Chapter 23
Self-aware Networks: The Cognitive Packet Network and Its Performance

Erol Gelenbe

Abstract This article is a summary description of the cognitive packet network (CPN) which is an early example of a completely software-defined network (SDN) and of a fully implemented self-aware computer network (SAN). CPN has been completely implemented and is used in numerous experiments. CPN is able to observe its own internal performance as well as the interfaces of the external systems that it interacts with, in order to modify its behaviour so as to adaptively achieve objectives, such as discovering services for its users, improving their quality of service (QoS), reducing its own energy consumption, compensating for components that fail or malfunction, detecting and reacting to intrusions, and defending itself against attacks.

23.1 Introduction

The cognitive packet network (CPN) was largely implemented by the year 2004 [8] and is arguably the world's first software-defined network. It encapsulates IP packets into "dumb packets" (DPs) and also uses its own smart packets (SPs) and acknowledgement packets (ACKs). It routes its packets over a set of virtual nodes, or CPN nodes, so that one or more CPN nodes can be installed on computers or conventional network routers. CPN establishes awareness of its ongoing connectivity and quality of service using the SPs which constantly explore the network.

The measurements collected by the SPs are then returned by the ACKs to each of the nodes where traffic originates. CPN is an example both of a software-defined network (SDN) and of a self-aware computer network (SAN) [8, 13, 26], which uses concepts from autonomous search [14, 16, 22] to discover paths and observe its own internal performance as well as the interfaces of the external systems that it interacts with, in order to modify its behaviour so as to adaptively achieve certain objectives.

CPN's objectives or "goals" can include discovering services for its users, improving their quality of service (QoS) such as packet forwarding delay, packet loss and jitter, reducing its own energy consumption [10, 21], compensating for compo-

E. Gelenbe (✉)
Department of Electrical and Electronic Engineering, Imperial College, London SW7 2AZ, UK
e-mail: e.gelenbe@imperial.ac.uk

S. Kounev et al. (eds.), *Self-Aware Computing Systems*,
DOI 10.1007/978-3-319-47474-8_23

nents that fail or malfunction, detecting and reacting to intrusions, and defending itself against external attacks [6, 19].

In addition to its strong link to software-defined networks (SDN), whose basic ideas go back to the earlier field of programmable networks [4], the study of networks such as CPN is also part of the field of autonomic communications [5] and of naturally based computation [15]. It also draws on a long experience in designing and implementing distributed systems that must occasionally synchronise across large topologies [23]. SDNs also relate to the need for network virtualisation [29] over different hardware/software architectures for packet networks.

CPN has been completely implemented in software [8] and can be ported to any infrastructure that supports Linux. The routing engine of CPN is installed in each of the virtual nodes and is used to select paths for SPs based on a numerically expressed "goal function"; it is based on reinforcement learning and uses a neural critic that is based on the random neural network [17, 25], which has also been used in many other applications [1, 7, 18, 24]. Such learning systems can also be modelled and analysed using probability models [2, 11, 12].

23.2 Self-Awareness

A commonly used term such as self-awareness can elicit different reactions in the layman and also among scientists and engineers from different disciplines. We all struggle with such concepts and may not have a clear understanding, or different understandings, about what such terms actually mean. Thus, we first review some relevant aspects of self-awareness and then discuss how these can be embodied in the context of computer networks.

According to the philosopher John Locke, the self "depends on consciousness, not on substance". Thus, we can identify our "self" by being conscious or aware of our past, present and future thoughts, and through the memory of actions and the planning of actions. Any concept is necessarily juxtaposed to other opposing concepts, so that "self" needs to be related to, or separated from, the "other".

The notion of self also includes an integrated view of our past sensory inputs over a period of time, as well as of current sensory inputs. Some form of "autobiographical memory" should be a part of self-awareness. Sensing should also include what engineers call "condition monitoring", are we in good health, do we feel well and are we performing our various tasks as expected.

Self-awareness allows the individual to select and then drive not only her/his actions but also thinking, by choosing (for instance) to turn to a past memory or to a future plan, rather than to musing over thoughts generated by current external stimuli.

When we turn our attention to ourselves, we can also compare our current and past behaviour to our own standards, and to the behaviour of other entities external to ourselves, so that we become critics or evaluators of ourselves.

This also leads to a notion of awareness of our location in different spaces: the physical space where we observe, feed, work, move, etc., but it may also be a space of actions, values, successes and so on. Different spatial dimensions can describe the attributes that we are interested in; the locations in attribute space are then different states that we may perceive that we (or others) are in, while the distances can represent the time and effort, or value, of moving from one set of values of the attributes to another.

This spatial paradigm [14], which we can consider to be a hippocampus-type internal representation, is present in most things we do and certainly in many of our planning activities: we are aware of "where we are", "where we would like to be" and "the distance between the two", including the time and effort needed to bridge the two. In many organisms (e.g. dogs), the notion of self is also accompanied by markers, e.g. smell, that the animal recognises and actually uses to mark its physical space, and to self-recognise its own presence and also to detect intrusion.

Thus, the sensory system, and the notion of self and space are intertwined. In much simpler organisms, much of the self may actually be contained in the immediate chemical or physical (heat, electromagnetic radiation and pressure) environment. At the other end, in complex living organisms such as mammals, self-awareness will use a neuronal system, so that the physical embodiment of "self-awareness" will include a network of interacting and self-adaptive components and sub-systems [13, 15, 23].

23.3 Desirable Properties

A self-aware network should offer a distributed internal representation of itself, coupled with the ability to discover actions that it can take, in the form of paths to destinations and services, and having a "motor" capability for forwarding streams of packets along selected paths. The internal representation of the past and present experience of the network, based on sensing and measurement, with proactive sensing as one of the concurrent activities undertaken by the SAN, would include performance monitoring to provide an internal evaluation of how well the network is "doing its job" and condition monitoring to evaluate the health of the network. The coupling of the internal representation with motor control is obviously necessary for the network to evaluate when its information is insufficient or obsolete and then sense its environment, such as other networks and network users, or to probe itself for condition monitoring or performance evaluation. A SAN should offer a distributed representation of potential future actions and plans, critical evaluations of past actions and behaviours, and potential courses of action. The internal representation should allow the system to locate its components and behaviour both in physical space and in different virtual spaces that are relevant to the network. Useful virtual spaces can include aspects such as security, energy consumption, delay, loss of packets and computational overhead. Network nodes placed in this geography could then be identified via these different attributes: e.g. those nodes that are secure, those that function properly and those that are parsimonious in energy usage.

Thus, such attributes could be used to distinguish between different parts (nodes and links) or sub-areas of the network. This distributed representation should be coupled to the computer network's distributed "sensor system" which probes and measures the network's behaviour, and its "motor control" which forwards streams of packets from node to node along paths in the network, to desired destinations. A SAN should be able to sense and evaluate threats, and detect intrusions and attacks.

The capability for threat evaluation [2, 3] should be coupled with distributed motor reactions concerning packet routing driven by self-preservation and the need to assure quasi-normal operation, even in the presence of transient or sustained attacks.

23.4 Practical Approach

Since routing and the conveying of packets reliably from some input or source point S to some other outpoint or destination D is a computer network's key function, we now focus on a practical scheme for implementing a self-aware network that offers many of the capabilities we describe, based on the cognitive packet network (CPN) [8] protocol.

Addressing one by one the requirements of the previous section, we first note that in CPN, each network node maintains an internal representation of its primary role as a forwarder of traffic through the network. Specifically, for any source and destination (S-D) pair, and relevant quality of service (QoS) characteristic:

(a) either a node does not know which of its neighbours is the best choice as the next hop for this S-D pair and the specified QoS; but it can discover this information using smart packets (SP), which are distinct from the payload packets which the network is forwarding as part of its "useful job", or
(b) it does have this information based on previous experience, and the information is stored in an oracle that is implemented in the form of a neural network.

This distributed representation is also coupled with the network's "motor" capability to forward packets. When a packet with a given S-D and QoS requirement arrives to the network, then in case (a), the packet is forwarded as prescribed, while in case (b), a stream of "smart packets" (SP) is forwarded by the node, starting with its immediate neighbours, in a search for the path that currently has the best QoS metric for reaching D. SPs that reach D will send back an acknowledgement (ACK) packet, whose information content is used to update the oracles using reinforcement learning (RL) concerning this S-D at all the intermediate nodes, and at S. The oracles can then be used to forward subsequent packets. When a packet that travels from S to D does reach D, an ACK containing the hop-by-hop QoS experienced by the packet returns to the source together with time stamps concerning the packet going forward and the ACK coming back. Thus, a record of the paths that was explored, the dates and the observed QoS create an "autobiographical memory" of the SAN's experience, which includes its past performance and the conditions that were prevalent at the times when this experience was collected. The most recent data in this

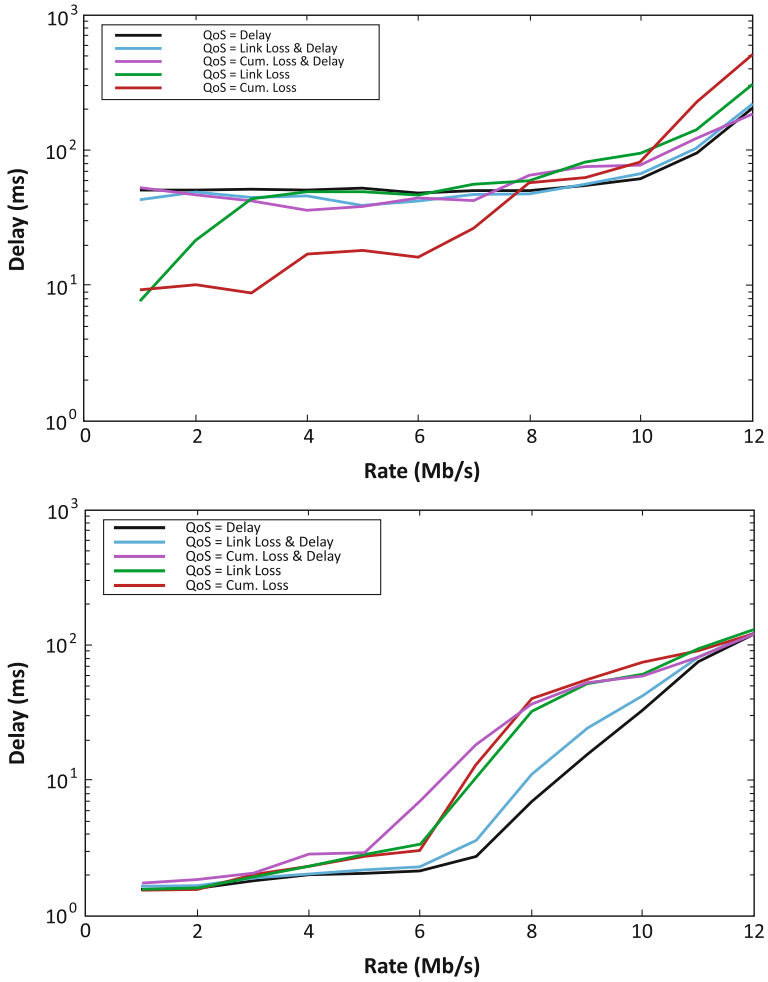

Fig. 23.1 The self-aware network observes its own performance and adapts itself to provide the users with the best possible QoS that the users have indicated. Thus, when the users specify "delay" as their desired QoS objective (*black*), the network achieves low delay (*top figure*) but is unable also to offer low loss rates (*bottom figure*). On the other hand, when low loss is desired (*red*), it is indeed achieved (*bottom*) but the delay may be higher (*top*)

autobiographical memory will be stored in the primary memory of the sources and nodes for rapid retrieval and for usage in decision making, while older elements may be moved to the secondary memory. SPs as well as payload packets that are sent out from a source and do not result in an ACK coming back also serve to estimate the packet loss rates and failures in the network (Fig. 23.1).

The QoS metrics used by the RL algorithm and by the SPs, and the information brought back by ACKs, can include different QoS characteristics that are either spec-

ified by network users as being relevant to themselves, or ingrained and relevant to the network's own performance criteria, such as delay, packet loss, jitter, security, energy consumption of nodes or links, or cumulative energy consumed by packets, or composite metrics that can be used to summarise several primary metrics. Essentially, any property that can be either directly measured (such as delay or energy consumption) or deduced from other direct measurements (such as jitter) may be used in CPN.

23.5 The Decision Engine

The decision engine or "oracle" in each of the CPN nodes is constituted by a random neural network [27, 28].

23.6 Avoiding Oscillations

When some node in a SAN makes a routing choice on the basis of its expectation that the outcome, e.g. in terms of QoS, will be favourable, it may not have the information concerning other nodes that might make the same choices based on similar information.

Thus, "best" decisions that are made independently at different entry nodes can then, some time after they are made and everyone tries to use the small number of initially best paths, become "bad" decisions when the quality of service perceived by subsequent users is themselves sensitive to the new traffic flows that use the previously identified "best" paths. This issue was also studied in terms of "sensitive" QoS metrics in previous work [12].

There are simple ways to avoid this from happening [13], such as randomisation over time, so that each user only switches to the best path in a manner which is random and hence less likely to synchronise across multiple users, or by the use of a performance threshold so that the user only makes path changes if it estimates that a significant performance improvement will occur.

Also, it is possible to introduce a random delay between the time that a decision is made and when it is actually applied, and furthermore, the node again senses the variables of interest to test whether the decision is sound. With time randomisation, the probability of a collision between different deciders [20] is negligible, similar to the way Ethernet operates using "carrier sensing" (Fig. 23.2).

Such schemes that exploit smart sensing and defer transmissions when interference, noise and other important channel characteristics can be handled in the best possible way, but also take the resulting delay, error and energy consumption into account, are also used to optimise various performance metrics in cognitive radio [9, 30].

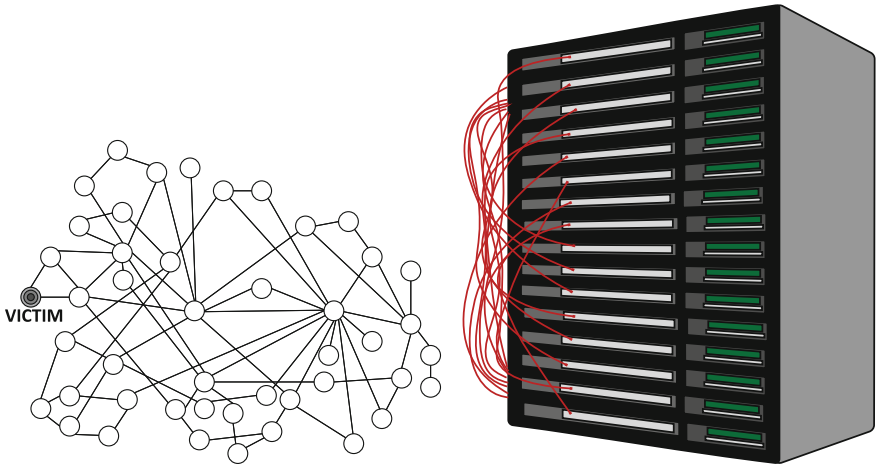

Fig. 23.2 Laboratory set-up for experiments on the self-aware network's defence capabilities to network attacks

23.7 Self-awareness and Network Attacks

Self-observation and coupled motor reaction carry over directly to the manner in which a SAN can detect attacks [6] and defend itself. These can be similar to attacks among living organisms: they are either very discrete, (1) trying to infiltrate "worms" or viruses which disable the network's useful activities, while letting through the attack itself which then delivers a "second" or "third" strike on the network's normal activities, or (2) they overwhelm the network's ability to handle traffic by an excessive number of requests that appear to be legitimate and may also in-filter into the network the attacking traffic of the first type (1).

Attacks of type (1) can require the SAN to conduct some form of eavesdropping or testing, which may or may not be allowed depending on the type of service the SAN has agreed to provide and the agreements with the users. Attacks of type (2) are usually detected without eavesdropping, using the observation of traffic characteristics. This reminds one of the local area networks [30], which react to changes in local network characterisics such as traffic in order to make decisions about whether to access the network, very similar to the way that a cognitive network operates (Fig. 23.3).

CPN offers the ability to monitor incoming traffic, to classify it into attack traffic, then counter-attack by destroying traffic that is identified as being part of an attack, and also storing it for offline analysis and comparison by deviating the attack traffic to a "honey pot". The honey pot may also be used to restore traffic which may have been mistakenly identified (i.e. a false alarm) as being part of an attack.

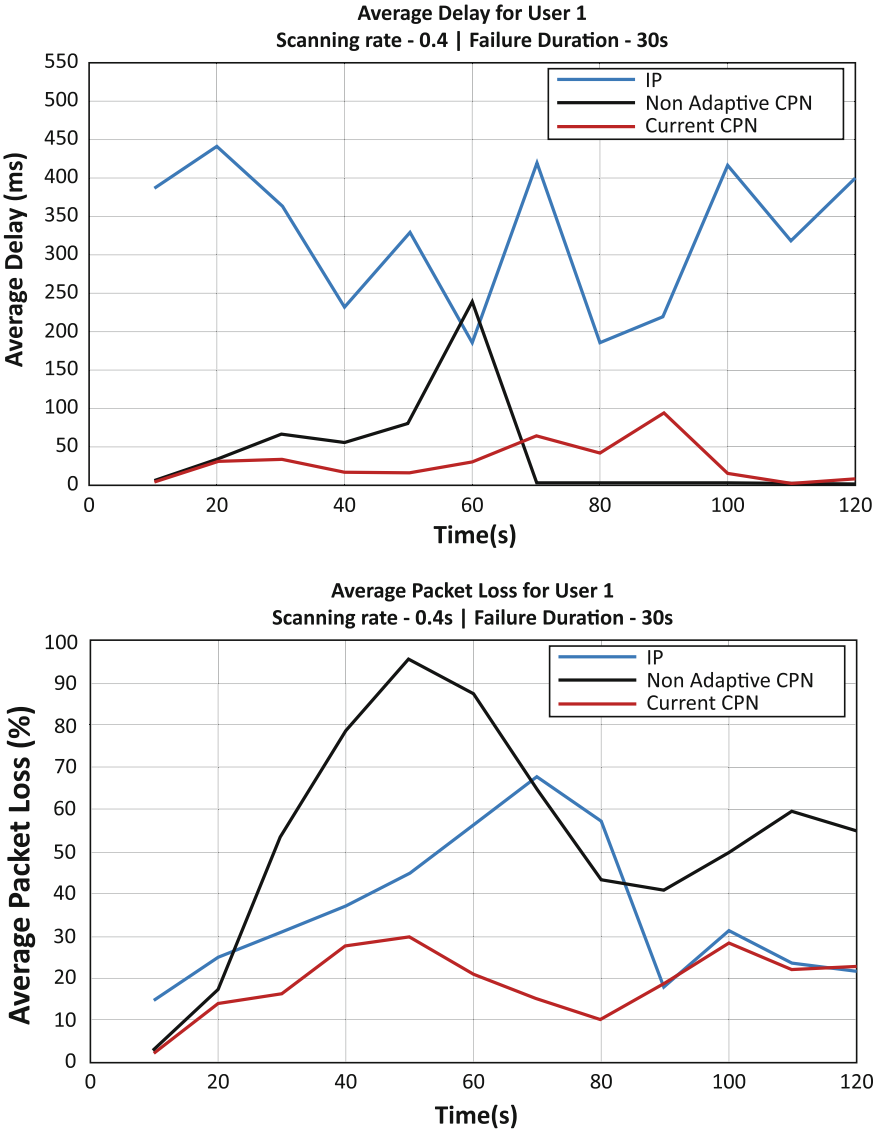

Fig. 23.3 Average packet delay (*top*) and average packet loss (*bottom*) experienced by one of the network users (the "victim") during a worm-based attack without (*blue* and *black*) and with (*red*) self-aware capabilities of the network

23.8 Conclusions

As packet networks become extremely large and diverse, a design based on a single view of how a network should operate becomes harder to justify [5]. Thus, each part of the network should be aware of its environment, its users, its resources, its performance and the threats that it is facing, and should use this awareness as an operational means to adapt its behaviour accordingly.

In addition, the possibility to adapt the network's algorithms and the software it uses, in response to changes in workload, updates in hardware infrastructure and network topology, should be incorporated into the network's core software. The software itself needs to be portable across different vendor hardware as well as various hardware generations. In addition to self-awareness and adaptivity, this creates a need for a software-defined network.

This paper summarises the work we have conducted over several years in an experimental setting for these ideas based on the cognitive packet network, which is a self-aware and completely implemented software-defined network, and reports on experiments with regard to specific choices and outcomes.

Acknowledgements The author thanks Esin Seref, Drs Pu Su, Peixiang Liu, Ricardo Lent, Arturo Nunez, Mike Gellman, Jrmie Lain, Laurence Hey, George Loukas, Georgia Sakellari, Omer H. Abdelrahman, Gokce Gorbil, Christina Morfopoulou and Ms Lan Wang for their contributions to various aspects of this research.

References

1. Jose Aguilar and Erol Gelenbe. Task assignment and transaction clustering heuristics for distributed systems. *Information Sciences*, 97(12):199 – 219, 1997. Load balancing in distributed systems.
2. Avgoustinos Filippoupolitis and Erol Gelenbe. A distributed decision support system for building evacuation. In *Proceedings of the 2Nd Conference on Human System Interactions*, HSI'09, pages 320–327, Piscataway, NJ, USA, 2009. IEEE Press.
3. Avgoustinos Filippoupolitis, Laurence Hey, Georgios Loukas, Erol Gelenbe, and Stelios Timotheou. Emergency Response Simulation Using Wireless Sensor Networks. In *Proceedings of the 1st International Conference on Ambient Media and Systems*, Ambi-Sys '08, pages 21:1–21:7, ICST, Brussels, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).
4. Alex Galis, Celestin Brou, Spyros Denazis, and Cornel Klein. *Programmable Networks for IP Service Deployment*. Artech House, Norwood, 2004.
5. E. Gelenbe and D.G. Chair. Users and services in intelligent networks. In *Next Generation Internet Networks*, pages 211–218, April 2005.
6. E. Gelenbe, M. Gellman, and G. Loukas. An autonomic approach to denial of service defence. In *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005.*, pages 537–541, June 2005.
7. E. Gelenbe and T. Kocak. Area-based results for mine detection. *IEEE Transactions on Geoscience and Remote Sensing*, 38(1):12–24, Jan 2000.
8. E. Gelenbe, R. Lent, and A. Nunez. Self-aware networks and QoS. *Proceedings of the IEEE*, 92(9):1478–1489, Sept 2004.

9. E. Gelenbe and B. Oklander. Cognitive users with useful vacations. In *2013 IEEE International Conference on Communications Workshops (ICC)*, pages 370–374, June 2013.
10. E. Gelenbe and S. Silvestri. Reducing power consumption in wired networks. In *24th International Symposium on Computer and Information Sciences, 2009. ISCIS 2009.*, pages 292–297, Sept 2009.
11. Erol Gelenbe. Probabilistic models of computer systems. *Acta Informatica*, 12(4):285–303.
12. Erol Gelenbe. Sensible decisions based on qos. *Computational Management Science*, 1(1):1–14.
13. Erol Gelenbe. Steps Toward Self-aware Networks. *Commun. ACM*, 52(7):66–75, Jul 2009.
14. Erol Gelenbe. Search in unknown random environments. *Phys. Rev. E*, 82:061112, Dec 2010.
15. Erol Gelenbe. Natural computation. *Comput. J.*, 55(7):848–851, 2012.
16. Erol Gelenbe and Yonghuan Cao. Autonomous search for mines. *European Journal of Operational Research*, 108(2):319 – 333, 1998.
17. Erol Gelenbe and Jean-Michael Fourneau. Random Neural Networks with Multiple Classes of Signals. *Neural Comput.*, 11(4):953–963, May 1999.
18. Erol Gelenbe, Khaled Hussain, and Varol Kaptan. Simulating autonomous agents in augmented reality. *Journal of Systems and Software*, 74(3):255 – 268, 2005.
19. Erol Gelenbe and George Loukas. A Self-aware Approach to Denial of Service Defence. *Comput. Netw.*, 51(5):1299–1314, Apr 2007.
20. Erol Gelenbe and Isi Mitrani. Control Policies in CSMA Local Area Networks: Ethernet Controls. *SIGMETRICS Perform. Eval. Rev.*, 11(4):233–240, August 1982.
21. Erol Gelenbe and Christina Morfopoulou. A Framework for Energy-Aware Routing in Packet Networks. *Comput. J.*, 54(6):850–859, Jun 2011.
22. Erol Gelenbe, Nestor Schmajuk, John Staddon, and John Reif. Autonomous search by robots and animals: A survey. *Robotics and Autonomous Systems*, 22(1):23–34, 1997. Biologically Inspired Autonomous Systems.
23. Erol Gelenbe and K. Sevcik. Analysis of Update Synchronization for Multiple Copy Data Bases. *IEEE Transactions on Computers*, C-28(10):737–747, Oct 1979.
24. Erol Gelenbe, Mert Sungur, Christopher Cramer, and Pamir Gelenbe. Traffic and video quality with adaptive neural compression. *Multimedia Systems*, 4(6):357–369.
25. Erol Gelenbe and Stelios Timotheou. Random Neural Networks with Synchronized Interactions. *Neural Computation*, 20(9):2308–2324, 2008.
26. Gokce Gorbil and Erol Gelenbe. Opportunistic communications for emergency support systems. *Procedia Computer Science*, 5:39–47, 2011. The 2nd International Conference on Ambient Systems, Networks and Technologies (ANT-2011)/The 8th International Conference on Mobile Web Information Systems (MobiWIS 2011).
27. L.A. Hey, P.Y.K. Cheung, and M. Gellman. FPGA based router for cognitive packet networks. In *Field-Programmable Technology, 2005. Proceedings. 2005 IEEE International Conference on*, pages 331–332, Dec 2005.
28. Laurence A. Hey. Reduced complexity algorithms for cognitive packet network routers. *Computer Communications*, 31(16):3822–3830, 2008. Performance Evaluation of Communication Networks (SPECTS 2007).
29. Pedro Martinez-Julia, Antonio F. Skarmeta, and Alex Galis. *The Future Internet: Future Internet Assembly 2013: Validated Results and New Horizons*, chapter Towards a Secure Network Virtualization Architecture for the Future Internet, pages 141–152. Springer Berlin Heidelberg, 2013.
30. Boris Oklander and Erol Gelenbe. *Information Sciences and Systems 2013: Proceedings of the 28th International Symposium on Computer and Information Sciences*, chapter Optimal Behaviour of Smart Wireless Users, pages 87–95. Springer International Publishing, 2013.