Optimal Algorithms for Detecting Network Stability

Dimitrios Koukopoulos,[1] Stavros D. Nikolopoulos,[2] Leonidas Palios,[2]
Paul G. Spirakis [3]

[1] *Department of Cultural Heritage Management and New Technologies, University of Ioannina, GR-30100 Agrinio, Greece.*
koukopou@ceid.upatras.gr

[2] *Department of Computer Science, University of Ioannina, GR-45110 Ioannina, Greece.*
{stavros, palios}@cs.uoi.gr

[3] *Department of Computer Engineering and Informatics, University of Patras, GR-26500 Patras, Greece.*
spirakis@ceid.upatras.gr

Abstract: A *packet-switching* network is *universally stable* if, for any greedy protocol and any adversary of injection rate less than 1, the number of packets in the network remains bounded at all times. A very natural question that arises in the context of network stability is whether there is a fast way to detect the existence of the universal stability property of a network based on the network structure. In this work, we embark on a systematic study of this question in the context of *Adversarial Queueing Theory*, which assumes that an adversary controls the rates of packet injections and determines packet paths. Also, the adversary can restrict packets to follow non-simple paths (paths do not contain repeated edges, but they contain repeated vertices) or simple paths (paths do not contain repeated edges and vertices). Within this framework, we consider the *number of edges* and the *number of vertices* as crucial structural parameters of the network, and we present a comprehensive collection of results for the detection of network universal stability in the form of optimal algorithms. In addition, we extend the proposed algorithms to show that the preservation or not of the stability properties of a network when it undergoes malicious attacks of an adversary/intruder whose power is to try to add links can be detected in constant time.

Keywords: Packet-Switched Communication Networks, Network Stability, Linear Algorithms, Graph Theory, Intrusion Detection, Adversarial Queueing Theory.

1 Introduction

Motivation-Framework. A lot of research has been done in the field of *packet-switched* communication networks for the specification of their behavior. In such networks, packets arrive dynamically at the nodes and they are routed in discrete time steps across the edges. A major issue that arises in such a setting is that of *universal stability*– will the number of packets in the network remain bounded at all times against any adversary and any protocol? The answer to this question is non-trivial; since the property of universal stability of networks is a predicate quantified over all protocols and adversaries,

it might at first appear that it is not a decidable property. But, this is not the case. Alvarez *et al.* in [4] show that detecting the universal stability of networks requires polynomial time. However, from a practical standpoint, the detection of universal stability in polynomial time maybe not enough. It remains open if there is an algorithm that can detect the universal stability of networks in linear time.

A key feature of contemporary large-scale platforms for distributed communication and computation, such as the *Internet,* is their *survivability*: the ability of computers and networks to be resilient in the face of an attack [6, 33]. However, there are no short-term solutions to eliminate attacks. Thus, it is impossible to close all security loopholes in a computer system by building firewalls or using cryptographic techniques. As a result, intrusion detection has emerged as a key technique for network survivability and, consequently, Internet security [27]. An algorithm that performs intrusion detection should be fast enough in order to eliminate the damage a network can suffer by an intruder. The universal stability property of a network is related to the time delay a packet suffers to reach its destination. Consequently, universal stability is also related to the quality of service that a network provides to its end-users. Thus, there is a necessity for fast detection of any change this network property can undergo by a malicious adversary/intruder. Such a change can be provoked by the dynamic addition of links on the network topology.

Objectives. The underlying goal of our study is to investigate the computational complexity of detecting universal stability of networks when packets are injected by an adversary. It may depend on the network structure, the traffic pattern defined by the adversary and the protocol employed to resolve packet conflicts. The traffic pattern controls where and how packets are injected into the network, and defines their path (trajectory). We choose, as a test-bed, the case of dynamic adversarial addition of links on a universally stable network. We ask, in particular, how fast can be detected any change in the universal stability property of a network under such an adversarial/intrusion attack.

Framework of Adversarial Queueing Theory. We consider a packet-switched communication network in which packets arrive dynamically at the nodes with predetermined paths, and they are routed at discrete time steps across the edges (links). We focus on a basic adversarial model for packet arrival and path determination that has been introduced in a pioneering work by Borodin *et al.* [10], under the name *Adversarial Queueing Theory*. Roughly speaking, this model views the time evolution of a packet-switched communication network as a game between an *adversary* and a *protocol*. At each time step, the adversary may inject a set of packets into some nodes. For each packet, the adversary specifies a path that the packet must traverse; when the packet arrives to destination, it is absorbed by the system. When more than one packets wish to cross a queue at a given time step, a *contention-resolution* protocol is employed to resolve the conflict. The protocols considered are usually greedy– ones that always advance a packet across a queue (but one packet at each discrete time step) whenever there resides at least one packet in the queue. A crucial parameter of the adversary is its *injection rate* ρ, where $0 < \rho < 1$. Among the packets that the adversary injects in any time interval I, at most $\lceil \rho |I| \rceil$ can have paths that contain any particular edge. Such a model allows for adversarial injection of packets, rather than for injection according to a randomized, oblivious process (cf. [12]).

In this work, we embark on a study of the impact of the topological structure of the networks and its dynamic changes on their correctness and performance properties. More specifically, we wish to pose the general question of whether it would be possible to detect universal stability from the knowledge of the topological structure of the network. This subfield of study was initiated by Andrews *et al.* in [7] where they show that the family of *undirected-path universally stable graphs* is minor-closed and that there exists a finite set of basic undirected graphs such that a graph is stable, if and only if it does not contain as a minor any of the graphs in that set. Additionally, we wish to investigate of which correctness and performance properties of *packet-switched* networks are maintained and which are not in the presence of dynamic additions of links.

Stability. Roughly speaking, a protocol P is *stable* [10] on a network $\mathcal{G}$ against an adversary $\mathcal{A}$ of rate ρ if there is a constant B (which may depend on $\mathcal{G}$ and $\mathcal{A}$) such that the number of packets in the system is bounded at all times by B. On the other hand, *a protocol* P *is universally stable* [10] if it is stable against every adversary of rate less than 1 and on every network. We also say that *a network* $\mathcal{G}$ *is universally stable* [10] if every greedy protocol is stable against every adversary of rate less than 1 on $\mathcal{G}$. Moreover, the property of universal stability can be viewed under two different approaches; we refer to *simple-path universal stability* when packets follow *simple paths* (paths do not contain repeated edges, and vertices), while we refer to *universal stability* when packets follow *not simple paths* (paths do not contain repeated edges, but they can contain repeated vertices) [4].

Contribution. Our work interestingly shows how network structure precisely affects the universal stability of networks. In particular, we study the universal stability property of networks in the Adversarial Queueing model. Our results are three-fold; they are summarized as follows:

1 First, we present an algorithm that detects whether a network where packets are adversarially injected with not simple paths, is universally stable or not in linear time $O(m+n)$ where m is the number of network edges and n is the number of network vertices. This result improves the result of Alvarez *et al.* [4, Theorem 13] which states that checking the universal stability of a given directed graph can be done in polynomial time.

2 Additionally, we propose an algorithm that detects whether a network where packets are adversarially injected with simple paths, is universally stable or not in linear time $O(m+n)$. This result improves the result of Alvarez *et al.* [4, Theorem 16] where a polynomial time algorithm is presented for checking the universal stability of a given directed graph.

3 We demonstrate two linear algorithms for handling attacks of an intruder/adversary that inserts dynamically additional links on a universally stable network. These algorithms check the preservation of the universal stability property of networks under such attacks when packets are injected into the network with non-simple or simple paths correspondingly. The proposed algorithms here are extensions of the ones that are proposed for the universal stability detection.

Related Work. Adversarial Queueing Theory was developed by Borodin *et al.* [10] as a more realistic model that replaces traditional stochastic assumptions in Queueing Theory [12] by more robust, worst-case ones. It received a lot of interest and attention in the study of stability and instability issues (see, e.g., [7, 4, 14, 23, 24, 25, 28]). Universal stability of various natural greedy protocols has been established by Andrews *et al.* [7]. Additionally, certain well-known network topologies (DAGs, trees, ring) have been proved universally stable in [7, 10].

The subfield of proposing a characterization for universally stable networks was first initiated in [7] proving that there exists a finite set of basic undirected graphs such that a graph is stable, if and only if it does not contain as a minor any of the graphs in that set. This result guarantees decidability in polynomial time; however a constructive proof is not presented. This result was significantly improved in [16, 18, 4]. Especially, in [4] a tight characterization of stable topologies for directed graphs is given. In particular, a digraph where packets are injected in non-simple paths is universally stable if and only if it does not contain as subgraph any of the extensions of $\mathcal{U}_1$ or $\mathcal{U}_2$ (see Figures 1, 2) [4, Lemma 7]. Moreover, a digraph where packets are injected in simple paths is universally stable if and only if it does not contain as subgraph any of the extensions of $\mathcal{S}_1$ or $\mathcal{S}_2$ or $\mathcal{S}_3$ or $\mathcal{S}_4$ (see Figures 3, 4) [4, Theorem 12]. Furthermore, in [4] they show that detecting universal stability of networks requires polynomial time. Recently, Blesa [9] presented a polynomial-time algorithm for both FIFO stability and FIFO simple-path stability of directed multigraphs.

Universal Stability in Dynamic Adversarial Settings-Intrusion Detection. Dynamic adversarial environments can be used to model intrusion attacks as an intruder can behave like an adversary that tries to change network environment parameters concerning network topology, packet service rate or the used contention-resolution protocol. In the community of Security, the study of intrusion detection received a lot of interest [8, 15, 17, 21, 27, 34]. Thus, there are more than 100 commercial tools and research prototypes for intrusion detection [29, 30]. Furthermore, there is a number of techniques for identifying sources of intrusion [31, 32]. On the other hand, in the community of Stability there are a few works considering dynamic adversarial settings [3, 11, 22]. However, to the best of our knowledge this is the first work that uses stability theory as a tool for detecting an intrusion attack.

2 Theoretical Framework

The model definitions are patterned after those in [10, Section 3]. We consider that a routing network is modelled by a finite multi-digraph G on n vertices and m edges; the term *multi-digraph* is used when multiple edges are allowed in a digraph. Each node $x \in V(G)$ represents a communication switch, and each edge $e \in E(G)$ represents a link between two switches. In each node, there is a buffer (queue) associated with each outgoing link. Time proceeds in discrete time steps. Buffers store packets that are injected into the network with a route, which is a simple directed path in G. A *packet* is an atomic entity that resides at a buffer at the end of any step. It must travel along paths in the network from its *source* to its *destination,* both of which are nodes in the network. When a packet is injected, it is placed in the buffer of the first link on its route. When a packet reaches its destination, we say that it is *absorbed.* During each step, a packet may be sent from its current node along one of the outgoing edges from that node.

We consider finite directed and undirected simple graphs and multi-digraphs with no loops. A directed (resp. undirected) edge is a directed (resp. undirected) pair of distinct vertices $x, y \in V(G)$, and is denoted xy. The *multiplicity* of a vertex-pair xy of a digraph G, denoted by $\lambda(xy)$, is the number of edges joining the vertex x to y in G. For a set $C \subseteq V(G)$ of vertices of the graph G, the subgraph of G *induced* by C is denoted $G[C]$; for a set $S \subseteq E(G)$ of edges, the subgraph of G *spanned* by S is denoted $G\langle S\rangle$.

Any packets that wish to travel along an edge e at a particular time step, but are not sent, wait in a queue for the edge e. At each step, an *adversary* generates a set of requests. A *request* is a *path* specifying the route that will be followed by a packet. We say that the adversary generates a set of packets when it generates a set of requested paths.

A *path* in G is a sequence of vertices $(v_0, v_1, \ldots, v_k)$ such that $v_i v_{i+1} \in E(G)$ for $i = 0, 1, \ldots, k-1$; we say that this is a path from v_0 to v_k (for short, v_0–v_k path) of *length* k. A path is called *simple* if none of its vertices occurs more than once; it is called *trivial* if its length is equal to 0. A path is *closed* (resp. *open*) if $v_0 = v_k$ (resp. $v_0 \neq v_k$). A closed path $(v_0, v_1, \ldots, v_{k-1}, v_0)$ is a *cycle* of length k; the closed path (v_0, v_1, v_0) is called 2-cycle.

A *connected component* (or component) of an undirected graph G is a maximal set of vertices, say, $C \subseteq V(G)$, such that for every pair of vertices $x, y \in C$, there exists a x–y path in the subgraph $G[C]$ of G induced by the vertices in C. A component is called *non-trivial* if it contains two or more vertices; otherwise, it is called *trivial.* A *biconnected component* (or bicomponent) of an undirected graph G is a maximal set of edges such that any two edges in the set lie on a simple cycle of G [13]; G is called *biconnected* if it is connected and contains only one biconnected component.

A *strongly connected component* (or scc) of a directed graph G is a maximal set of vertices $C \subseteq V(G)$ such that for every pair of vertices x and y in the set C, there exists both a (directed) x–y path and a

(directed) $y-x$ path in the subgraph of G induced by the vertices in C; the graph G is called *strongly connected* if it is connected and contains only one scc. The *acyclic component graph* G_{scc} of the digraph G is an acyclic digraph obtained by contracting all edges within each strongly connected component of G so that only a single vertex remains in each component. The *underlined graph* $G_{u\ell}$ of the digraph G is an undirected graph which results after making all the edges of G undirected and consolidating any duplicate edges.

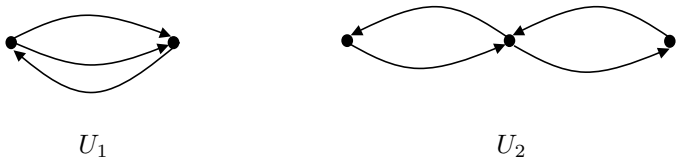
A *strongly biconnected component* (or bi-scc) of a directed graph G is a maximal set of edges $S \subseteq E(G)$ such that $G\langle S\rangle$ is a strongly connected graph and its underlined graph $G\langle S\rangle_{u\ell}$ is biconnected; the graph G is called *strongly biconnected* if it is strongly connected and contains only one bi-scc. The graph U_1 of Fig. 1 is strongly biconnected, while the graph U_2 contains two bi-scc.

The *subdivision* operation on an edge xy of a digraph G consists of the addition of a new vertex w and the replacement of xy by the two edges xw and wy; hereafter, we shall call it *edge-subdivision* operation.

Given a digraph G on n vertices and m edges, $\mathcal{E}(G)$ denotes the family of digraphs which contains the digraph G and all the digraphs obtained from G by successive edge-subdivisions.

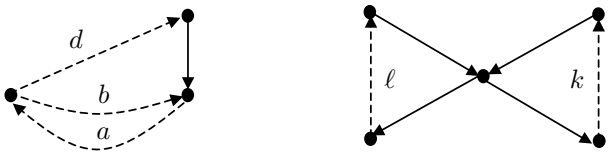
It has been proved that the digraphs U_1 and U_2 of Fig. 1 are not universally stable; they are the minimum forbidden subgraphs characterizing universal stability. Moreover, the family of digraphs obtained from U_1 and U_2 by successive edge-subdivisions are also not universally stable, i.e., the digraphs in $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$ [4] (see Fig. 2). The following result holds:

Lemma 2.1. (Alvarez, Blesa, and Serna [4]): *A digraph G is universally stable if and only if G does not contain as a subgraph any of the digraphs in $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$.*



U_1 U_2

Figure 1: Not universally stable digraphs.



(a) (b)

Figure 2: Family of digraphs formed by extensions of U_1 and U_2, where $a \geq 1$, $b \geq 1$, $d \geq 0$, $\ell \geq 0$, and $k \geq 0$. (a) a digraph in $\mathcal{E}(U_1)$; (b) a digraph in $\mathcal{E}(U_2)$.

Observation 2.1. Let G be a directed graph of the family $\mathcal{E}(U_1) \cup \mathcal{E}(U_2)$. Then, the graph G has the following structure:

(a) it consists of a cycle $C = (x_0, x_1, x_2, \ldots, x_\ell, x_0)$, $\ell \geq 1$, and

(b) a path $P = (x_i, y_1, y_2, \ldots, y_k, x_j)$ such that, $y_1, y_2, \ldots, y_k \notin C$, $x_i, x_j \in C$ and $k \geq 0$.

It is easy to see that, if P is an open path, i.e, $x_i \neq x_j$, then $G \in \mathcal{E}(U_1)$, whereas if P is a closed path, i.e, $x_i = x_j$, then $G \in \mathcal{E}(U_2)$.

Let G be a digraph on n vertices and m edges. We denote by G^* the element of $\mathcal{E}(G)$ which obtained from G by applying an edge-subdivision operation on each edge $uv \in E(G)$. Obviously, G^* has $n+m$ vertices and $2m$ edges. Moreover, G^* does not contain 2-cycles; in particular, every cycle in G^* has length greater than or equal to 4. We call G^* *one-subdivided* graph of G.

Below we present some results on which our algorithm for detecting universal stability rely.

Lemma 2.2. *Let G be a directed graph and let G^* be its one-subdivided graph. The graph G is not universally stable if and only if G^* contains a subgraph $H \in \mathcal{E}(U_1) \cup \mathcal{E}(U_2)$.*

Lemma 2.3. *Let G be a directed graph and let G^* be its one-subdivided graph. Let $S_1, S_2, \ldots, S_k$ be the strongly connected components of G^* and let n_i and m_i be the number of vertices and edges of the strong component S_i, respectively. Then, G is not universally stable if and only if G^* has a strong component S_i such that $m_i > n_i$, $1 \leq i \leq k$.*

According to the classification introduced in [4], we will also differentiate and refer to the property of universal stability in the case in which packets follow *simple paths* (paths do not contain repeated edges, and vertices) as *simple-path universal stability*, leaving then the term *universal stability* to refer to the case in which packets follow *not simple paths* (paths do not contain repeated edges, but they can contain repeated vertices). We say *forbidden subgraphs* for network stability when the packets follow non-simple paths [4, 18] any graph obtained by replacing any edge of the graphs $\mathcal{U}_1$ and $\mathcal{U}_2$ (see Figure 1) by disjoint directed paths. We say *forbidden subgraphs* for network stability when the packets follow simple paths [4] any graph obtained by replacing any edge of the graphs $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, and $\mathcal{S}_4$ (see Figures 3 and 4) by disjoint directed paths.

It has been showed that all the digraphs in $\mathcal{E}(S_1) \cup \mathcal{E}(S_2) \cup \mathcal{E}(S_3) \cup \mathcal{E}(S_4)$ are not simple-path universally stable [4].

Lemma 2.4. (Alvarez, Blesa, and Serna [4]): *A digraph G is simple-path universally stable if and only if G does not contain as a subgraph any of the digraphs in $\mathcal{E}(S_1) \cup \mathcal{E}(S_2) \cup \mathcal{E}(S_3) \cup \mathcal{E}(S_4)$.*

Let G be a digraph on n vertices and m edges. We denote by $\widehat{G}$ the digraph which is obtained from G by setting the multiplicity of each edge xy of G to 1, if $\lambda(xy) \geq 2$. Obviously, $\widehat{G}$ is simple digraph and it has n vertices and $m' \leq m$ edges. We call $\widehat{G}$ *reduced* graph of the graph G.

Below we present some results on which our algorithm for detecting simple-path universal stability relies.

Lemma 2.5. *Let G be a directed graph, $\widehat{G}$ be the reduced graph of G, and $S_1, S_2, \ldots, S_k$ be the scc of $\widehat{G}$. Let $S_{i1}, S_{i2}, \ldots, S_{ik_i}$ be the bi-scc of the scc S_i and let n_{ij} and m_{ij} be the number of vertices and edges of the bi-scc S_{ij}, respectively. Then, G is not simple-path universally stable if and only if $\widehat{G}$ has a strong component S_i which satisfies one of the following conditions:*

- *(i) S_i contains a bi-scc S_{ij} such that: $n_{ij} \geq 3$, $G\langle S_{ij}\rangle_{u\ell}$ is a cycle, and there exists an edge xy in $G\langle S_{ij}\rangle$ such that $\lambda(xy) \geq 2$;*
- *(ii) S_i contains a bi-scc S_{ij} such that: $n_{ij} \geq 3$ and $G\langle S_{ij}\rangle_{u\ell}$ is not a cycle;*
- *(iii) S_i contains two bi-scc S_{ip} and S_{iq} such that: $n_{ip} \geq 3$, $n_{iq} \geq 3$, and both graphs $G\langle S_{ip}\rangle_{u\ell}$ and $G\langle S_{iq}\rangle_{u\ell}$ are cycles;*

where $1 \leq i \leq k$ and $1 \leq j, p, q \leq k_i$.

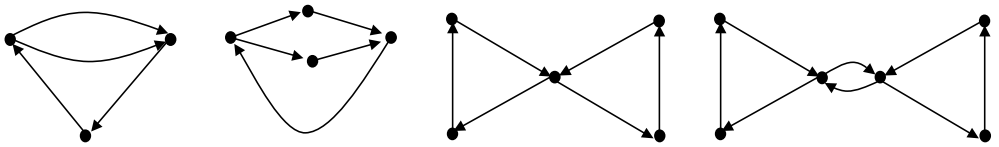


Figure 3: Not simple-path universally stable digraphs.

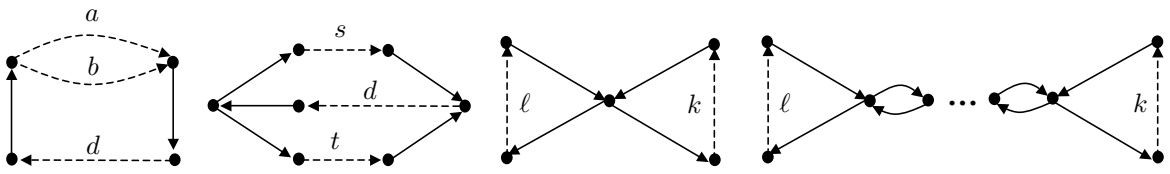


Figure 4: Family of digraphs formed by extensions of S_1, S_2, S_3, and S_4, where $a \geq 1$, $b \geq 1$, $d \geq 0$, $s \geq 0$, $t \geq 0$, $\ell \geq 1$, and $k \geq 1$. (a) a digraph in $\mathcal{E}(S_1)$; (b) a digraph in $\mathcal{E}(S_2)$; (c) a digraph in $\mathcal{E}(S_3)$; (d) a digraph in $\mathcal{E}(S_4)$.

Sketch of Proof: By definition, a bi-scc S_{ij} of the scc S_i of the digraph $\widehat{G}$ consists of either a cycle $C = (x_0, x_1, x_2, \ldots, x_r, x_0)$, $r \geq 2$, or a cycle $C = (x_0, x_1, x_2, \ldots, x_r, x_0)$, $r \geq 2$, and a path $P = (x_i, y_1, y_2, \ldots, y_{r'}, x_j)$ such that $y_1, y_2, \ldots, y_{r'} \notin C$, $x_i \neq x_j$ and $r' \geq 1$.

($\Longleftarrow$) It is easy to see that if condition (i) holds, then the graph $G\langle S_{ij}\rangle$ contains a subgraph $H \in \mathcal{E}(S_1)$. If condition (ii) holds, then the bi-scc S_{ij} consists of a cycle $C = (x_0, x_1, x_2, \ldots, x_r, x_0)$, $r \geq 2$, and a path $P = (x_i, y_1, y_2, \ldots, y_{r'}, x_j)$ such that $x_i \neq x_j$ and $r' \geq 1$. Thus, the graph $G\langle S_{ij}\rangle$ contains a subgraph $H \in \mathcal{E}(S_2)$. If condition (iii) holds, then the graph $G\langle S_{ij}\rangle$ contains a subgraph $H \in \mathcal{E}(S_2) \cup \mathcal{E}(S_4)$.

($\Longrightarrow$) Suppose now that G is not simple-path universally stable. Then, G contains a subgraph $H \in \mathcal{E}(S_1) \cup \mathcal{E}(S_2) \cup \mathcal{E}(S_3) \cup \mathcal{E}(S_4)$, and, thus, H contains a cycle $C = (x_0, x_1, x_2, \ldots, x_r, x_0)$, $r \geq 2$. It follows that C belongs to a scc S_i of the graph $\widehat{G}$, $1 \leq i \leq k$. Let S_{ij} be the bi-scc of S_i which contains the cycle C. Since $r \geq 2$, the bi-scc S_{ij} has at least three vertices, i.e., $n_{ij} \geq 3$. We distinguish two cases:

Case (a): S_{ij} contains the cycle C and a path $P = (x_i, y_1, y_2, \ldots, y_{r'}, x_j)$, $r' \geq 1$. Then, $H \in \mathcal{E}(S_2)$ and $G\langle S_{ij}\rangle_{u\ell}$ is not a cycle. Thus, the condition (ii) holds.

Case (b): S_{ij} contains only the cycle $C = (x_0, x_1, x_2, \ldots, x_r, x_0)$, $r \geq 2$. If there exists an edge $x_i x_{i+1 mod r}$ in C such that $\lambda(x_i x_{i+1 mod r}) \geq 2$ in G, then $H \in \mathcal{E}(S_1)$ and, since $G\langle S_{ij}\rangle_{u\ell}$ is a cycle, the condition (i) holds. If there exists no such edge in C, then $H \in \mathcal{E}(S_3) \cup \mathcal{E}(S_4)$. Thus, H contains another cycle $C' = (x'_0, x'_1, x'_2, \ldots, x'_r, x'_0)$, $r \geq 2$, which belongs to a bi-scc, say, S_{iq}, of S_i. If the conditions (i) and (ii) do not hold for the bi-scc S_{iq}, then the graph $G\langle S_{iq}\rangle_{u\ell}$ is a cycle and $n_{iq} \geq 3$. Thus, the condition (iii) holds. ∎

3 Detecting Universal Stability

In this section we present optimal algorithms for detecting universal and simple-path universal stability on a digraph G. Both algorithms take as input a digraph G on n vertices and m edges, and decide if G

is universally stable or simple-path universally stable, resp., in $O(n+m)$ time using $O(n+m)$ space.

3.1 Universal stability

Our algorithm for detecting universal stability on a digraph G relies on the result stated in Lemma 2.3; it works as follows:

Algorithm Univ_Stability

Input: a digraph G on n vertices and m edges;

Output: yes, if G is universally stable; otherwise, no.

1. Construct the one-subdivided graph G^* of the input digraph G;
2. Compute the strongly connected components $S_1, S_2, \ldots, S_k$ of the digraph G^*, and the number of vertices n_i and edges m_i of each strong component S_i, $1 \leq i \leq k$;
3. for $i = 1$ to k do
   if $m_i > n_i$ then return no; exit;
4. return yes;

The correctness of the algorithm Univ_Stability follows from Lemma 2.3. We next compute its time and space complexity. The one-subdivided graph G^* of the input graph G can be constructed in $O(n+m)$ time, where n is the number of vertices and m is the number of edges of the graph G. The graph G^* has $n+m$ vertices and $2m$ edges, and, thus, the computation of the strong components of G^* can be done in $O(n+m)$ time. Thus, the whole algorithm runs in $O(n+m)$ time; the space needed is $O(n+m)$.

From the above description we conclude that the detection algorithm Univ_Stability runs in linear time and requires linear space. Hence, we have:

Theorem 3.1. *Let G be a digraph on n vertices and m edges. The algorithm Univ_Stability decides whether G is universal stable in $O(n+m)$ time and space.*

Theorem 3.2. *Let G be a digraph on n vertices and m edges. The universal stability of G can be decided in $O(n+m)$ time and space.*

3.2 Simple-path universal stability

Our simple-path universal stability detection algorithm relies on the result of Lemma 2.5; it works as follows:

Algorithm Simple-Path_Univ_Stability

Input: a digraph G on n vertices and m edges;

Output: yes, if G is simple-path universally stable; otherwise, no.

1. Construct the reduced graph $\widehat{G}$ of the input digraph G;
2. Compute the strong components $S_1, S_2, \ldots, S_k$ of the graph $\widehat{G}$, $1 \leq i \leq k$;
3. Compute the bi-scc $S_{i1}, S_{i2}, \ldots, S_{ik_i}$ of each strong component S_i, $1 \leq i \leq k$, and the number of vertices n_{ij} and edges m_{ij} of the bi-scc S_{ij}, $1 \leq j \leq k_i$;

4. for $i = 1$ to k do
   for $j = 1$ to k_i do
      if $n_{ij} \geq 3$ and $G\langle S_{ij}\rangle_{u\ell}$ is a cycle, and
         there exists an edge xy in $G\langle S_{ij}\rangle$ such that $\lambda(xy) \geq 2$, then return no; exit;
      if $n_{ij} \geq 3$ and $G\langle S_{ij}\rangle_{u\ell}$ is not a cycle, then return no; exit;
      if $n_{ij} \geq 3$ and $G\langle S_{ij}\rangle_{u\ell}$ is a cycle, then mark the bi-scc S_{ij};
   end-for
   if S_i contains at least two marked bi-scc then return no; exit;

5. return yes;

The correctness of the algorithm Simple-Path_Univ_Stability follows from Lemma 2.5. It is easy to see that the construction of the reduced graph $\widehat{G}$ of the input graph G can be done in $O(n+m)$ time. The graph $\widehat{G}$ has n vertices and $m' \leq m$ edges, and, thus, the computation of the strong components of $\widehat{G}$ can be completed in $O(n+m)$ time. The the bi-scc $S_{i1}, S_{i2}, \ldots, S_{ik_i}$ of each strong component S_i, $1 \leq i \leq k$, can be computed in $O(n+m)$ time because $\sum_{j=1,k} m_{ij} \leq m_i$ and $n_{ij} \leq m_{ij}$ since the bi-scc are biconnected and do not share edges. It is easy to see that all the operation of step 4 are executed in linear time. Thus, the algorithm runs in $O(n+m)$ time; the space needed is $O(n+m)$.

Thus, we can state the following results:

Theorem 3.3. *Let G be a digraph on n vertices and m edges. The algorithm Simple-Path_Univ_Stability decides whether G is simple-path universal stable in $O(n+m)$ time and space.*

Theorem 3.4. *Let G be a digraph on n vertices and m edges. The simple-path universal stability of G can be decided in $O(n+m)$ time and space.*

4 Detecting Intrusion Attacks

In this section, we prove that the malicious intentions of an adversary/intruder to lead a stable network in instability by adding links in specific parts of the network can be detected in constant time after a preprocessing phase in which we compute path information. In particular, given a universally stable digraph G and a pair of distinct vertices $x, y \in V(G)$, we want to decide whether the graph $G + xy$ is also universally stable, where xy is the directed edge from x to y.

Based on the results of Section 3, we can decide whether $G + xy$ is universally stable in linear time without any preprocessing by executing algorithm Univ_Stability. Thus, we can state the following result.

Theorem 4.1. *Let G be a (simple-path) universally stable digraph on n vertices and m edges. The preservation or not of (simple-path) universal stability of G after the dynamic addition of a link into the topology of G by an intruder can be decided in $O(n+m)$ time and space.*

We are interested in contracting a structure such that queries of the form "is the graph $G + xy$ universally stable?" can be answered in $O(1)$ time.

Let G be a universally stable digraph on n vertices and m edges, and let $\widehat{G}$ be the reduced graph of G; recall that $\widehat{G}$ has n vertices and $m' \leq m$ edges. As in Lemma 2.3, each non-trivial strong component of $\widehat{G}$ forms a cycle; a trivial strong component consists of only one vertex. Thus, we have the following observation.

Observation 4.1. Let G be a universally stable digraph, $\widehat{G}$ be its reduced graph, and let $S_1, S_2, \ldots, S_k$ be the strong components of $\widehat{G}$, $1 \leq i \leq k$. The acyclic component graph $\widehat{G}_{scc}$ of the digraph $\widehat{G}$ has the following property: it consists of k vertices $v_1, v_2, \ldots, v_k$, the vertex v_i corresponds to the strong component S_i of $\widehat{G}$, and the strong component S_i is either a cycle, i.e., $m_i = n_i$, or a trivial component, i.e., $n_i = 1$ and $m_i = 0$.

Let G be a multi-digraph and let $x, y \in V(G)$ be a pair of distinct vertices. We say that the vertices x and y form an *xy-pair* if there exists a directed path from x to y in G, and we say that they form an *xy-multi-pair* if there exist more than one directed path from x to y in G, or a directed path $(x = u_0, u_1, \ldots, u_k = y)$ containing an edge $u_i u_{i+1}$ with $\lambda(u_i u_{i+1}) > 1$, $k \geq 1$. Then, we can prove the following lemma.

Lemma 4.1. *Let G be a universally stable digraph and let $x, y \in V(G)$ be a pair of distinct vertices. Let $\widehat{G}$ be the reduced graph of G, $S_1, S_2, \ldots, S_k$ be the strongly connected components of $\widehat{G}$ and let n_i and m_i be the number of vertices and edges of the strong component S_i, respectively. The graph $G + xy$ is not universally stable if and only if one of the following conditions holds:*

(i) $x, y \in S_i$, $1 \leq i \leq k$;

(ii) $x \in S_i$ *and* $y \in S_j$ *where* $i \neq j$ *and at least one of* n_i, n_j *is larger than* 1, *and* x, y *form an yx-pair in* G;

(iii) $x \in S_i$ *and* $y \in S_j$ *where* $i \neq j$ *and* $n_i = n_j = 1$, *and* x, y *form an yx-multi-path in* G.

Based on this lemma, we next present an algorithm for detecting whether a graph G preserves its universal stability after the addition of a link xy into the topology of G. Our algorithm works on the acyclic component graph $\widehat{G}_{scc}$ of the digraph $\widehat{G}$ and uses path information of $\widehat{G}_{scc}$ which we have computed in a preprocessing stage. Let $v_1, v_2, \ldots, v_k$ be the vertices of $\widehat{G}_{scc}$. We say that an edge $v_i v_j$ in $\widehat{G}_{scc}$ is *thick* if the exists more than one edge in G with their start-points in S_i and their end-points in S_j. We say that there is a *$v_i v_j$-path* in $\widehat{G}_{scc}$ if there is a (directed) path from v_i to v_j, where $v_i, v_j \in V(\widehat{G}_{scc})$.

We can keep all the information regarding the types of directed paths connecting the vertex v_i to v_j in $\widehat{G}_{scc}$ using $O(n^2)$ space; hereafter, we call this information *path-information*. Our algorithm takes as input the path-information of a universally stable digraph G on n vertices and m edges and a pair of vertices $x, y \in V(G)$, and detects in $O(1)$ time whether the graph $G + xy$ is universally stable; the description of our algorithm is as follows:

Algorithm Test_Link

Input: the path-information of a universally stable digraph G on n vertices and m edges and an (ordered) pair of vertices $x, y \in V(G)$;

Output: yes, if $G + xy$ is universally stable; otherwise, no.

1. if x, y belong to the same strong component of the digraph $\widehat{G}$, then return no; exit;
2. if $x \in S_i$ and $y \in S_j$, where $i \neq j$, and at least one of n_i, n_j is larger than 1 then
   if there exists a $v_j v_i$-path in $\widehat{G}_{scc}$, then return no; exit;
3. if $x \in S_i$ and $y \in S_j$, where $i \neq j$, and $n_i = n_j = 1$ then
   if there exist more than one $v_j v_i$-path in $\widehat{G}_{scc}$, then return no; exit;
   if there exists a $v_j v_i$-path in $\widehat{G}_{scc}$ with a thick edge, then return no; exit;
4. return yes;

The correctness of the algorithm follows from Lemma 4.1. Moreover, the time taken by the algorithm to test the addition of a (directed) link xy takes $O(1)$ time thanks to the stored path-information and the information on the strong components of $\widehat{G}$ (their sizes and the vertices participating in them). Hence, we state the following theorem.

Theorem 4.1. *Let G be a universally stable digraph on n vertices and m edges. Given the path-information of the acyclic component graph $\widehat{G}_{scc}$ of the digraph $\widehat{G}$, the preservation or not of universal stability of G after the addition of a link into the topology of G by an intruder can be decided in $O(1)$ time.*

Algorithm Test_Link only tells us whether the addition of a link on the same base network will preserve its universal stability; this has the advantage of guaranteeing constant time complexity but also the limitation that the much more interesting approach of dynamically maintaining the network under the addition of links that would not lead to instability is not supported (note that such a dynamic maintenance does not seem possible in constant time). We are currently working on exploiting ideas and results for the poly-logarithmic dynamic maintenance of the biconnected components of an undirected graph [20] in order to achieve dynamic maintenance of a universally stable network under the addition of links in poly-logarithmic time.

5 Directions for Further Research

In this work, we studied how efficiently the property of network universal stability can be detected considering directed graphs where packets are injected with non-simple or simple paths under the Adversarial Queueing Model. A lot of problems remain open. First, an interesting question concerning stability is that of deciding the stability of a network under a fixed protocol. In the literature there are only three results: deciding stability under FFS [2], NTG-LIS [4] and FIFO [9] protocol is polynomially solvable. Another interesting problem is whether there are upper bounds on injection rate that guarantee stability for forbidden subgraphs for universal stability.

References

[1] W. Aiello, E. Kushilevitz, R. Ostrovsky, and A. Rosén, Adaptive Packet Routing for Bursty Adversarial Traffic, *Proc. 30th Annual ACM Symposium on Theory of Computing*, Dallas, Texas, USA, pp. 359–368, 1998.

[2] C. Alvarez, M. Blesa, J. Diaz, A. Fernandez, M. Serna, The Complexity of Deciding Stability under FFS in the Adversarial Model, *Information Processing Letters* **90**, 261–266, 2004.

[3] C. Alvarez, M. Blesa, J. Diaz, A. Fernandez, M. Serna, Adversarial Models for Priority-Based Networks, *Proc. 28th International Symposium on Mathematical Foundations of Computer Science*, Bratislava, Slovakia. LNCS 2747, pp. 142–151, 2003.

[4] C. Alvarez, M. Blesa, M. Serna, A Characterization of Universal Stability in the Adversarial Queuing Model, *SIAM Journal on Computing* **34**, 41–66, 2004.

[5] C. Alvarez, M. Blesa, M. Serna, The Impact of Failure Management on the Stability of Communication Networks, *Proc. 10th Int'l Conference on Parallel and Distributed Systems*, Newport Beach, California, pp. 153–160, 2004.

[6] E. Amoroso, *Fundamentals of Computer Security Technology*, Prentice-Hall PTR, Upper Saddle River, NJ, 1994.

[7] M. Andrews, B. Awerbuch, A. Fernandez, J. Kleinberg, T. Leighton, Z. Liu, Universal Stability Results for Greedy Contention-Resolution Protocols, *Journal of the ACM* **48**, 39–69, 2001.

[8] M. Bishop, S. Cheung, C. Wee, J. Frank, J. Hoagland, S. Samorodin, The Threat from the Net, *IEEE Spectrum*, **34**, 56–63, 1997.

[9] M. Blesa, Deciding Stability in Packet-Switched FIFO Networks Under the Adversarial Queuing Model in Polynomial Time, *Proc. 19th International Symposium on Distributed Computing*, LNCS 3724, pp. 429–441, 2005.

[10] A. Borodin, J. Kleinberg, P. Raghavan, M. Sudan, D. Williamson, Adversarial Queueing Theory, *Journal of the ACM* **48**, 13–38, 2001.

[11] A. Borodin, R. Ostrovsky, Y. Rabani, Stability Preserving Transformations: Packet Routing Networks with Edge Capacities and Speeds, *Proc. 12th Annual ACM-SIAM Symposium on Discrete Algorithms*, Washington, DC, USA, pp. 601–610, 2001.

[12] H. Chen, D. D. Yao, *Fundamentals of Queueing Networks*, Springer-Verlag, 2000.

[13] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms* (2nd edition), MIT Press, Inc., 2001.

[14] J. Diaz, D. Koukopoulos, S. Nikoletseas, M. Serna, P. Spirakis, D. Thilikos, Stability and Non-Stability of the FIFO Protocol, *Proc. 13th Annual ACM Symposium on Parallel Algorithms and Architectures*, Crete, Greece, pp. 48–52, 2001.

[15] Y. Dong, S. Rajput, S. Hsu, Application Level Intrusion Detection using Graph-based Sequence Learning Algorithm, *Proc. IASTED Conference on Modelling and Simulation*, Cancun, Mexico, 2005.

[16] D. Gamarnik, Stability of Adaptive and NonAdaptive Packet Routing Policies in Adversarial Queueing Networks, *SIAM Journal on Computing* **32**, 371–385, 2003.

[17] S. Garfinkel, G. Spafford, *Practical UNIX and Internet Security* (Second Edition), O'Reilly & Associates, Inc., 1996.

[18] A. Goel, Stability of Networks and Protocols in the Adversarial Queueing Model for Packet Routing, *Networks* **37**, 219–224, 2001.

[19] M.C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, Academic Press, New York, 1980.

[20] J. Holm, K. de Lichtenberg, and M. Thorup, Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity, *Proc. Symposium on Theory of Computing'98*, 79–89, 1998.

[21] J. Howard, An Analysis of Security Incidents on the Internet, Ph.D. Dissertation, Carnegie Mellon University, 1995.

[22] D. Koukopoulos, The Impact of Dynamic Link Slowdowns on Network Stability, *Proc. 8th Int'l Symp. on Parallel Architectures, Algorithms and Networks*, Las Vegas, USA, pp. 340–345, 2005.

[23] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas, P. Spirakis, On the Stability of Compositions of Universally Stable, Greedy, Contention-Resolution Protocols, *Proc. 16th Int'l Symposium on DIStributed Computing*, Toulouse, France, 2002. LNCS 2508, pp. 88–102, 2002.

[24] D. Koukopoulos, M. Mavronicolas, S. Nikoletseas, P. Spirakis, The Impact of Network Structure on the Stability of Greedy Protocols, *Theory of Computing Systems* **38**, 425–460, 2005.

[25] D. Koukopoulos, S. Nikoletseas, P. Spirakis, Stability Issues in Heterogeneous and FIFO Networks under the Adversarial Queueing Model, *Proc. 8th Int'l Conference on High Performance Computing*, Hyderabad, India. LNCS 2228, pp. 3–14, 2001.

[26] D. Koukopoulos, M. Mavronicolas, P. Spirakis, Instability of Networks with Quasi-Static Link Capacities, *Proc. 10th Int'l Colloquium on Structural Information and Communication Complexity*, Sweden, pp. 179–194, 2003.

[27] S. Kumar, Classification and Detection of Computer Intrusions, Ph.D. Dissertation, Computer Sciences Department, Purdue University, 1995.

[28] Z. Lotker, B. Patt-Shamir, A. Rosén, New Stability Results for Adversarial Queuing, *SIAM Journal on Computing* **33**, 286–303, 2004.

[29] V. Paxson, BRO: A System for Detecting Network Intruders in Real Time, *Computer Networks*, **31**, 2435–2463, 1999.

[30] M. Roesch, The SNORT Network IntrusionDetection System, http://www.snort.org, 2002.

[31] S. Savage, D. Wetherall, A. Karlin, T. Anderson, Practical Network Support for IP Traceback, *Proc. ACM SIGCOMM*, pp. 295–306, 2000.

[32] A. Snoeren, C. Partridge, L. Sanchez, C. Jones, F. Tchakountio, S. Kent, HashBased IP Traceback, *Proc. ACM SIGCOMM*, pp. 3–14, 2001.

[33] W. Stallings, *Network and Internetwork Security Principles and Practice*, Prentice Hall, Englewood Cliffs, NJ, 1995.

[34] S. Upadhyaya, Real-Time Intrusion Detection with Emphasis on Insider Attacks, *Proc. 2nd Int'l Workshop on Mathematical Methods, Models, and Architectures for Computer Network Security*, St. Petersburg, Russia. LNCS 2776, pp. 82–85, 2003.