Balls and Bins Distributed Load Balancing Algorithm for Channel Allocation

Marina Petrova, Natalia Olano and Petri Mähönen
Department of Wireless Networks, RWTH Aachen University
Kackertstrasse 9, D-52072 Aachen, Germany
email: {mpe, nol, pma}@mobnets.rwth-aachen.de

***Abstract*—In this paper we validate through simulations the convergence properties of our earlier proposed *balls and bins* channel allocation algorithm. The algorithm supports dynamic channel allocation based on local link quality information and does not require exchange of network information. Each radio evaluates periodically a cost function related to the desired QoS. Based on the calculated cost the radio may change the channel with probability proportional to the excess of the current cost over a cost threshold set by the user application.**

Earlier theoretical work showed the algorithm has low complexity and excellent convergence properties. In this paper we show that radio network simulations confirm the results. In terms of aggregate network throughput the algorithm offers on average modest gains. The most remarkable improvement that the proposed algorithm offers is in terms of fairness. The results show that the worst link throughput in the network can be improved up to 70%. The algorithm guarantees a certain minimum performance to all the links in the network that is determined by the maximum cost set. If all the transmitters have the same cost threshold and there exists a suitable link allocation for the cost threshold set then fairness in the convergence state is guaranteed.

I. Introduction

Finding efficient channel allocation strategies for WLANs has been an active research topic for some time now. The way wireless channels are (re)used strongly influences the performance and the capacity of the system. This is one of the reasons simple and smart frequency reuse techniques are still attractive to investigate. A handful of wireless technologies operate in the 2.4 and 5 GHz bands where the available spectrum is very limited. Therefore smart techniques are needed to efficiently manage interference and provide better service to clients. The unpredictability of the wireless channel in time and the dynamic behaviour of the wireless networks make the design of a robust and universal channel allocation solution a challenging task. There is plethora of both centralized and distributed solutions for channel assignment in the literature, covering variety of scenarios and methodologies. In [1], [2], [3], authors proposed centralized schemes for automated channel and interference using graph colouring. Other solutions deploy biologically inspired algorithms such as simulated annealing to address the channel allocation problem [4], [5]. For further reading on various channel allocation techniques the reader is referred to [6], [7], [8], [9], [10].

Also game theoretical approaches have been often used to model dynamic channel allocation. Especially in the domain of cognitive radio and dynamic spectrum access, game theory has been extensively applied to study the interaction between the primary and secondary users and identify optimal strategies for channel access. For reference see for example [11], [12], [13], [14].

In [15] and [16] we introduced a distributed load balancing algorithm for dynamic channel allocation. We showed analytically that it is possible to design a simple and yet fast converging algorithm for channel allocation which relies only on local information. Several similar algorithms on load balancing exist in the literature, see for example [17], [18].

In this paper we extend the previous work by comprehensive simulation analysis in order to validate our theoretical findings. We study the convergence properties of the algorithm in a simulated network environment. Furthermore we observe the aggregate throughput and the fairness in the network under our channel allocation algorithm. Our simulation results show moderate gains in throughput and a clear fairness improvement.

The remainder of the paper is structured as follows: In Section II we describe our system model and formulate the proposed protocol for distributed load balancing channel allocation. Section III and Section IV outline the simulation setup and present our simulation results respectively. We conclude the paper with final remarks and discussions in Section V.

II. Dynamic Load Balancing Using The Balls and Bins Approach

A. System Model

We model our distributed channel allocation problem using the *balls and bins* game [19]. The number of balls is n and it is mapped to the number of communicating pairs of agents (radios). The balls are distributed to m bins, which in this case represent the channels to be allocated[1]. The load of each channel $i \in [m]$, n_i corresponds to the number of balls in the bin. For simplicity, we assume that all balls are equal in size i.e. each agent generates the same traffic load in the network. Moreover the agents use the same radio technology to communicate, in our particular case IEEE 802.11. The model will also apply if different agents impose unequal traffic loads. The number of agents is not necessarily fixed in time. It might

[1]Throughout the paper we will interchangeably use the terms balls and agents. The same applies to the terms bins and channels.

happen that new agents come into the network and acquire a channel.

The goal of the balls and bins game is to allocate the available channels to the agents in a *balanced* way such that the overall system achieves the best possible performance. The desired best performance of the system may be, e.g., maximum throughput, minimal interference or minimum delay. Each bin $i \in [m]$ is associated with a *cost or utility function* c_i. We denote the sustained cost for all the balls choosing the bin i as $c_i(n_i)$. We may choose the cost each ball measures to be delay, packet error rate, throughput, etc. or any weighted combination of those. Each of the balls will aim at minimizing the cost which is implicitly related to the number of balls in the bin. One should note that if the measured cost is the achieved throughput, the ball will minimize its reciprocal value.

During the game, the ball has an incentive to change the bin only if it gains more in performance by doing so. The game is played in rounds and all balls are allowed to migrate to another bin concurrently. The agent is "happy" in the current bin as long as its sustained cost is at most C_{max}, where C_{max} denotes a common cost threshold. It is important to mention that the agents are memoryless and they do not take previous states or decisions into account. Furthermore there is no need for information exchange among the agents. The agents use only the locally calculated cost to decide on the next move in the game. The game has reached a stable state if no ball has an incentive to migrate. This implies existence of a Nash equilibrium.

In our earlier work we have shown through thorough analysis that, in the fluid limit, our load balancing algorithm converges rapidly to a feasible solution if such a solution exists [15], [16]. Furthermore the convergence behaviour of our approach is independent of the type of the cost function. We have considered both linear and exponential family of functions in our analyzes. Encouraged by the good convergence properties shown in the previous numerical studies and the simplicity of the algorithm, in this paper we study the performance of the balls and bins channel allocation in a more realistic network environment using Qualnet network simulator [20].

B. Channel Allocation Protocol

Now we will formalize the load balancing channel allocation protocol, which we also refer to as a threshold protocol. Let us consider a number of communication transmitter-receiver pairs (Tx-Rx), which correspond to the numbers of balls n in our model. The cost assigned to each Tx-Rx pair is calculated as an inverse value of the throughput of the link between the communicating agents. The number of channels to be allocated is m. The maximum cost of communication on each channel is set to be C_{max}. The threshold algorithms works as follows: At the beginning all Tx-Rx pairs are assigned a channel randomly. Each transmitter is periodically calculating the current channel cost c_i and if it exceeds the threshold C_{max} the transmitter will decide to switch to another randomly drawn channel j with a probability p defined as follows:

$$p = \frac{c_i - C_{max}}{c_i}.$$

A pseudocode which describes the protocol flow is given below.

Algorithm 1 The Threshold Protocol

```
loop
  for all transmitters in parallel do
    Transmit
    After interval seconds query receiver for feedback
    Calculate cost $c_i$ in current channel $i$
    if $c_i > C_{max}$ then
      With probability $p = \frac{c_i - C_{max}}{c_i}$ do
      1. Draw $j \in [m]\backslash i$ uniformly at random
      2. $i \leftarrow j$
    end if
  end for
end loop
```

The algorithm is very simple and it ensures load balancing among the available channels in a distributed fashion. Because of its stateless nature, it adapts well to fluctuations of number of agents in time, and reaches stable state quickly. Since the algorithm does not require exchange of information among the Tx-Rx pairs, the generated overhead is minimal and limited to the feedback information needed from the receiver, so that the transmitter can calculate the cost c_i. As the algorithm runs in rounds a synchronization between the transmitters is needed in order to provide calculation of the cost almost at the same time. However, very accurate synchronization is not absolutely necessary. The slight synchronization offset can be compensated with a moderately longer convergence time.

III. Simulation setup

The simulations are carried out using the Qualnet Wireless Library for WLAN 802.11. The PHY is set to IEEE 802.11b either at 2 Mbps or 11 Mbps. In the application layer we run a constant bit rate traffic generator for UDP traffic at 1.95 Mbps or 10.9 Mbps respectively. Slight modification has been made to the MAC in order to execute the *threshold algorithm*.

The transmitters and receivers are uniformly distributed in an area of 500x500 m. All nodes transmit with the same output power and they are all in communication range of each other. The simulations are carried out using two-ray propagation model.

We use four non-overlapping frequency channels (2.4 GHz, 2.45GHz, 2.47 GHz and 2.5 GHz) and vary the number of Tx-Rx pairs from 4 to 16. In all the simulations the radios have the same cost threshold C_{max} for a particular simulation setup. The threshold is chosen depending on the maximum number of radios a channel has to support given a certain number of channels and users in order to guarantee a balanced load over all available channels. For example if the number of available channels is 4 and the number of communication pairs 5, it is enough if the the maximum number of radios that the channel has to support is 2. This means that if we want to balance the traffic among all the existing channels, the maximum cost needs to match the performance of the most loaded channel and it should not be higher. The maximum cost is defined as the inverse value of the throughput and it is determined heuristically depending on the maximum realizable throughput for the number of users and channels in a particular simulation setting. For 2 Mbps links the maximum cost is calculated to be 0.008. It corresponds to a case when there are 4 channels and 4 Tx-Rx pairs so that each pair will get a channel. If the number of pairs is between 5-8, there will exist at least one channel which will have to serve 2 Tx-RX pairs, so the maximum cost will be 0.016, etc. For 11 Mbps the logic of calculating the C_{max} is the same. Table I shows the values for different numbers of transmitter-receiver pairs. One should note that in this paper we consider a simplified case where all the channels are of the same "quality". More specifically each channel is using the same channel model, and we consider only interference generated by IEEE 802.11 nodes. Any external interference or co-channel interference is not taken into account in the simulations. Obviously in a real setup some of the channels may suffer from external interference and this should be carefully considered in the modeling.

TABLE I
MAXIMUM COSTS SET DEPENDING ON THE NUMBER OF TX-RX PAIRS IN THE SIMULATION SCENARIO

No. of Tx-Rx	Max. No of Tx-Rx per Ch.	2 Mbps	11 Mbps
4	1	0.008	0.002
5, 6, 7, 8	2	0.016	0.004
9, 10, 11, 12	3	0.024	0.006
13, 14, 15, 16	4	0.035	0.008

The algorithm is iteratively run in rounds, each $10s$ long. This is done to ensure that simulations reach stable conditions, and no initial transient effects influence the results. In principle the interval can be shortened but we wanted to ensure that we get reliable results and avoid possible numerical artifacts. In every round, the successfully transmitted packets are counted. Based on this information each transmitter calculates the current cost in the channel c_i as the ratio between the round time and the number of successfully transmitted packets. If a switch of the channel is to happen the receiver will be informed of the new channel to be used.

IV. RESULTS

A. Convergence Statistics

Figure 1 below shows the average and standard deviation of the convergence time in rounds versus the number of Tx-Rx pairs for ten simulations. Quite naturally the convergence time increases with the number of users. Especially when the number of Tx-Rx pairs is multiple of the number of available channels the algorithm needs more rounds to converge since there are less combinations of valid channel allocations. The standard deviation of the convergence time is also larger for a large number of users. Increasing the number of simulation repetitions can make the error bars smaller. Similar results are observed when the simulations are run at 11 Mbps.

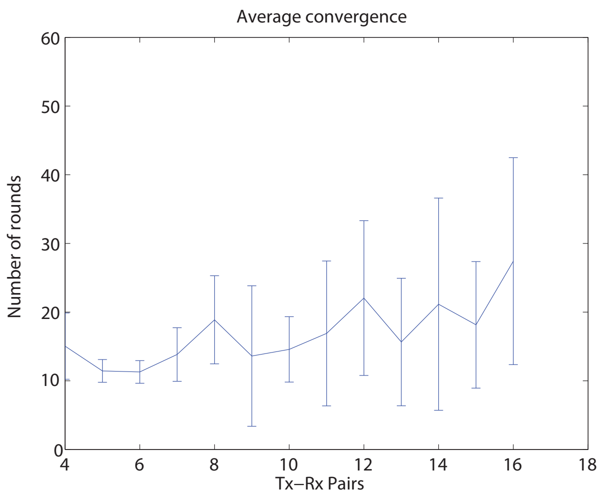

Fig. 1. Convergence time versus number of Tx-Rx pairs with 2 Mbps links.

In Figure 2 we show example of a dynamic channel occupation in time for a simulation run with ten Tx-Rx pairs. The algorithm converges relatively fast to a stable and balanced state. It is easy to spot from the figure that before the algorithm converges, the channel allocation is random and no guarantees of the performance or QoS can be assured.

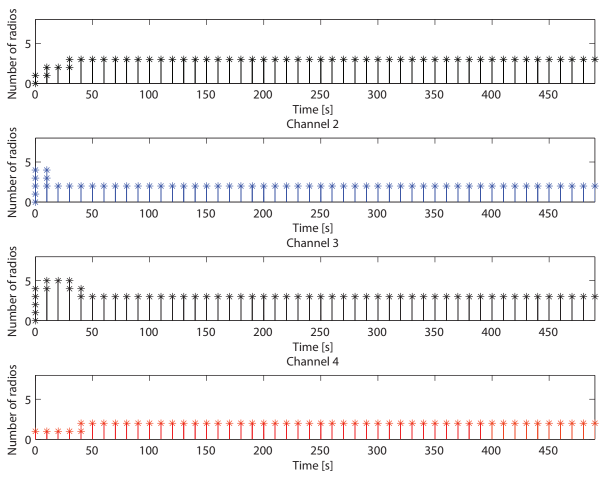

Fig. 2. Channel occupation in time for a run with four channels and ten Tx-Rx pairs.

B. Number of Frequency Changes

The results of the average number of frequency changes per Tx-Rx pair for a $500s$ run are plotted in Figure 3 and Figure 4. As expected, there are more frequency changes for those configurations that take longer to converge. The number of channel changes is relatively small, due to the load balancing algorithm. A high probability to switch the channel occurs only when the maximum cost is exceeded by large margin.

In the worst case (11 Mbps links and eight Tx-Rx pairs) the average number of channel changes for a $500s$ run is 2.5, which is a reasonable overhead for a realistic usage of the algorithm. For several configurations the average number of channel changes per radio is even lower than one per run.

It must also be noted that these are the number of channel changes until convergence of the algorithm. After the algorithm has converged, the event of a channel change should happen only if there are external changes, like the arrival of a new agent or interference from another system.

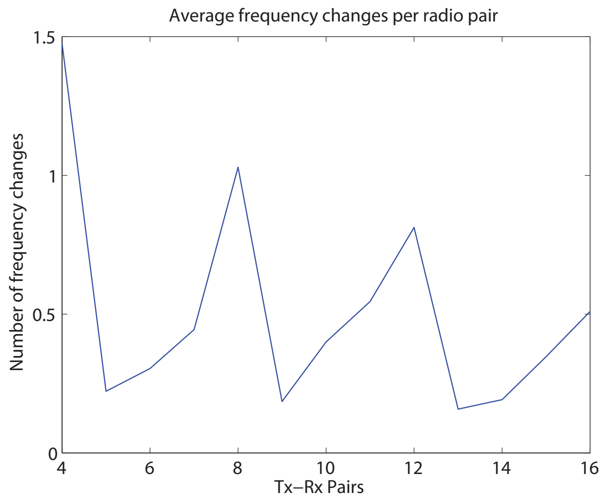

Fig. 3. Number of frequency changes per Tx-Rx pair with 2 Mbps links. The figure shows the number of channel changes till the algorithm converges.

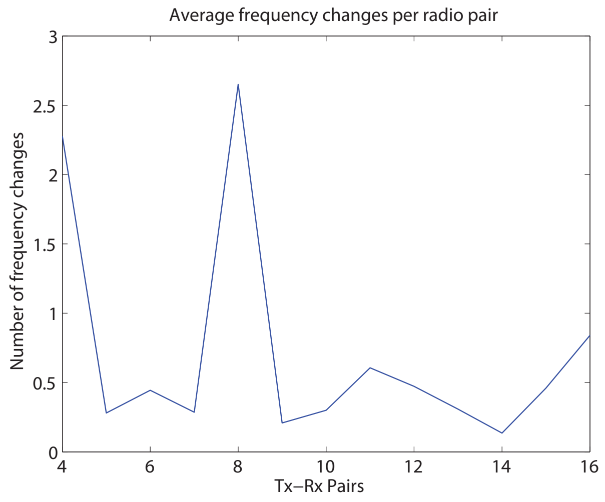

Fig. 4. Average number of channel changes per radio pair for 11 Mbps links. The figure shows the number of channel changes till the algorithm converges.

In this work we have assumed that the cost threshold, C_{max}, is the same for all radios. This can be seen as a universal minimum utility that is required to keep a user happy. One could release this assumption and introduce user specific threshold $C_{i,max}$ in the future. Our expectation is that introducing threshold classes is better choice than allowing total freedom in choosing the threshold value, as it could lead to increased convergence time.

C. Throughput and Fairness

In addition to the convergence properties of the balls and bins algorithm we studied the average throughput over all existing links in the setup after the balls and bins algorithm has run. We compare the throughput results with the those obtained from a WLAN network with the same settings, in the same simulation environment and with the same node distribution in which the Tx-Rx pairs are assigned random frequency and do not execute any further optimization. The results for the minimum and average achieved throughput in both scenarios are shown in Figures 5 and 6.

We observe both in 2 Mbps and 11 Mbps modes that the average throughput is moderately higher if the balls and bins game is played. However, the performance gain in the worst cases simulated with a minimum achieved throughput is significant and leads to up to 60% increase in the throughput for a small number of Tx-Rx pairs in the 11 Mbps mode.

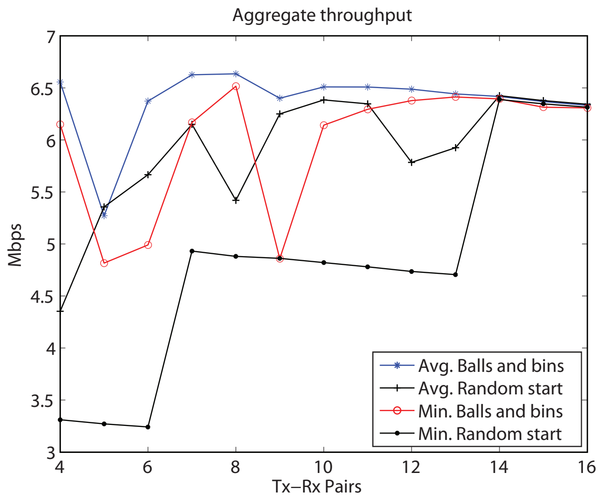

Fig. 5. Average throughput for 2 Mbps links.

Figures 7 and 8 show the throughput in the worst performing links for both channel allocation schemes at 2 Mbps and 11 Mbps. The balls and bins load balancing channel allocation ensures a throughput increase of 71% to 76% for the worst links.

The fact that balls and bins algorithm is also increasing fairness, especially ensuring that there are no big losers in the system, might in the first though look surprising. However, this is one of the strong benefits of the stochastic load balancing algorithm like balls and bins. The protocol guarantees a certain minimum performance to all the links in the network. This is due to the maximum cost associated to each agent. If all the transmitters have the same cost threshold and there exists a suitable link allocation for the cost threshold set then fairness in the convergence state is guaranteed. The algorithm is essentially ensuring that any user experiencing very bad payback from some channel will migrate to the new one.

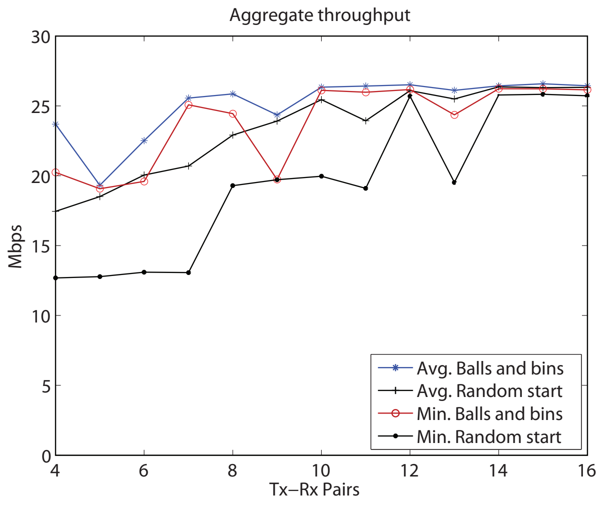

Fig. 6. Average throughput for 11 Mbps links.

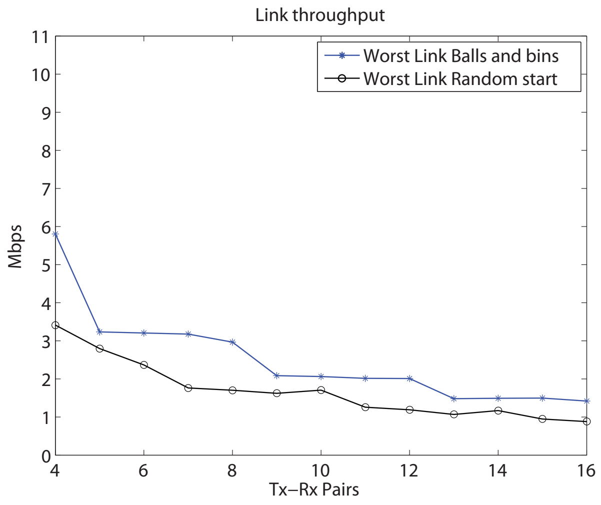

Fig. 8. Average throughput of the worse 11 Mbps link.

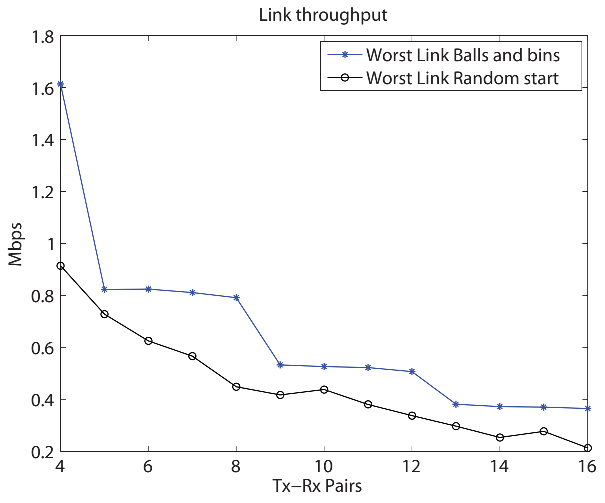

Fig. 7. Average throughput of the worse 2 Mbps link.

Moreover, the distributed nature of the algorithm removes some of the anomalies the centralized protocols generate.

V. Conclusions and Discussion

We have introduced and analyzed through detailed simulations a load balancing algorithm for channel allocation with a low-complexity and minimal information exchange. Our earlier analytical results under quite many simplifications indicated that the balls and bins algorithm is potentially an interesting and efficient distributed resource allocator. In this paper we have extended our earlier work by conducting a number of realistic wireless simulations, and have thus relaxed most of the earlier unrealistic simplifying assumptions. The simulations confirm the qualitative results of earlier analysis, and quantitative results show that the algorithm is able to converge fast to the state, where the system can benefit from the use of the algorithm.

We have shown that the number of iterations required to converge depends mildly on the number of radio equipments. For a realistic number of devices for local area communications, the number of iterations stays a low enough for practical purposes. Although, as expected, more iterations are required in realistic channel conditions than in our earlier analytical model, generally less than 30 iteration rounds is enough to find equilibrium solution. The main benefit of the algorithm is that it finds a load balancing solution by using only local information and it is strictly locally executed algorithm. In the paper we have shown that apart of doing load balancing over the devices, it is also rather surprisingly also providing good fairness conditions. In the highly congested wireless situations, such as WLAN conference hot spots, this feature is a very important.

Because the cost function evaluation is based on the actual measurements, we can guarantee a maximum flexibility for our algorithm. Many other channel allocation algorithms require *a priori* information on wether the channel allocation is done with overlapping or non-overlapping channels. In our case there is no such issue, as the method is not based on explicit interference models. As the number of bins can be flexibly allocated, we can also dynamically select if the system allows the use of overlapping channels.

Our load balancing channel allocation scheme is particularly suitable for many on-demand and mesh network solutions, since it does not require centralized controller to be in the place. The convergence properties in conjunction with a minimal need of signaling make this algorithm a very promising candidate, e.g. when compared against some distributed variations of colouring algorithms. The algorithm is relevant also for self-configuring networks, and it is particularly suitable for cognitive radio networks, where it could be used to balance load between secondary users. As the algorithm can adapt rapidly to changes in the number of radios and allowed channels, it has desirable properties for load balancing system of wireless on-demand networks. The simulations show that our algorithm has good performance, and its algorithmic complexity is extremely low. Its convergence speed is very impressive. The presented approach is thus warranting further studies and experimentation, especially when one takes into account that it is an distributed algorithm with a low signaling requirements (in fact, the load balancing algorithm itself does not necessarily require signaling).

We furthermore think that our paper is of wider interest for on-demand networking community, since balls and bins algorithm itself is a suitable modeling and analysis framework for many load balancing problems. As a simplified protocol and algorithm, it has a dual benefit of being suitable for practical implementations, as well as rigorous analytical analysis.

We are also working on a small scale implementation of our algorithm by using USRP (Universal Software Radio Platform) radio platform with modified gnuRadio software to experiment with the algorithm. Some of the preliminary results are reported in [21]. Although the results from the measurement campaign are still too preliminary, they indicate that the simulation results are accurately predicting the performance over real wireless channels. It is also known that many of balls and bins type of algorithms allow relaxation of the synchronization requirement, i.e. asynchronous and parallel algorithms are possible. We will be considering these extensions in our future work.

Acknowledgment

The authors would like to thank the RWTH Aachen University and the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) for providing financial support through the UMIC research center. We would also like to thank the European Union for providing partial funding of this work through the ARAGORN project. We also thank our collaborators, Simon Fischer and Berthold Vöcking for the intensive and fruitful discussions.

References

[1] J. Riihijärvi, M. Petrova, P. Mähönen, and J. A. Barbosa, "Performance evaluation of automatic channel assignment mechanism for IEEE 802.11 based on graph colouring," in *Proc. of 17th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC'06), Helsinki, Finland*, September 2006.

[2] A. Mishra, V. Brik, S. Banerjee, A. Srinivasan, and W. Arbaugh, "A client-driven approach for channel management in wireless lans," in *Proc. of 25th IEEE International Conference on Computer Communications, (INFOCOM '06), Barcelona, Spain.*, April 2006, pp. 1–12.

[3] J. Riihijärvi, M. Petrova, and P. Mähönen, "Frequency allocation for WLANs using graph colouring techniques," *Journal of Ad Hoc & Sensor Wireless Networks*, vol. 3, no. 2, pp. 121–139, 2007.

[4] B. Kauffmann, F. Baccelli, A. Chaintreau, V. Mhatre, K. Papagiannaki, and C. Diot, "Measurement-based self organization of interfering 802.11 wireless access networks," in *Proc. of 26th IEEE International Conference on Computer Communications, INFOCOM 2007, Anchorage, Alaska, USA.*, vol. 2, May 2007.

[5] J. Chen, S. Olafsson, X. Gu, and Y. Yang, "A fast channel allocation scheme using simulated annealing in scalable wlans," in *Proc. of 5th International Conference onBroadband Communications, Networks and Systems, (BROADNETS'08), London, UK.*, September 2008, pp. 205–211.

[6] S. Chieochan, E. Hossain, and J. Diamond, "Channel assignment schemes for infrastructure-based 802.11 WLANs: A survey," *IEEE Communications Surveys and Tutorials*, to appear.

[7] W. Wang and C. K. Rushforth, "An adaptive local-search algorithm for the channel-assignment problem (CAP)," *IEEE Trans. on Vehicular Technology*, vol. 45, pp. 459–466, August 1996.

[8] D. Leith and P. Clifford, "A self-managed distributed channel selection algorithm for WLANs," in *Proc. of RAWNET, 2006, Boston. MA, USA*, April 2006.

[9] K. Leung and B. Kim, "Frequency assignment for IEEE 802.11 wireless networks," vol. 3, 2003, pp. 1422–1426.

[10] F. Gamba, J.-F. Wagen, and D. Rossier, "A simple agent-based framework for adaptive wlan frequency management," in *Proc. of MOBICOM, San Diego, CA, USA*, September 2003.

[11] J. Suris, L. DaSilva, Z. Han, and A. MacKenzie, "Cooperative game theory for distributed spectrum sharing," in *Proc. of IEEE International Conference on Communications, (ICC '07), Glasgow, Scotland*, June 2007, pp. 5282–5287.

[12] L. Berlemann, S. Mangold, G. Hiertz, and B. Walke, "Spectrum load smoothing for cognitive medium access in open spectrum," in *Proc. of IEEE PIMRC*, vol. 3, 2005, pp. 1951–1956.

[13] R. Etkin, A. Parekh, and D. Tse, "Spectrum sharing for unlicensed bands," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 3, pp. 517–528, April 2007.

[14] S. H. Wong and I. Wassell, "Application of game theory for distributed dynamic channel allocation," in *Proc. of 55th IEEE Vehicular Technology Conference, VTC Spring 2002, Birmingham, Al, USA*, vol. 1, May 2002, pp. 404–408.

[15] S. Fischer, M. Petrova, P. Mähönen, and B. Vöcking, "Distributed load balancing algorithm for adfaptive channel allocation for cognitive radio," in *Proc. of IEEE Conference on Cognitive Radio Oriented Wireless Networks and Communications (CrownCom'07), Orlando, FL, USA*, August 2007, pp. 508–513.

[16] S. Fischer, P. Mähönen, M. Schöngens, and B. Vöcking, "Load balancing for dynamic spectrum assignment with local information for secondary users," in *Proc. of the 3rd IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks (DySPAN'08), Chicago, Illinois, USA*, October 2008.

[17] E. Even-Dar and Y. Mansour, "Fast convergence of selfish rerouting," in *Proc. 16th Ann. ACM–SIAM Symp. on Discrete Algorithms (SODA)*, 2005, pp. 772–781.

[18] P. Berenbrink, T. Friedetzky, L. A. Goldberg, P. Goldberg, Z. Hu, and R. Martin, "Distributed selfish load balancing," in *Proc. 17th Ann. ACM–SIAM Symp. on Discrete Algorithms (SODA)*, 2006.

[19] M. Raab and A. Steger, "Balls into bins– a simple and tight analysis," *Lecture Notes in Computer Science*, vol. 1518, p. 159, 1998.

[20] *The QualNet Simulator*, http://www.scalable-networks.com/.

[21] N. Olano, M. Petrova, and P. Mähönen, "SDR implementation and verification of distributed load balancing spectrum allocation algorithm," in *Proc. of 20th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, (PIMRC'09), Tokyo, Japan*, September 2009.