

DES-Chan: A Framework for Distributed Channel Assignment in Wireless Mesh Networks

Felix Juraschek Mesut Güneş Matthias Philipp Bastian Blywis Oliver Hahm
Distributed Embedded Systems - Institute of Computer Science
Freie Universität Berlin and Humboldt University Berlin, Germany
{jurasch, guenes, mphilipp, blywis, hahm}@inf.fu-berlin.de

Abstract—In this paper we present the DES-Chan framework for experimentally-driven research on distributed channel assignment algorithms in wireless mesh networks. The implementation process of channel assignment algorithms is a difficult task for the researcher since common operating systems do not support channel assignment algorithms out of the box. DES-Chan provides a set of common services required by distributed channel assignment algorithms. The modular architecture of DES-Chan allows the extensions with further modules or the modifications of existing ones. As a proof of concept, we present a reference implementation of a distributed greedy channel assignment algorithm. We evaluate its performance in the DES-Testbed, a multi-transceiver wireless mesh network (WMN) with 100 nodes at the Freie Universität Berlin.

Index Terms—distributed channel assignment, wireless mesh network (WMN), testbed, experimentation

I. INTRODUCTION

Channel assignment for multi-transceiver *wireless mesh networks* (WMNs) attempts to increase the network performance by decreasing the interference of simultaneous transmissions. Multi-transceiver mesh routers allow the communication over several wireless network interfaces at the same time. However, this can result in high interference of the wireless interfaces leading to a low network performance. With channel assignment, the reduction of interference is achieved by exploiting the availability of fully or partially non-overlapping channels. Channel assignment can be applied to all wireless networks based on technologies that provide non-overlapping or orthogonal channels. Currently, wide-spread technologies are IEEE 802.11a/b/g, IEEE 802.11n, and IEEE 802.16 (WiMAX). With the low cost for IEEE 802.11 hardware, the number of deployments based on this technology is increasing and channel assignment algorithms are gaining in importance.

Although channel assignment is still a young research area, many different approaches have already been developed [15]. These approaches can be distinguished into *centralized* and *distributed* algorithms. Centralized algorithms rely on a central entity, usually called *channel assignment server* (CAS), which calculates the network-wide channel assignment and sends the result to the network nodes. In distributed approaches, each node calculates its channel assignment based on local information. Distributed approaches can react faster to topology changes due to node failures or mobility and usually introduce less protocol overhead, since communication with the CAS is not necessary. As a result, distributed approaches are more

suitable once the network is operational and running. Another classification considers the frequency of channel switches on a network node. In *fast channel switching* approaches, channel switches may occur frequently, in the extreme for every subsequent packet a different channel is chosen. The limiting factor for such algorithms is the relative long channel switching time with commodity IEEE 802.11 hardware, which is in the order of milliseconds. *Slow channel switching* approaches switch the interfaces to a particular channel for a longer period, usually in the order of minutes or hours. *Hybrid* approaches combine both methods.

The focus of this paper is on the experimentally driven research of distributed, slow channel switching algorithms on wireless testbeds. This process yields several challenges and pitfalls because the researcher has to deal with operating system specifics, drivers for the wireless interfaces, and the capabilities and limitations of the particular hardware. If more than one particular algorithm should be studied, the same problems and services have to be addressed multiple times because algorithms of this domain often require a set of common services. Among them are interface management, message exchange for node-to-node communication over the wireless medium, and data structures for network and conflict graphs. Additionally, a research framework for channel assignment algorithms can speed up the development process significantly by using the already available services.

The contribution of this paper is the presentation of DES-Chan, a framework for distributed channel assignment in multi-transceiver WMNs that provides these services. We analyze several algorithms for distributed channel assignment and derive the required services for their implementation. We describe the architecture of DES-Chan, that provides these services, as well as the implementation of the *greedy distributed algorithm* (DGA) as a proof of concept. We present results obtained from the algorithm implementation in the DES-Testbed, a multi-transceiver testbed with more than 100 nodes.

The remainder of this paper is structured as follows. In Section II we present channel assignment algorithms and derive their required services. Section III presents the DES-Chan framework for distributed channel assignment. A reference implementation for DES-Chan is described in Section IV and its evaluation in Section V. The paper concludes with an outlook and future work.

II. RELATED WORK

In this section, we present different distributed algorithms for channel assignment. The list of algorithms is not complete, for a more complete overview we refer to the survey in [15].

In the greedy channel assignment algorithm by Ko et. al [8], one wireless interface of each node is switched to a common channel in order to ensure the network connectivity. For additional interfaces, a greedy algorithm selects the least interfering channel in the interference set using an interference cost function which takes the degree of overlap of two channels into account. The interference set comprises all nodes within the 3-hop neighborhood. As an additional constraint, at least one neighbor must have a radio tuned to the selected channel in order to avoid dead interfaces. Channel changes are communicated with a 3-way handshake.

The *distributed greedy algorithm* (DGA) assigns channels to links instead of interfaces and is therefore topology preserving, meaning that all links are sustained during the channel assignment procedure [17]. A binary interference model, which specifies an interference range of m hops is used. A conflict graph is used to formulate the problem so that the number of edges in the conflict graph shall be minimized. In order to avoid oscillation, each vertex and channel combination can only be changed once.

The link-based channel assignment approach by Sridhar et. al is similar to DGA but additionally takes the expected traffic-load on the nodes into account [16]. The weights for the edges in the conflict graph are specified using a load-matrix that takes the expected traffic for each link into account. The channel assignment problem is then defined as minimizing the sum of the weighted edges of the conflict graph.

The *Skeleton Assisted Partition Free* (SAFE) algorithm uses *minimal spanning trees* (MSTs) to preserve the network connectivity [14]. The 2-hop neighborhood is used as interference model. A conflict graph is used and the goal of the algorithm is to minimize the number of edges in the conflict graph. The channel assignment algorithm consists of two components. A random channel assignment is applied if $C < 2K$, where K is the number of wireless network cards on every node and C the number of non-overlapping channels. Due to the pigeonhole principle, two nodes will share a common channel although they are assigned randomly. The second component of the algorithm introduces the condition that all edges of a MST, the *skeleton*, of the network have to be preserved when $C \geq 2K$. For this, every node randomly chooses a channel set with $K - 1$ channels, leaving one interface unassigned. The node broadcasts its chosen channel set, and if links to all skeleton neighbors are already established it assigns a random channel to the unassigned interface. Otherwise it tries to establish links with the not connected skeleton neighbors by assigning a channel which is in the channel set of all skeleton neighbors. If there is such a channel, it is assigned to the interface, if not, a global common channel is used.

A fast channel switching algorithm is proposed by Kyasanur et. al in [9]. The set of network interfaces on each node are

divided into *fixed* interfaces, which stay on a fixed channel, and *switchable* interfaces. If a node wants to communicate with a neighbor, it tunes one of the switchable interfaces to a channel of a fixed interface of the receiving node. The crucial part of this approach is the way how channels are assigned to the fixed interfaces. The authors present a simple solution, in which a well-known function calculates the channel for fixed interfaces based on the node ID. As an alternative, neighborhood information is considered for the channel selection which requires the periodic exchange of topology information messages. Each node then selects the least used channel for its fixed radio. The NET-X framework was created to implement this fast channel switching algorithm for a wireless testbed environment based on the 2.4 Linux kernel [18]. The implementation of this particular algorithm requires several changes to the Linux network stack and to the driver of the wireless network interfaces which is supported by the framework.

III. DES-CHAN

The DES-Chan framework has been developed for experimentally-driven research on distributed channel assignment in real network environments. It is currently in use at the DES-Testbed at the Freie Universität Berlin. The DES-Testbed comprises a stationary wireless mesh backbone with 100 indoor and outdoor DES-Nodes scattered in three different buildings [4]. Every mesh router runs the Linux operating system and is equipped with three IEEE 802.11a/b/g wireless network interfaces. DES-Chan has been implemented as a Python framework and it is available at the website of the DES-Testbed <http://www.des-testbed.net>.

With DES-Chan a wide range of different algorithms can be implemented, validated, and compared in a real network environment. DES-Chan does not require any changes of the Linux kernel or the wireless network interface drivers. It is therefore easy to integrate into existing wireless mesh network testbeds. However, DES-Chan is not suitable for fast channel switching, since the channel switching time without changes to the WNIC drivers is in the order of milli seconds.

The benefits of DES-Chan are two-fold. Firstly, DES-Chan provides an abstraction layer to the low-level and operating system specific tasks. This abstraction layer enables the researcher to spend most development time on the algorithm logic instead of, for instance, memory management and handling the wireless interfaces. Secondly, DES-Chan provides basic services and data structures often required for typical tasks in channel assignment algorithms. For instance, appropriate data structures have been provided for network graphs, conflict graphs, and interference models. We start the description with the analysis of the common requirements of the presented channel assignment algorithms.

A. Required Services

The presented algorithms require several key services for their implementation and should be provided by a generic

Algorithm	Interface Management	Neighbor Discovery	Topology Monitoring	Message Exchange	Simple Interference Model	Conflict Graph	Queues per Channel
Ko	•	•	–	•	•	–	–
DGA	•	•	–	•	•	•	–
Sridhar	•	•	–	•	–	•	–
Net-X	•	• / –	–	• / –	•	–	•
SAFE	•	•	•	•	•	•	–

Table I: The table shows the key services in relation to the algorithms. The bullet indicates that the algorithm requires the particular service, a dash means it does not. For slow channel switching algorithms, the interface management, neighbor discovery, message exchange, topology monitoring, and interference models are important.

channel assignment framework. The derived services are the following:

- *Interface handling*: This basic service is mandatory for all algorithms in order to change the wireless network interface settings, for example to carry out channel switches.
- *Neighborhood discovery*: Information about the local topology is needed as input for channel assignment decisions in the presented algorithms.
- *Message exchange*: A message exchange service for node to node communication between the instances of an algorithm is needed to exchange topology information messages and to implement handshake mechanisms.
- *Topology monitoring*: A topology monitoring service periodically assesses the local topology and its state. Neighbors, links, and their respective quality can be monitored which enables the algorithms to adapt to topology changes by refining the channel assignment.
- *Interference models*: These models are used to estimate the local interference. They range from simple heuristics such as the m -hop neighborhood, where usually $m \in \{2, 3\}$, to measurement-based approaches for particular network topologies [10], [13]. Even though, the simple heuristics are not very realistic [10], they are easy to implement and therefore widely used. Another common model is the *conflict graph* with algorithms trying to minimize its edges in order to minimize the network-wide interference [7]. Therefore, a framework should provide appropriate data structures and operations for modeling conflict graphs.

In addition to these services, *fast channel switching* algorithms usually need a modified queuing system to avoid channel switching on a per-packet basis. Frames need to be stored which are sent over a channel which is currently not utilized by the wireless network card. In slow channel switching algorithms, different queues are not required since an interface operates on the same channel for minutes or hours.

Table I shows the described services in relation to the presented algorithms. As a conclusion, a generic framework for off-the-shelf IEEE 802.11 hardware should provide all described services but the per-channel-queuing. Due to the relative long channel switching time, slow channel switching are better suitable for these kinds of networks.

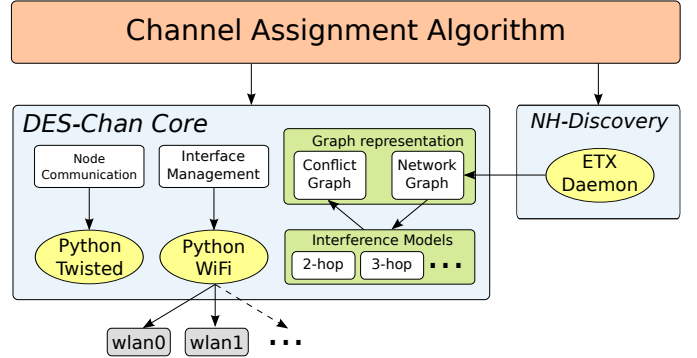


Figure 1: The DES-Chan framework for channel assignment comprises the DES-Chan core and the Neighborhood Discovery service.

B. Architecture and Components

The DES-Chan framework comprises two main components as depicted in Figure 1. *DES-Chan Core* is a Python library that provides common functions and data structures for channel assignment algorithms. It comprises the following services: *interface management*, *node communication*, *graph representation*, and *interference models*. The *Neighborhood-Discovery* module provides a basic service for each node to get information about all neighboring nodes. As future algorithms will require additional services, DES-Chan has been designed to be extensible by additional modules. Therefore, a modular architecture has been chosen. As next, the components are described in detail.

1) *Interface Management*: The interface management module acts as interface to the operating system and hides testbed-specific characteristics from the channel assignment algorithms. It provides various functions for configuring network interfaces and retrieving information about their state. Thus it is possible to set up and shut down interfaces, check whether an interface has been set up, and get information of unused interfaces. The module furthermore allows to tune an interface to a specific channel and thereby implements a crucial requirement for all channel assignment algorithms. The settings of the wireless interfaces are changed with *Python WiFi* [5], a library based on the Linux Wireless Extensions.

2) *Neighborhood-Discovery*: The Neighborhood-Discovery module determines the links and their quality on each network node based on the ETX link metric. The ETX link metric estimates how many transmissions for a packet are required until it is successfully received [2]. ETX values are calculated by each node sending broadcast probes and logging how many probes from their neighbors were successfully received. The forward and reverse delivery ratio are then used for the calculation of ETX, because for unicast communication in IEEE 802.11, an ACK frame has to be successfully received at the sender. The ETX value for a link is then calculated as

$$ETX = \frac{1}{d_f \cdot d_r} \quad (1)$$

where d_f is the forward delivery ratio and d_r the reverse delivery ratio.

The ETX implementation `etxd` for the DES-Chan framework has been realized as a Linux daemon. By default, a probe interval of one second and a window size of 10 seconds is used. The values can be configured via command line arguments. The link quality values are averaged over a moving window, which spans 10 probe intervals. The daemon sends UDP probe packets on the broadcast addresses of the specified network interfaces and listens for incoming probes. In order to always provide up-to-date information, `etxd` dynamically adapts its configuration if network interfaces have been reconfigured, shut down, or brought up.

To offer the neighborhood and link quality information to other programs, `etxd` provides an *inter process communication* (IPC) interface that can be accessed via sockets. A simple, textual protocol allows other applications such as channel assignment algorithms to get the neighbors of a node, as well as the quality and the channel of a certain link. The daemon can be queried to return only those neighbors, which are reached via reliable links, i.e., links whose quality exceeds a certain value. In addition to the link quality, `etxd` also returns the current channel of the WNIC that is used to reach the respective neighbor. As a result, the ETX daemon provides the neighborhood discovery and the topology monitoring service. Via the IPC interface, channel assignment algorithms can query the current state of their links and react adaptively to topology changes.

3) *Node Communication*: Different channel assignment algorithms use different communication protocols and message formats. Thus, a flexible networking library is needed, that allows implementing various protocols with few effort. The Python Twisted[1] library serves as the foundation for the node to node communication in DES-Chan. The library provides an asynchronous networking engine and hides technical details like creating sockets and establishing connections from the developer. The core of Twisted is a global reactor object that can be instructed to monitor sockets and to connect to servers. Based on Twisted, the researcher can quickly develop the required protocol implementation for exchanging messages among the network nodes, for instance in order to propagate changes in channel assignment or to carry out three-way

handshakes prior to the actual channel switch.

4) *Graph Representation*: DES-Chan provides several data structures and functions for graph representation. A data structure for the network graph contains a vertex for every network node and an edge for every link between two network nodes. The edges can be labeled with the number of the channel that is used for the communication. The NetworkGraph class also allows to store a list of channels for each edge, thereby supporting multiple links between node pairs. The NetworkGraph class offers intuitive methods for accessing vertices and edges, and provides several utility functions. It supports printing a human-readable adjacency matrix, which can be used for debugging purposes and for saving the graph in a file. Graph objects can also be stored in the *DOT* format, that can be processed to generate an image of the graph [11]. A function provides the possibility to merge two NetworkGraph objects together by creating the union of both vertex sets and edge sets. This can be used for example to merge network graphs that were obtained from different neighbors.

Additionally, a corresponding ConflictGraph data structure is provided. It allows to apply an interference model to a network graph, in order to calculate the interference between all link pairs in the corresponding network. The class maintains the relation between edges in the network graph and vertices in the conflict graph, and provides corresponding transformation methods. It allows to manipulate the channel of the corresponding edge in the network graph and automatically updates the interference information in the conflict graph. This functionality is needed by algorithms that successively apply several channel assignments to find a configuration that minimizes interference. The ConflictGraph can be easily extended to implement other concepts, such as multi-radio conflict graph [12].

5) *Interference Models*: The DES-Chan framework supports the implementation of various interference models. Currently, only the 2-hop interference model is implemented for reference, but the framework can be easily extended. The current implementation of the 2-hop heuristic supports a binary notion of interference, i.e., two links either interfere or do not interfere, and a more accurate notion of the interference cost taking the spectral distance of two channels into account.

Equation 2 shows the interference cost function I for two center frequencies f_1 and f_2 . The additional parameter α denotes the minimum frequency difference of orthogonal channels.

$$I(f_1, f_2, \alpha) = \begin{cases} 0 & \text{if } |f_1 - f_2| \geq \alpha \\ 1 - \frac{(|f_1 - f_2|)}{\alpha} & \text{otherwise} \end{cases} \quad (2)$$

For IEEE 802.11b/g three channels of the 2.4GHz frequency band are theoretically orthogonal. This can be modeled by setting α to 30MHz, which results in the set of orthogonal channels $\{1, 7, 13\}$. However, experiments showed that this channel distance does not guarantee orthogonality in real network deployments [3], [6].

IV. DISTRIBUTED GREEDY ALGORITHM (DGA)

The *distributed greedy algorithm* (DGA), which has been proposed in [17], has been implemented based on DES-Chan as a proof-of-concept for the capabilities of the framework. DGA assigns channels to links and is therefore topology preserving, meaning that all links are sustained during the channel assignment procedure. A conflict graph is used to formulate the problem so that the number of edges in the conflict graph shall be minimized. Each wireless link between two nodes is owned by the node with the higher node ID and only this node may assign a channel to the link.

At the network initialization, all links are assigned to the same channel. Each node then iterates over all owned links and changes the channel of the link which results in the largest decrease of interference in the local neighborhood. The largest decrease is achieved with the combination of link u and channel k that removes the highest numbers of edges in the local conflict graph. The interface constraint is respected, which means that no more channels can be assigned to a node than it has interfaces. In order to avoid oscillation, each vertex and channel combination can only be changed once.

Channel switches are carried out using a 3-way handshake and update information message for the interference set. This procedure is repeated until the local interference cannot be reduced any further, i.e. all possible (u, k) combinations have been tried. Since each combination is only tried once, the total number of iterations over all instances of the algorithm is $O(|V_c|K)$, where $|V_c|$ is the number of vertices in the network conflict graph and K is the number of available channels.

A. Implementation

Since the approach is a distributed algorithm, it is executed simultaneously on all nodes. The following description refers to the instance running at one specific node. Before the algorithm is started, one of the WNICs is tuned to a common channel and all other WNICs are shut down. Then, the event loop is launched and the program flow is according to the diagram in Figure 2.

The start method uses the *topology monitoring* module from DES-Chan to retrieve the initial network graph. The local conflict graph, which is still empty at that point, is updated with the information of the network graph. Afterwards, the *findMinimum* method is called. It tries to find a vertex-channel combination, that minimizes the interference in the local conflict graph. If the interference can be further reduced, the program tries to apply the combination that provides the largest decrease. Therefore, it sends a `channel request` message to the corresponding neighbor, asking the DGA instance on that node to change the channel accordingly.

After the `channel request` message has been sent, the program flow is interrupted until a reply is received. Thereby, three message types are distinguished: `channel request`, `channel reply`, and `channel reject`. When receiving the `channel request` message, the node checks whether it is able to change the channel without violating the interface constraint. If there is an unused interface available, or no

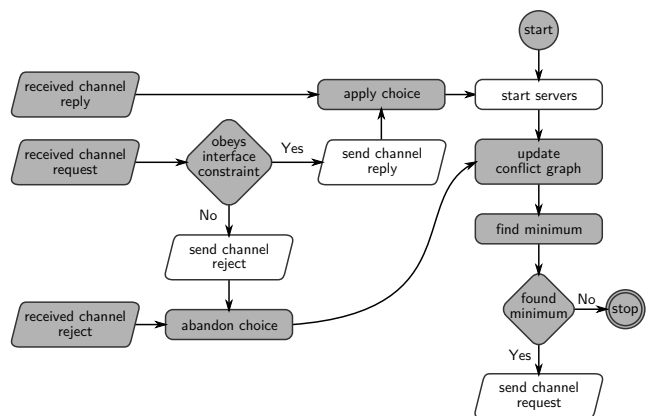


Figure 2: Flowchart of the distributed DGA implementation. Methods of the *DGA* class are colored gray and methods of the *Messaging* class are colored white. Input/Output operations are represented by parallelograms.

other links would be affected by the channel switch, the node answers with a `channel reply` message, indicating that the channel can be changed. After the reply has been sent, the node configures its interfaces according to the new assignment. Otherwise the request is answered with a `channel reject` message and the vertex-channel combination is abandoned. When the originator of the request receives a `channel reply` message, it applies the chosen channel.

If the channel request is answered with a `channel reject` message, the link-channel combination is removed from the set of possible combinations and will not be considered again. Thereby the termination of the algorithm is guaranteed. After the requested channel has been either approved or rejected, the next iteration is started.

V. EVALUATION

The DES-Testbed was used for the evaluation of the DGA implementation based on DES-Chan. The evaluation comprises a graph-theoretic analysis of the remaining interference after channel assignment as proposed by the authors of DGA. The results were compared to the single channel case, where each node uses only one of its interfaces on the same channel. The interference reduction was determined for the already presented DGA algorithm and an additional RAND algorithm which assigns channels to links randomly.

The channels 1 to 13 of the 2.4 GHz spectrum were set as the available channels for the algorithms. For DGA, an interference cost function taking the spectral distance of the channels was used as described in Section III. The experiments were performed on 98 nodes in the DES-Testbed and repeated for 30 times. Each replication comprised three phases. First, all nodes were configured to set up only one interface on channel 1 in order to create the single channel network case. Afterwards the single-hop throughput on 10 non-adjacent links was measured. In the second phase, the DGA algorithm was executed. After the algorithm had finished, the throughput on

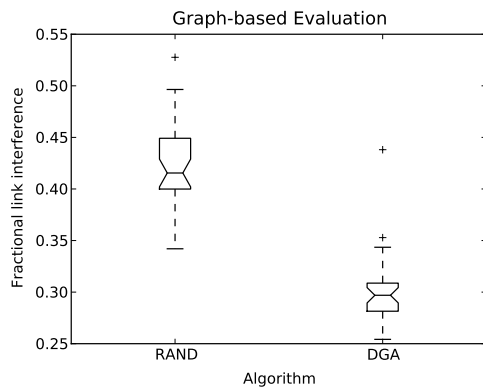


Figure 3: FNI evaluation of DGA and RAND. DGA reduces interference by 72 % compared to a single channel network and by 15 % compared to RAND.

the 10 links was measured again. The same was repeated for the RAND algorithm.

As in the evaluation in [17], the *fractional network interference* (FNI) was used as metric for the graph-theoretic consideration. The FNI is given by:

$$FNI = \frac{\sum_{i \in E_{CA}} weight(i)}{\sum_{i \in E_{SC}} weight(i)} \quad (3)$$

where E_{CA} is the set of edges of the conflict graph after channel assignment and E_{SC} is the set of edges of the conflict graph in the single channel network.

The results of 30 experiment repetitions are shown in Figure 3. After the channel assignment with DGA, the FNI was reduced to 0.28 in the median. Thus, the overall interference in the network was reduced by 72 % compared to the interference in a single-channel network. Furthermore, the low variation in the results shows that the algorithm achieves reproducible results. The results from RAND show a FNI of 0.43 in the median. Thus, even random channel assignment significantly reduces the interference compared to a single-channel network, but the DGA algorithm still achieves a further reduction of 15 %. The presented results correspond to those that Subramanian et al. obtained for their distributed greedy algorithm on random graphs. For a setup with 50 nodes, three orthogonal channels, and three WNICs per node, the authors found that their algorithm achieves an average FNI of 0.3.

VI. SUMMARY AND OUTLOOK

The DES-Chan framework has been developed to support the development process of distributed channel assignment algorithms in wireless mesh networks. We presented the components of DES-Chan and a reference implementation of a well-known algorithm. The evaluation showed that the obtained graph-based results are close to the results published by the authors of the algorithm. However, the results rely on the accuracy of the applied 2-hop interference model. Recent studies have shown, that the simple interference models are

not very accurate [10]. The results of studies on *adjacent channel interference* (ACI) [3], [6] will be incorporated to develop more accurate interference models. Therefore, in a next step we will evaluate the channel assignment algorithms with throughput-based methods by measuring the network saturation in order to validate the results of the FNI analysis.

REFERENCES

- [1] The Python Twisted Documentation (Last visit: 2011). URL <http://twistedmatrix.com/projects/core/documentation/howto/book.pdf>
- [2] De Couto, D.S.J., Aguayo, D., Bicket, J., Morris, R.: A high-throughput path metric for multi-hop wireless routing. In: *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pp. 134–146. ACM, New York, NY, USA (2003). DOI <http://doi.acm.org/10.1145/938985.939000>
- [3] Fuxjager, P., Valerio, D., Ricciato, F.: The myth of non-overlapping channels: interference measurements in ieee 802.11. In: *Proc. Fourth Annual Conference on Wireless on Demand Network Systems and Services WONS '07*, pp. 1–8 (2007). DOI 10.1109/WONS.2007.340486
- [4] Günes, M., Juraschek, F., Blywis, B., Mushtaq, Q., Schiller, J.: A testbed for next generation wireless networks research. *Special Issue PIK on Mobile Ad-hoc Networks IV*, 208–212 (2009). DOI 10.1515/piko.2009.0040. URL <http://www.reference-global.com/doi/abs/10.1515/piko.2009.0040>
- [5] Joost, R., Robinson, S.: *pythonwifi*. <http://pythonwifi.wikispot.org/> (Last visit: 2011)
- [6] Juraschek, F., Günes, M., Philipp, M., Blywis, B.: Insights from experimental research on distributed channel assignment in wireless testbeds. *International Journal of Wireless Networks and Broadband Technologies (IJWNBT)* **1**(1), 32–49 pp. (2011). DOI 10.4018/ijwnbt.2011010103
- [7] Katzela, I., Naghshineh, M.: Channel assignment schemes for cellular mobile telecommunication systems. *IEEE Personal Communications* **3**, 10–31 (1996)
- [8] Ko, B.J., Misra, V., Padhye, J., Rubenstein, D.: Distributed channel assignment in multi-radio 802.11 mesh networks. pp. 3978–3983 (2007). URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=4204903
- [9] Kyasanur, P., So, J., Chereddi, C., Vaidya, N.H.: Multichannel mesh networks: challenges and protocols. *Wireless Communications, IEEE [see also IEEE Personal Communications]* **13**(2), 30–36 (2006). URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1632478
- [10] Padhye, J., Agarwal, S., Padmanabhan, V.N., Qiu, L., Rao, A., Zill, B.: Estimation of link interference in static multi-hop wireless networks. In: *IMC '05: Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pp. 28–28. USENIX Association, Berkeley, CA, USA (2005)
- [11] Project, G.: *Graphviz project* (2010). URL <http://www.graphviz.org/>
- [12] Ramachandran, K.N., Belding, E.M., Almeroth, K.C., Buddhikot, M.M.: Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks. In: *25th IEEE International Conference on Computer Communications (INFOCOM)*, pp. 1–12 (2006). DOI 10.1109/INFOCOM.2006.177. URL <http://dx.doi.org/10.1109/INFOCOM.2006.177>
- [13] Reis, C., Mahajan, R., Rodrig, M., Wetherall, D., Zahorjan, J.: Measurement-based models of delivery and interference in static wireless networks. *SIGCOMM Comput. Commun. Rev.* **36**(4), 51–62 (2006). DOI <http://doi.acm.org/10.1145/1151659.1159921>
- [14] Shin, M., Lee, S., Kim, Y.: Distributed channel assignment for multi-radio wireless networks. *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference* **0**, 417–426 (2006). DOI <http://doi.ieeeecomputersociety.org/10.1109/MOBHOC.2006.278582>
- [15] Si, W., Selvakennedy, S., Zomaya, A.Y.: An overview of channel assignment methods for multi-radio multi-channel wireless mesh networks. *Journal of Parallel and Distributed Computing* pp. – (2009). DOI DOI:10.1016/j.jpdc.2009.09.011
- [16] Sridhar, S., Guo, J., Jha, S.: Channel Assignment in Multi-Radio Wireless Mesh Networks : A Graph-theoretic approach (2009)
- [17] Subramanian, A.P., Gupta, H., Das, S.R., Cao, J.: Minimum interference channel assignment in multiradio wireless mesh networks. *IEEE Transactions on Mobile Computing* **7**(12), 1459–1473 (2008). DOI <http://dx.doi.org/10.1109/TMC.2008.70>
- [18] Wireless Networking Group at UIUC: Net-x channel assignment framework. <http://www.crhc.illinois.edu/wireless/netx.html> (Last visit: 2010)