.M. (1998), Complex adaptive systems: concepts and power industry applications, *IEEE Control System Magazine*, Vol. 17, No. 6, pp. 77-88.

Annex A Structural analzer

A.1 Matlab script used to call igraph routine

In this section, the Matlab script that manages the network structural analysis performed with igraph is provided. A graph Laplacian matrix (see Section A.3 for an example of Laplacian matrix) and a vector of edge capacities are provided as input arguments to the Matlab function `StructuralProperties_Ava`. A text file containing a $|\mathscr{E}|\times 2$ matrix[28] whose rows are the graph edges (couples of nodes) is generated from the Laplacian matrix. This text file is then read by the igraph routine presented in Section A.2. This igraph routine, which is called by `StructuralProperties_Ava`, provides a text file for each structural property analyzed. `StructuralProperties_Ava` then generates a feature matrix that stacks row vectors each of which is contained in an igraph-generated text file.

```
function[MatFea,MatFea_Edge]=StructuralProperties_Ava(TrueLaplaci
anMat,CapacityOut)

% This function derives the structural properties of the graph
% associated with TrueLaplacianMat by calling igraph code.
% Data is saved in txt files.

% TrueLaplacianMat: Network Laplacian matrix
% Capacityout     : Vector of edge capacity


% A list of edges is built from the network Laplacian matrix
% TrueLaplacianMat

[rTL,cTL]=size(TrueLaplacianMat);
i=1;
for u=1:rTL-1
    for v=u+1:cTL
        if TrueLaplacianMat(u,v)~=0
            Edges(i,:)=[u v];
        i=i+1;
        end
    end
end


% Create a file with edge list when a Laplacian Matrix is given
f=fopen('EdgesMatIgraphNet.txt','w');
fprintf(f,'%d %d\r\n',Edges');    % List of edges
fclose(f);
```

[28] $|\mathscr{E}|$ is the number of edges of a graph $\mathscr{G}=(\mathscr{N},\mathscr{E})$.

```
% Create a file with edge attribute (edge capacity)
capa=CapacityOut(:,3);
save EdgesAttributesNet.txt capa -ASCII

capaAbsVal=abs(capa);
save EdgesAttributesValAbsNet.txt capaAbsVal -ASCII


% Run igraph code in StructuralFeaturesPR and save the resulting
% matrix, which is obtained for every structural feature, in a
% txt file.

system('"C:\Program Files\R\R-2.11.1\bin\R" CMD BATCH
"C:\NL\TIF_PREVU\SIM\Test\Avalanche_1\StructuralFeaturesPR"
ResultRMatlab.txt')

load betweenness_ava.txt
load degree_ava.txt
load closeness_ava.txt
load alpha_centrality_ava.txt
load eigenvector_centrality_ava.txt
load page_rank_ava.txt
load graph_knn_ava.txt
load graph_strength_ava.txt
load transitivity_ava.txt
load graph_coreness_ava.txt
load authority_score_ava.txt
load shortest_path_node_ava.txt
load edge_betweenness_ava.txt

% NaN in transitivity_ava are replaced by 0

transitivity_ava_aux=transitivity_ava;
iii=find(isnan(transitivity_ava)==1);
transitivity_ava_aux(iii)=0;
transitivity_ava=transitivity_ava_aux;

% Create an overall feature matrix from the matrices in txt
files; First row is deleted
[ro,co]=size(betweenness_ava);

MatFea=[betweenness_ava(2:ro,1) degree_ava(2:ro,1) ...
closeness_ava(2:ro,1) alpha_centrality_ava(2:ro,1) ...
eigenvector_centrality_ava(2:ro,1) page_rank_ava(2:ro,1)...
graph_knn_ava(2:ro,1) graph_strength_ava(2:ro,1) ...
transitivity_ava(2:ro,1) graph_coreness_ava(2:ro,1) ...
authority_score_ava(2:ro,1)];
```

```
[roE,coE]=size(edge_betweenness_ava);
MatFea_Edge=[edge_betweenness_ava(1:roE)];
```

A.2 igraph routine

The igraph routine that provides the structural feature is presented next. The graph representation and edge attributes are obtained from `EdgesMatIgraphNet` text file as a $|\mathscr{E}|\times 2$ matrix and from `EdgesAttributesNet` text file as a $|\mathscr{E}|$-dimensional vector, respectively. Each structural feature vector is saved in a corresponding text file; for instance, the between vector is saved in `betweenness_ava`. A feature matrix is output by Matlab function `StructuralProperties_Ava` when all the feature text files have been read.

```
# Structural properties of the electrical network

library(igraph)

# Load the graph file

g <- read.graph("EdgesMatIgraphNet.txt","edgelist",
directed=FALSE)
edge_attrib <- read.table("EdgesAttributesNet.txt",
col.names=c("Attrib1"))

E(g)$attrib1 <- edge_attrib$Attrib1
summary(g)
plot(g)

exo<-
c(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)

# Edge attribute list
edge_attrib2 <- read.table("EdgesAttributesValAbsNet.txt",
col.names=c("Attrib2")) #
read.table("EdgesAttributesValAbsNet50_3.txt",
col.names=c("Attrib2"))
E(g)$attrib2 <- edge_attrib2$Attrib2
E(g)$weight <- edge_attrib2$Attrib2
summary(g)

# 1] Local (representation) structural properties: Nodes

bet<-betweenness(g)
write(bet, 'betweenness_ava.txt',ncolumns=1)
remove(bet)

deg<-degree(g)
write(deg, 'degree_ava.txt',ncolumns=1)
```

```
remove(deg)

clo<-closeness(g)
write(clo, 'closeness_ava.txt',ncolumns=1)
remove(clo)

alcentrality<-alpha.centrality(g,alpha=1) #,exo)
write(alcentrality, 'alpha_centrality_ava.txt',ncolumns=1)
remove(alcentrality)

eigcentral<-evcent(g)
# Eigenvector Centrality Scores of Network
write(eigcentral$vector, 'eigenvector_centrality_ava.txt',
ncolumns=1)
remove(eigcentral)

parank<-page.rank(g)
write(parank$vector, 'page_rank_ava.txt',ncolumns=1)
remove(parank)

grknn<-graph.knn(g)
write(grknn$knn, 'graph_knn_ava.txt',ncolumns=1)
remove(grknn)

G_strength<-graph.strength(g)    # Edge weights are needed
write(G_strength, 'graph_strength_ava.txt',ncolumns=1)
remove(grknn)

trans<-transitivity(g,"local")
write(trans, 'transitivity_ava.txt',ncolumns=1)
remove(trans)

coren<-graph.coreness(g)
write(coren, 'graph_coreness_ava.txt',ncolumns=1)
remove(coren)

authsco<-authority.score(g)
write(authsco$vector, 'authority_score_ava.txt',ncolumns=1)
remove(authsco)

# Computation of the average shortest path for each node: sp_node
sp<-shortest.paths(g)
dimen<-dim(sp)
sp_node<-numeric()
for (i in 2:dimen[1]){
sp_node[i-1]<-sum(sp[i,2:dimen[1]])/(dimen[1]-1)
}
write(sp_node, 'shortest_path_node_ava.txt',ncolumns=1)
remove(sp_node)
```

```
# 2] Global properties(not used)

# Average shortest path for the whole graph
av_path_length<-average.path.length(g)
# Diameter of the graph, i.e. length of the longest geodesic
# (shortest path)
diam<-diameter(g)
ggg<-girth(g)
# length of the shortest circle in it.
girth<-ggg$girth
```

A.3 Laplacian matrix of the 7-node graph in Figure 20

The Laplacian matrix L ($=D-A$) of the unweighted graph in Figure 20 is expressed as follows (see an example of weighted Laplacian matrix in Section 3.2):

$$L=\begin{bmatrix} 2 & -1 & 0 & 0 & 0 & -1 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & -1 & -1 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 & 3 & -1 & 0 \\ -1 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}.$$

This page intentionally left blank

List of symbols/abbreviations/acronyms/initialisms

AF	Aggravating Factor
DND	Department of National Defence
GPS	Global Positioning System
JC	Jump Condition
PR	Pattern Recognition
PREVU	Prediction and REcognition of VUlnerabilities
ROC	Receiver Operating Characteristic
UAV	Unmanned Aerial Vehicle

This page intentionally left blank

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Defence R&D Canada – Valcartier
2459 Pie-XI Blvd North
Quebec (Quebec)
G3J 1X5 Canada

2. SECURITY CLASSIFICATION (Overall security classification of the document including special warning terms if applicable.)

UNCLASSIFIED
(NON-CONTROLLED GOODS)
DMC A
Review: GCEC June 2010

3. TITLE (The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)

Pattern Recognition of Socio-technical Network Vulnerabilities: Modeling and preliminary results

4. AUTHORS (last name, followed by initials – ranks, titles, etc. not to be used)

Léchevin, N.; Jousselme, A.-L.; Maupin P.

5. DATE OF PUBLICATION (Month and year of publication of document.)	6a. NO. OF PAGES (Total containing information, including Annexes, Appendices, etc.)	6b. NO. OF REFS (Total cited in document.)
December 2013	78	52

7. DESCRIPTIVE NOTES (The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)

Technical Report

8. SPONSORING ACTIVITY (The name of the department project office or laboratory sponsoring the research and development – include address.)

Defence R&D Canada – Valcartier
2459 Pie-XI Blvd North
Quebec (Quebec)
G3J 1X5 Canada

9a. PROJECT OR GRANT NO. (If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)	9b. CONTRACT NO. (If appropriate, the applicable number under which the document was written.)
15af07	

10a. ORIGINATOR'S DOCUMENT NUMBER (The official document number by which the document is identified by the originating activity. This number must be unique to this document.)	10b. OTHER DOCUMENT NO(s). (Any other numbers which may be assigned this document either by the originator or by the sponsor.)
DRDC Valcartier TR 2013-409	

11. DOCUMENT AVAILABILITY (Any limitations on further dissemination of the document, other than those imposed by security classification.)

Unlimited

12. DOCUMENT ANNOUNCEMENT (Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.))

Unlimited

13. ABSTRACT (A brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual.)

When faced with potentially disruptive events, the state of a network may unexpectedly evolve to regions of the state space where safe operating conditions are no longer ensured. It thus highly desirable to relate the network's characteristics and operating conditions to its vulnerabilities, if any, in order to mitigate risk expressed as a function of network inoperability and loss of quality of service. A pattern recognition approach is adopted to relate the structural features of the network to the loss of operating nodes and edges. Two types of networks are considered in this document. A network characterized by flow conservation and capacity constraints is adapted from a fuse model, which may lead, in some instances, to cascading events. A tactical swarm of robots is deployed either to achieve terrain surveillance coverage or to maintain client connectivity so that every client can communicate in remote area. In both cases, the swarm of robots should maintain its connectivity at each time instant. The swarm deployment adapts to the loss of a robot caused by such factors as hardware/software failure, enemy action, or the presence of malware. The motion strategy prioritizes the client coverage, which may entail possible losses of connectivity. Given the motion strategy at hand, the swarm presents vulnerabilities related to the loss of some nodes. The classifier, instrumental in performing pattern recognition, is trained from a sample of networks obtained by some probabilistic generator. The classifier is shown to model, and to some extent, predict quickly the vulnerabilities of a class of networks as a function of their structural properties.

La présence d'événements perturbant le fonctionnement des réseaux peut entraîner, sous certaines conditions, des dysfonctionnements importants. Il est donc souhaitable de cerner les conditions de fonctionnement et les caractéristiques pertinentes du réseau afin de modéliser ses vulnérabilités et d'en atténuer les risques encourus exprimant, entre autres, les baisses du niveau de qualité de service du réseau. Les techniques de reconnaissance de forme sont appliquées avec comme hypothèse de travail, une corrélation importante entre caractéristiques structurelles et vulnérabilité du réseau. Deux types de réseaux présentant des phénomènes d'avalanche sont proposés pour tester l'approche. Un réseau respectant le principe de conservation de l'énergie et intégrant des contraintes sur la capacité maximale de transport des liens (inspiré d'un réseau de fusibles) permet de générer, dans certain cas, des phénomènes de cascade. Un réseau tactique de robots est déployé dans un environnement dynamique afin de répondre au problème posé par deux scénarios possibles : (i) la couverture d'un terrain pour y accomplir des tâches de surveillance, et (ii) le maintien de la connectivité d'un réseau de clients voulant communiquer à partir de régions éloignées. Le déploiement des robots est capable de s'adapter à d'éventuelles pertes, notamment causés par des bris, une attaque ennemie ou par la présence de programmes malveillants. La stratégie de déplacement priorise la couverture des clients et maintient, lorsque cela est possible, la connectivité du réseau. La perte d'un ou plusieurs robots peut constituer une vulnérabilité car le redéploiement du réseau peut engendrer de nouvelles pertes de liens et des bris de connectivité. Le classifieur est entraîné à partir d'un échantillon de réseaux obtenu à l'aide d'un générateur aléatoire de réseaux. Il est montré que ce classifieur représente assez bien le lien entre les caractéristiques structurelles de la classe de réseaux étudiée et ses vulnérabilités potentielles, permettant ainsi une prédiction rapide des vulnérabilités d'une classe de réseaux.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (Technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus, e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus identified. If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title.)

Network, infrastructure, vulnerability, pattern recognition, prediction

Defence R&D Canada

Canada's Leader in Defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca