DYNAMIC CONTROL OF MOBILE AD-HOC NETWORKS
Network Protocol Parameter Adaptation using Organic Network Control

Sven Tomforde, Björn Hurling and Jörg Hähner
Institute of Systems Engineering, Leibniz Universität Hannover, Appelstr. 4, 30167 Hannover, Germany

Keywords: Organic computing, Self-optimising data communication protocols, Framework for protocol control, Mobile ad-hoc networks.

Abstract: Data communication protocols show an increasing complexity in terms of variable configurations – especially if their target execution area is highly dynamic. One domain that represents these characteristics are mobile ad-hoc networks (MANets). Since nodes are moving, the situation surrounding a particular node is steadily changing. This provides the opportunity to significantly increase the system's performance by continuously adapting the protocol. This paper demonstrates the benefit of such an adaptation using the Organic Network Control (ONC) system. Based upon a brief overview of ONC, the adjustment of the framework to enable the control of MANets is described, followed by a simulation-based evaluation using an exemplary broadcast protocol.

1 INTRODUCTION

As the number of interconnected devices and the corresponding transfer load of data communication networks is steadily increasing, networks reach their limits. This leads to the insight that the currently used techniques (e. g. protocols and infrastructure) will not be able to cope with the demand in the near future (Handley, 2006). Based upon this assumption, researchers develop new concepts (e. g. for the Internet (Siekkinen et al., 2007)). To cope with the problem, two approaches are possible: Develop new protocols (with higher complexity in terms of more parameters) and increase the quality of service or dynamically adapt existing protocols to changing environments, which is done by the Organic Network Control (ONC) system (introduced in (Tomforde et al., 2009a; Tomforde et al., 2009b)).

The ONC system is based on the principles of Organic Computing (Schmeck, 2005) which is a recent research area focusing self-organisation to deal with complex problems. Autonomous entities are acting without strict central control and achieve global goals although their decisions are mainly based on local knowledge. The authors assume that due to the complexity of the particular tasks not all situations can be foreseen during the development process of the system. Therefore, the system must be adaptive and equipped with learning capabilities, which leads to the ability to learn new actions and strategies for previously unknown situations. The self-control of network entities is also part of the focus of Autonomic Computing (Kephart and Chess, 2003).

A demanding challenge for the ONC system is the control and adaptation of mobile ad-hoc network (MANet) protocols as they are processed in highly dynamic environments. The possible movement of nodes leads to a continuous change of the situation: Neighbours are getting out of reach or joining the sending distance. This does not only lead to complex problems of how to configure the protocol, it also offers high potential for an improvement of the system performance. Within this paper, we explain how the ONC system is applied to a MANet broadcast protocol and how the overall performance of the MANet system can be increased using ONC.

This paper demonstrates the application of the ONC system to MANet-based broadcast protocols by dynamically adapting network protocol parameters (e. g. values for timeouts, maximum number of re-transmissions, number of open connections, etc.). Section 2 describes the related work and gives a summary of approaches to adapting network protocols dynamically to changing environments, followed by an overview of the ONC system in Section 3. In Section 4, we explain how the ONC system can be applied to a new protocol and what is actually done to enable MANet protocol control using ONC. Af-

Tomforde S., Hurling B. and Hähner J. (2010).
DYNAMIC CONTROL OF MOBILE AD-HOC NETWORKS - Network Protocol Parameter Adaptation using Organic Network Control.
In *Proceedings of the 7th International Conference on Informatics in Control, Automation and Robotics*, pages 28-35

terwards, Section 5 demonstrates the benefit of using ONC by explaining and analysing the achieved results. Finally, Section 6 summarises the presented system and names further research to be done.

2 RELATED WORK

Autonomic adaptation of networks is part of the focus of Autonomic Computing (Kephart and Chess, 2003). Researchers have considered the problem to adapt network protocols for many years: from off-line optimisation when presenting a new protocol to adapting protocol configurations during runtime. For the off-line optimisation, several examples can be found in literature, but none aims at providing a generalised approach for more than one specific protocol (see, e.g., (Montana and Redi, 2005; Sözer et al., 2000; Turgut et al., 2002)). Due to time and computational restrictions, on-line adaptation is a more complex task compared to the off-line part. Besides the ONC approach, different directions of research are known to cope with the problem: adaptive protocols, composition of protocol stacks, or centralised solutions to adapt protocol configurations.

The most obvious way of dealing with the problem is to develop adaptive protocols which are able to handle dynamic environments. One example has been presented in (Whiteson and Stone, 2004). They introduced an on-line learning mechanism to increase the performance of a routing protocol. Based on the Q-routing techniques presented in (Boyan and Littman, 1994), they learn the best routes by receiving immediate answers of the next hop. Another example has been introduced in (Huang et al., 2009). They present an adaptive medium access control (MAC) protocol framework. Since the radio node density and service requirements can vary widely over time, they defined the need of an adaptation to changing environments and needs. Their protocol prototype can switch between CSMA and TDMA within a radio platform scenario. Both approaches rely on the existence of a protocol extension covering the learning/adaptation information. In contrast to the ONC system, they are system-specific solutions and cannot be applied to other protocols.

Since developing new protocols for all possible adaptation and learning processes is not feasible, a research field called *protocol stack composition* emerged covering the upcoming tasks by exchanging protocols and stacks dynamically (Rosa et al., 2007). In contrast to the ONC system which keeps the existing and currently used techniques and optimises their behaviour, this field of research re-combines the protocols. Although the target deviates from the ONC approach (the protocol stack exchange has impact on all involved systems and can hardly be done locally), the approach has some similarities. The most important representatives are *Appia* (Miranda et al., 2001), *Cactus* (Hiltunen et al., 2000), *Ensemble* (van Renesse et al., 1998), and *Horus* (van Renesse et al., 1996). Additionally, the recent work done by *Mena et al.* (Mena et al., 2003) has to be named. Besides the locality aspect, some characteristics of the approach separate it from the requirements of the ONC framework: the protocols and their configurations have to be known in advance and further extensions with new behavioural repertoire are not possible.

The approach presented in (Schöler and Müller-Schloer, 2004) is also dealing with a kind of protocol composition, but is already a bit more focused on the techniques used within the ONC system. The authors describe their adaptive monitoring architecture for protocol stack configuration and demonstrate the integration in the Observer/Controller pattern of Organic Computing. The learning part is covered by a Fuzzy Learning Classifier System (Casillas et al., 2004). Due to the usage of the same architectural pattern (Observer/Controller), the approach has some similarities with the ONC system. Unlike the ONC framework, the approach is built again without offering the opportunity of handling different protocols and extending the set of solutions on demand.

On-line adaptation of protocols itself has been focused by researchers before. (Sudame and Badrinath, 2001) presents a first TCP- and UDP-based study and defined the need of dynamic adaptation, but detailed examination and a demonstration of the re-usability for other protocols is still missing. Currently, there exist only two approaches covering a similar target as ONC: the systems introduced by *Ye et al.* and by *Georganopoulos and Lewis*. The former one (Ye and Kalyanaraman, 2001) introduces an adaptive random search algorithm which tries to combine the stochastic advantages of pure random search algorithms with threshold-based knowledge. Their approach is based on the initial system as presented in (Ye et al., 2001). In contrast to our approach, *Ye et al.* propose a centralised system that tackles the optimisation task for each node. To allow for such a division of work between a central server and the particular network nodes, problems like e. g. bandwidth usage, single point of failure, or local knowledge accessible from server-side have to be covered.

The second system has been presented in 2007 (Georganopoulos and Lewis, 2007) and introduces a dynamic optimisation framework for the reconfiguration of network protocols at all layers of the protocol

stack. In order to optimise the performance of the system depending on given goals, different entities can be adjusted (applications, protocols, etc.) or replaced. Again, the system relies mainly on a centralised element being responsible for the optimisation tasks. The focus of the initial paper has been set on cross-layer optimisation for the protocol stack, but less on considering environmental conditions. Hence, the authors demonstrated the performance of the solution by applying it to two different layers of the protocol stack: the link and the network layer. A detailed proof of the approach and insights on the currently vague blackbox *dynamic optimisation engine* are still missing, consequently a suitability of the approach cannot be estimated – although some criteria (centralised element, low re-usability of existing protocols, etc.) are contradicted for the ONC requirements.

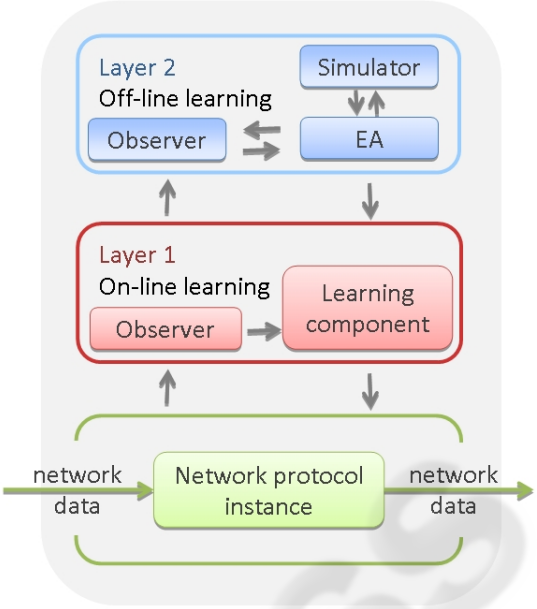

Figure 1: Architecture of the ONC system.

3 THE ORGANIC NETWORK CONTROL SYSTEM

The Organic Network Control (ONC) system has been introduced in (Tomforde et al., 2009a; Tomforde et al., 2009b). The system's architecture is founded on the generic Observer/Controller approach as presented by *Richter et al.* in (Richter et al., 2006) and is organised using three consecutive layers, see Figure 1.

Layer 0 encapsulates an existing network protocol instance, e.g. a broadcast algorithm for mobile ad-hoc networks (MANets) or a Peer-to-Peer (P2P) protocol. In terms of Organic Computing, this controlled network protocol instance is the "System under Observation and Control" (SuOC). The ONC system aims at providing a basic solution to control existing protocols dynamically without the need of knowing internals of the particular protocol or interfering with the protocol logic. However, it is required that the parameters of the protocol can be altered by the ONC system. Additionally, the current status of the protocol instance and the environment it acts in have to be observable and accessible locally. For MANets, one of the most important factors describing the current status of the protocol instance's environment is the neighbourhood of other nodes. Besides this observable environment, a performance measure (also called fitness or evaluation function) quantifying good and bad performance has to be provided in order to evaluate the current performance of the protocol.

Layer 1 of the ONC architecture aims at adapting the SuOC dynamically to changes in the environment. It therefore consists of two basic components: an Observer and a Controller containing a machine learning component (which is currently realised as a Learning Classifier System (LCS)). The Observer is responsible for locally collecting status information about the network protocol instance and its settings. Additionally, it aggregates the observed figures and augments them with optional further knowledge (e.g. prediction values, historic knowledge, etc.) and builds a vector describing the current situation at the node. This situation vector then serves as input to the Controller which has to fulfill two tasks: evaluate the system's performance within the last evaluation cycle and decide about the next action to be taken. The main component of the Controller is a LCS which is realised as an adapted variant of Wilson's XCS (Wilson, 1995). The LCS is responsible for choosing the next action – based on the situation vector. The result of the selection process is an action (a configuration of parameter sets for the controlled protocol) and a prediction value of how the system will perform after applying the action. To evaluate the system's performance, the Controller compares the system's performance measure (the fitness function) with the last prediction and calculates the reward for the LCS to enable the automated learning. In case the LCS does not contain a matching parameter set, new classifiers need to be created. In contrast to the original LCS algorithm, however, the ONC architecture does not allow new classifiers (pairs of situation/conditions and parameters/actions) to be created randomly by Genetic Algorithms. Instead, control is transferred to Layer 2 of the ONC architecture.

Layer 2 of the ONC system is again designed us-

ing the Observer/Controller pattern: The Observer monitors the Layer 1 component and realises the need of a new classifier, it therefore receives the situation vector. The Controller part contains two basic components: a simulator and an Evolutionary Algorithm (EA). The Controller creates an appropriate simulation scenario from the situation vector and triggers the EA to repeatedly evolve a number of parameter sets for the network protocol. These parameter sets are evaluated in the simulator. This bears the advantage that newly created parameter sets are not directly used in the live system, as this can cause the system to perform badly or even malfunction. Only those parameter sets that qualify in the simulator of Layer 2 are passed back to Layer 1 and may then be applied in the real world. Therefore, Layer 2 allows for a kind of *sandbox*-learning without the risk of applying arbitrary parameter sets to the live system.

The ONC approach as described before provides a black-box solution to control different types of network protocols. In order to integrate a new protocol into the ONC architecture and consequently enable ONC controlling the protocol, an engineer has to fulfill three major tasks: Specify the performance metric, describe the situation (what are the dynamic factors defining the need of an adaptation, e.g. available neighbours and their positions in MANets) accompanied by a distance function between two situations, and provide a simulation model to enable the simulation-based optimisation process of Layer 2. Within the following Section, we describe how the ONC system is applied to MANet protocols.

4 DYNAMIC CONTROL OF MOBILE AD-HOC NETWORKS

This Section describes how ONC is adapted to allow for the control of MANet protocols - based on the tasks named within the last Section. In order to keep the same organisation as before, this Section again distinguishes between the three layers and describes what has to be done on each layer.

The focus of Layer 0 is to integrate a new protocol into the framework. Therefore, the engineer has to describe its observation and control process leading to the need of two interfaces: one for accessing the protocol parameters and one for collecting information about the local status of the system. The former interface enables the framework to adapt the behaviour of the protocol which means the parameter settings can be adapted at runtime. In the latter interface, the engineer has to define what is relevant and influences the protocol's performance - we call this the *situation* the system is in. In a MANet environment, the most important factor influencing the protocol's performance is the distribution of other nodes within its sending and sensing range. Therefore, a sector-based approach as depicted in Figure 2 has been developed. The radius of the outer circle is equal to the sensing distance (*sensDist*) of the node, as this is the most remote point where messages of this node can interfere with other ones. Typically, the transmission range for Wireless-LAN based MANets is about 250 meter (half of the sensing distance). The radii of the inner circles have been chosen empirically.

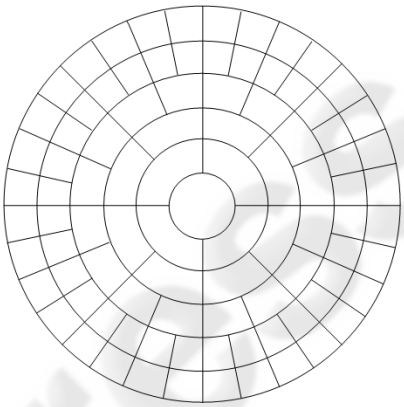

Figure 2: Environment representation.

As nodes within the first circle are really close (50m), their exact position does not matter. The second circle (125m) has been partitioned into 4 sectors, the third circle (200m) into 8 sectors, and the forth circle (250m, maximum transmission range) into 16 sectors. The next two circles (375m and 500m) are representing the area within sensing range - with both circles divided into 32 sectors each. We assume that a node is able to determine the current positions of its neighbours in sensing range relative to its own position (e. g. based on GPS, see (Pahlavan and Krishnamurthy, 2001)). Additionally, the node's direction of movement is needed since it has high influence on the best parameter set (e. g. moving towards/away from a set of nodes influences the delay). Due to the sector-based approach, situations are generalised which is necessary to avoid evolving a rule for each situation.

The Layer 1 component is responsible for the adaptation process and for increasing the system's performance by learning. Again, two aspects have to be considered: a learning feedback and a measurement to compare different situations. To enable the learning feedback, a fitness or evaluation function is needed. Several fitness functions have been proposed for MANet protocols. Since our current focus is set on MANet-based broadcast algorithms, the standard functions are *Packet Delivery Ratio* and *Packet La-*

tency – both cannot be measured locally at each node. Based upon the locally available information, the target is to reduce the number of forwarded broadcasts and assure the delivery of the broadcast to each node at the same time. Therefore, we introduce the following formula:

$$Fit(x) = \frac{\#RecMess}{\#FwMess}$$

The variable x stands for the currently observed network protocol instance. Since a new parameter set has to be applied for a minimum duration to show its performance, we use *evaluation cycles* defining discrete time slots. The duration of the cycle depends on how dynamic an environment is: The faster it changes, the shorter is the cycle to be chosen. For the last evaluation cycle, the function takes into account the sum of all messages being forwarded by all of the neighbours and the node to be evaluated within the last evaluation cycle ($\#FwMess$), and the sum of all messages being received by them ($\#RecMess$).

The second aspect on Layer 1 is the comparison of situations: we need to quantify the distance of two situations. The target is that more similar situations will receive a low distance value and those having low similarity will receive a high distance value. Based upon the sector-based situation description as introduced before, a measure for the similarity of two entities (A, B) can be defined. To be able to determine the distance, the possible influence of rotation and reflection are deducted initially. Afterwards, the formula for the distance (δ) can be defined with $r \in RADII$ and $s \in SECTORS$ as follows:

$$\delta(A,B) = \sum_r \sum_s (A_{r,s} - B_{r,s})^2 / r.distance$$

The function *r.distance* defines the radius size as introduced before (50m, 125m, ...). $A_{r,s}$ gives the number of neighbours within the sector s of radius r for the situation description A. This means that the importance of a node's neighbour decreases if it is situated within an outer radius.

Finally, Layer 2 has to be able to build adequate simulation scenarios out of the information obtained by Layer 1. In ONC, we use the standard network simulation tool *NS-2* (Fall, 1999), but this can easily be exchanged by other solutions. The network simulator *NS-2* has a large set of integrated or available standard protocols, but for recently developed or proprietary protocols a simulation model probably does not exist. The engineer has to provide a realistic model (as it is also used during the protocol development) which can be adapted to the observed situation by generating an appropriate scenario. The adaptation of the scenario is done using the configuration interface by considering the observed situation of the node, e.g., a randomised instance of the sector-model is created defining the distribution of the neighbouring nodes and the movement direction of the node is transmitted to *NS-2* using the same coordinate system as for the observed system. After finishing the previously described tasks, ONC is able to control MANet-based protocols. The benefit of the dynamic control is demonstrated in the next Section.

5 EVALUATION

Based upon the previously described adaptations, ONC is able to control MANets. Within this Section, the results of the evaluation are presented. Since the adaptation of protocols is organised locally - but has high influence on the network-level, both aspects are taken into consideration. Therefore, we start with the experimental setup, followed by a short introduction of the analysed protocol, and conclude with the achieved results for the local and the network-wide view.

5.1 Experimental Setup

The ONC framework is implemented in JAVA. The moving agents communicating via the MANet protocol are simulated using the Multi-Agent Simulation Toolkit *MASON* (Luke et al., 2004), with each agent's protocol instance representing a SuOC of the architecture as depicted in Figure 1. The respective Layer 1 Controller is an adapted Learning Classifier System as described in (Tomforde et al., 2009b). At Layer 2, the standard network simulation tool *NS-2* (Fall, 1999) is used to evolve new parameter sets in combination with a standard Genetic Algorithm (population size: 15, new children per iteration: 7, mutation rate: 0.2 per child, all children via crossover with fitness-based selection of parents). We use two different simulation tools in order to avoid having exactly the same conditions while optimising rules, since a complete copy of the current situation observed in the real environment within the simulator is not realistic. 100 agents have been created and applied to the simulated area, which has dimensions of 1000 x 1000 cells (corresponds to 1000 x 1000 meters). The agents move according to a random-waypoint-model. The Physical/Mac layer is an IEEE 802.11 in ad-hoc mode at 2 Mbps.

To demonstrate the performance of ONC controlling MANets, we choose the *Reliable Broadcast Protocol* (R-BCast) as introduced in (Kunz, 2003), since

this protocol is representative for the research field of reliable broadcast protocols in MANets. In order to achieve reliability and increase the packet delivery ratio compared to other protocols, additional effort is made by equipping the nodes with extra buffers. These round-robin based buffers are used to store the last p unique packets the particular nodes received. In contrast to other protocols, the R-BCast protocol has significantly more variable parameters and consequently the task to control the protocol is more complex, but it also offers a higher potential benefit due to a dynamic adaptation. The parameters being subject to ONC control actions are: **Delay** (Maximum deceleration time between receiving and forwarding of a message), **Allowed Hello-loss** (Maximum number of Hello-messages, which may be lost until a node is assumed to be out of transmission range), **HelloInterval** (Interval between two Hello-messages), δ **Hello-Interval** (Randomises Hello-Interval), **Packet count** (Number of the last x stored NACK messages), and **Minimum difference** (Minimum difference between NACK messages). Details on the parameters and the protocol can be found in (Kunz, 2003).

5.2 Experimental Results

In order to analyse the performance of the ONC system, the simulation is repeated for two cases under the same restrictions and using the same seeds for the randomised values: a) all nodes are uncontrolled (no ONC system) and use the manually optimised standard configuration of the protocol, and b) all nodes have an own instance of the ONC system to control their protocol configuration. All values presented in the remainder of this Section are averaged values received from three runs, where each run has a duration of 10,000 simulated seconds. During one run of the scenario, 17,400 BCast-messages have been simulated. The learning component has been trained using 10 complete runs with different seeds – leading to completely different movements of the nodes and along with these to different situations.

The performance measurement relies on the fitness function as described for the local feedback mechanism of the learning component (Section 4). Since the evaluation takes both views into account (local and network-wide view), the x in $Fit(x)$ refers to different systems: a) In the local view x stands for the local network protocol instance of the node and b) on network-level obtained for reference, x represents the set of protocol instances within the network and gives an averaged value for all instances.

Figure 3 plots the system's performance considering only one node. The X-axis describes the simulation time (in simulated seconds) and the Y-axis the measured fitness value. In principle, all simulated nodes show a comparable behaviour; this specific node has been explicitly chosen to demonstrate the typical differences between an ONC-controlled and an uncontrolled node. During the simulation, the node gets separated from the rest of the network (no other nodes within *sending* distance) between simulation seconds 7,350 and 7,700. Within this interval, the fitness is 0 for both cases. But especially these situations demonstrate the benefit of ONC control: The delays have been lengthened so that the node receives more *old* messages when it arrives back in sending distance of another node resulting in a quicker recovery of the ONC-controlled system.

Another observation that can be made considering Figure 3 is the impact of the learning module. To be able to learn, it has to be allowed to try different rules and not to use always the best matching one. E.g. at simulation second 1,800, the learning component tried a rule that results in a performance slightly worse than the standard protocol configuration. These small drawbacks have to be taken into account to achieve an improvement for the system. Averaged over the complete simulation time (10,000 simulated seconds), the performance of the protocol instance has been enhanced in terms of the fitness function from 0.8270 (all nodes perform the standard protocol without any adaptation) to 0.8991 which is an increase of 8.71%.

Figure 4 depicts the averaged performance of the network protocol instances on network-level. The averaging leads to the effect that separation of single nodes influences the performance only slightly. Nevertheless, Figure 4 shows two drops (simulation seconds 3,850 to 4,200 and 7,450 to 7,550). The first drop can be explained by a split of the set of nodes – about 30 nodes are not within sending distance of the rest. Here, two different networks have been established. Within the second drop, again, a larger group (18 nodes) has been separated from the rest of the network. Despite these separation effects, the performance of the system has been increased. When all nodes perform just the standard protocol configuration without any adaptation, the resulting averaged fitness is 0.8760. The same simulation with additional ONC control for all nodes leads to an averaged fitness value of 0.9456 which is an increase of 7.94%. Both aspects of the fitness function are responsible for the increase: The number of forwarded messages has been decreased slightly, whereas the number of received messages has been increased more significantly.

Hiltunen, M. A., Schlichting, R. D., Ugarte, C. A., and Wong, G. T. (2000). Survivability through customization and adaptability: the cactus approach. In *Proc. of DARPA Information Survivability Conference and Exposition (DISCEX'00)*, volume 1, pages 294 – 307.

Huang, K.-C., Jing, X., and Raychaudhuri, D. (2009). Mac protocol adaptation in cognitive radio networks: An experimental study. *Int. Conf. on Computer Communications and Networks*, 0:1–6.

Kephart, J. O. and Chess, D. M. (2003). The Vision of Autonomic Computing. *IEEE Computer*, 36(1):41–50.

Kunz, T. (2003). *Reliable Multicasting in MANETs*. PhD thesis, Carleton University.

Luke, S., Cioffi-Revilla, C., Panait, L., and Sullivan, K. (2004). MASON: A New Multi-Agent Simulation Toolkit. In *Proc. of the 2004 Swarmfest Workshop*.

Mena, S., Schiper, A., and Wojciechowski, P. (2003). A step towards a new generation of group communication systems. In *Proceedings of the International Conference on Middleware*, pages 414–432. Springer-Verlag, New York.

Miranda, H., Pinto, A., and Rodrigues, L. (2001). Appia: A flexible protocol kernel supporting multiple coordinated channels. In *Proceedings of the 21st Int. Conf. on Distributed Computing Systems (ICDCS'01)*.

Montana, D. and Redi, J. (2005). Optimizing parameters of a mobile ad hoc network protocol with a genetic algorithm. In *Proc. of the Conf. on Genetic and Evolutionary Computation (GECCO'05)*, pages 1993–1998.

Pahlavan, K. and Krishnamurthy, P. (2001). *Principles of Wireless Networks: A Unified Approach*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Richter, U., Mnif, M., Branke, J., Müller-Schloer, C., and Schmeck, H. (2006). Towards a generic observer/controller architecture for Organic Computing. In *GI Jahrestagung (1)*, pages 112–119.

Rosa, L., Rodrigues, L., and Lopes, A. (2007). Appia to r-appia: Refactoring a protocol composition framework for dynamic reconfiguration. Technical report, Department of Informatics, University of Lisbon.

Schmeck, H. (2005). Organic Computing – A new vision for distributed embedded systems. In *Proc. of the 8th Int. Symp. on Object-Oriented Real-Time Distributed Computing (ISORC'05)*, pages 201–203.

Schöler, T. and Müller-Schloer, C. (2004). Design, implementation and validation of a generic and reconfigurable protocol stack framework for mobile terminals. In *Proc. of the 24th Int. Conf. on Distributed Computing Systems (ICDCSW'04)*, pages 362–367.

Siekkinen, M., Goebel, V., Plagemann, T., Skevik, K.-A., Banfield, M., and Brusic, I. (2007). Beyond the Future Internet–Requirements of Autonomic Networking Architectures to Address Long Term Future Networking Challenges. *Future Trends of Distributed Computing Systems, IEEE International Workshop*, pages 89–98.

Sözer, E. M., Stojanovic, M., and Proakis, J. G. (2000). Initialization and routing optimization for ad-hoc underwater acoustic networks. In *Proc. of Opnetwork'00*.

Sudame, P. and Badrinath, B. R. (2001). On providing support for protocol adaptation in mobile wireless networks. *Mob. Netw. Appl.*, 6(1):43–55.

Tomforde, S., Cakar, E., and Hähner, J. (2009a). Dynamic Control of Network Protocols - A new vision for future self-organised networks. In *Proc. of the 6th Int. Conf. on Informatics in Control, Automation, and Robotics (ICINCO'09)*, pages 285 – 290.

Tomforde, S., Steffen, M., Hähner, J., and Müller-Schloer, C. (2009b). Towards an Organic Network Control System. In *Proceedings of the 6th International Conference on Autonomic and Trusted Computing (ATC'09)*, pages 2 – 16. Springer Verlag.

Turgut, D., Daz, S., Elmasri, R., and Turgut, B. (2002). Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach. In *Proc. of the IEEE Global Telecommunications Conference (GLOBECOM '02)*, pages 62 – 66.

van Renesse, R., Birman, K., Hayden, M., Vaysburd, A., and Karr, D. (1998). Building adaptive systems using ensemble. *Softw. Pract. Exper.*, 28(9):963–979.

van Renesse, R., Birman, K. P., and Maffeis, S. (1996). Horus: a flexible group communication system. *Communications of the ACM*, 39(4):76–83.

Whiteson, S. and Stone, P. (2004). Towards autonomic computing: adaptive network routing and scheduling. In *Proceedings of the International Conference on Autonomic Computing (ICAC'04)*, pages 286–287.

Wilson, S. W. (1995). Classifier fitness based on accuracy. *Evolutionary Computation*, 3(2):149–175.

Ye, T., Harrison, D., Mo, B., Sikdar, B., Kaur, H. T., Kalyanaraman, S., Szymanski, B., and Vastola, K. (2001). Network Management and Control Using Collaborative On-line Simulation. In *Proceedings of IEEE ICC*, Helsinki, Finland. IEEE.

Ye, T. and Kalyanaraman, S. (2001). An adaptive random search algorithm for optimizing network protocol parameters. Technical report, Rensselaer Polytechnic Inst.