A Network-Driven Multi-Access-Point Load-Balancing Algorithm for Large-Scale Public Hotspots

Patrick Bosch(✉), Bart Braem, and Steven Latré

University of Antwerp - iMinds, Department of Mathematics and Computer Science
Middelheimlaan 1, B-2020 Antwerp, Belgium
{patrick.bosch,bart.braem,steven.latre}@uantwerpen.be

Abstract. Wireless networks are getting more and more popular and are a basic part of our life with the daily use of smartphones. Users expect high quality connectivity even in public spaces where a high number of clients connect to a limited spectrum on a geographically small area. Therefore, large-scale, high density wireless networks, like they are present at events, are getting more common, but provide a serious resource allocation challenge. Thousands of clients want to connect to a network consisting of multiple APs and a limited spectrum, while all of them should receive a decent connection quality, throughput and delay. Therefore, none of the APs should be overloaded, so that they can provide service for each connected client. The *IEEE 802.11* standard stipulates that the client makes the decision to which AP to connect to. In high-density networks, the individual decision of the client can lead to an AP overload and oscillations in AP association as a client typically has limited information about the network performance and does not collaborate with other clients in taking its decision. This provides unwanted behaviour for load-balancing, as there is no control over the clients. Therefore, we present a method where the APs get control over the client and realise load balancing in such a network. The AP evaluates through a score if the client can connect and, if the client is connected, checks regularly if it is the best option for the client.

1 Introduction

With the rise of smartphones, users expect to be always connected to the Internet. Besides 3G and 4G in outside areas, we have Wi-Fi at home, at work, in public spaces or at events. Because of the increasing demand for broadband connectivity, we can see an increase in deployment of public hotspots. For example, cities offer free Wi-Fi in public areas. The architecture of those hotspots consists of multiple access points (APs). Users expect wireless broadband connectivity with the same quality as at home in these locations. The main challenge for public hotspots is the number of users that use the hotspots and the resulting high density of users. Each AP has only a limited spectrum available, which make proper management techniques necessary to avoid low quality connections with very little bandwidth or even losing the connection.

S. Latré et al. (Eds.): AIMS 2015, LNCS 9122, pp. 30–42, 2015.
DOI: 10.1007/978-3-319-20034-7_3

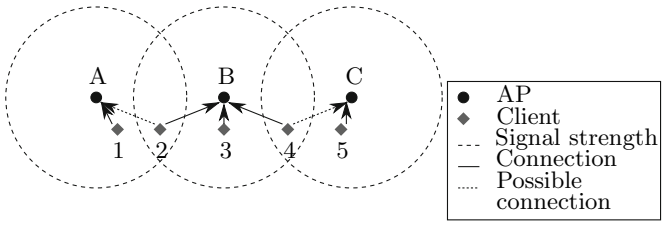

Fig. 1. Possibility of client distribution among APs. Client 2 and 4 have two possible APs each. Client 2 can connect to AP A and B and client 4 to AP B and C.

In *IEEE 802.11*, further mentioned as Wi-Fi, a client is free to choose an AP. The standard selection criterion to choose an AP is the received signal strength indication (RSSI) of said AP. The exact decision is implementation dependent, but usually a client has a list of all available APs and tries to connect to the one with the strongest signal. This can lead to a degradation of the connection quality on an AP due to overload, if most of the clients try to connect to the same AP. Only the AP has knowledge about the number of clients connected to it and how much bandwidth is already consumed. If an AP gets overloaded, the overall throughput and throughput per client drastically decreases, resulting in a bad connection for the client. To avoid this, we need load-balancing to distribute the clients evenly among all available APs.

There are two main approaches to balance the load in such a system. Either a client-driven approach, where the selection criterion of the client is changed, or a network-driven approach, where the AP does the balancing. There are several proposals that are targeting the client and increase throughput through a new selection process [2] [9] [6] [15] [5] [11] [8]. However, this assumes we have control over the client (e.g., by ways of a modified MAC protocol), which is normally not the case. It is more sensible to assume that we cannot change the client and concentrate on the AP to achieve a balanced load.

In this article we present a network-driven load-balancing approach that is able to actively influence clients to connect to the best AP. The contributions of this paper are three-fold. First, we provide a distributed load-balancing algorithm that evaluates the association of the clients via a score computation. Second, we introduce a new way for APs to exchange information, so every AP can compute the score of its neighbours for a client. Third, we provide a way to encourage clients to connect to a more advantageous AP. We use existing standards to control where a client can connect and try to move clients if the score suggests this, making the procedure completely transparent for the mobile device. To demonstrate the possibilities of our algorithm, we set up a simulation in *ns-3* and will provide the results.

The remainder of this article is structured as follows. In Section 2 we discuss previous work. Following in Section 3, we state the problem formally. The algorithm is explained in Section 4 and the performance is evaluated in Section 5. Finally, we conclude in Section 6.

2 Related Work

Load-balancing algorithms for a multi-AP architecture can be organised in two main areas: client-driven and network-driven. As the names suggest, the client-driven driven approach includes changes in the client (e.g., through the MAC protocol) and the network-driven approach requires changes to the network infrastructure (i.e., by changing the protocol implementation of the provided large-scale public hotspots). Most of the client-driven approaches are targeting the selection mechanism of the client and propose a new one that offers load-balancing and increases performance. In practice, a network infrastructure provider typically wants to optimise the performance of the large-scale public hotspot he provides. As he cannot alter clients, an approach that requires the client to change is not possible.

As multi-AP architectures are becoming more common, the *IEEE 802.11* standards committee has also investigated roaming between APs by proposing several extensions to the *IEEE 802.11* standard. *IEEE 802.11-2012* includes the standards previously known as *IEEE 802.11k*, *r* and *v*, which are targeting roaming and information gathering [1] [10]. *IEEE 802.11v* offers the possibility for the AP to suggest a client to move to another AP, but it is still the client, which decides if it will follow the suggestion. These standards may seem very attractive at first, but are not mandatory for a vendor to implement and are not widely used today. Depending on the acceptance of the vendors, these may be a possibility in the future.

Because of the poor acceptance by vendors of roaming-based amendments to *IEEE 802.11*, new load-balancing algorithms are currently an open research challenge in literature. In the default *IEEE 802.11* standard family, the RSSI is the standard selection method for a client. This method is not ideal, because it does not consider the actual load on the AP or how many clients are connected to it. Therefore new selection criteria regarding load balancing were proposed. One possibility is to target the data rate. A new client will evaluate the data rate, while considering the data rate of already connected clients. Additional to modifications on the client, changes to the AP are also necessary to provide more information to the client [2]. Another possibility is to estimate the bandwidth that will be available for the client by sending control frames [9]. Based on that, the client then decides which AP to connect to. Regarding its own throughput and not to reduce the throughput of other clients, a decentralised algorithm is proposed, where each station can decide for itself which AP to connect to [6]. Estimating both, downstream and upstream, a decision is made and the client can connect to the best possible AP [15]. The approach can also be solved by forming zones of devices that are supported by the same set of APs [5]. The client estimates the bit rate of all APs in the zones and selects the one, which can provide enough bandwidth for it. There is also an approach, which implements a whole system that checks out every AP in its vicinity and tests the performance for each network, to find out the best AP to associate with [11]. There are also approaches with less changes to the client [8]. A new field to the beacon message is added, which provides more information for the client. The problem with all

of these approaches, however, is that they completely rely on the support of the client. While this might work for an individual client, in a large-scale public hotspot, it is not realistic to assume a wide adoption of one algorithm. As such, all these algorithms fail in providing a global network optimisation across APs.

The latter calls for more network-driven approaches. One such approach was proposed by Scully et al. [12]. There, a centralised genetic algorithm is presented that can distribute the clients among the APs. However, while the decision is calculated, it does not offer a solution as how to influence the client to connect to the correct AP and the algorithm can not make a decision immediately. In this paper, we provide a way to influence these clients and present our own algorithm for making the decision.

To the best of our knowledge, our algorithm is the first to propose a network-driven solution with direct influence on the clients and without any support of the client itself.

3 Problem Statement

The multi-AP load-balancing algorithm can be formulated as follows. We define a multi-AP architecture with a set of APs $A = \{1, ..., n\}$, which are set up in infrastructure mode and connected via wire. The APs are deployed in such a way that their coverage areas are overlapping (see Figure 1). Further, we have a set of clients $C = \{1, ..., m\}$, with a size of several thousands. The clients can move around freely in a restricted area that is limited in size. C can be divided into two subsets, $MC = \{1, ..., k\}$ and $NMC = \{k + 1, ..., m\}$, according to the behaviour of the clients. MC is the set of *movable clients* whereas NMC is the set of *non-movable clients*. A *movable client* is defined as a client that can be moved to another AP and accordingly a *non-movable client* is defined as a client that cannot be moved to another AP, which means it will stick to the one to which it is connected to and tries to reconnect there. This is an implementation dependent behaviour, as it is not standardised how a client should behave in this situation. Obviously, a client cannot be both, therefore, $MC \cap NMC$. The goal is to have a mapping $C \rightarrow A$, so that all m clients are connected, but every individual client is only connected to one and only one of the APs. This is to be done in such a way that the overall throughput as well as the bandwidth usage and the bandwidth per client is maximised. Additionally to the assignment, it also has to be executed. That is to say, we need a way to encourage clients to connect to the AP that is the best choice for the client and therefore for the network itself. This has to be done without any change to the client.

One possible application would be a public event like a concert. At events, we have a large amount of people, usually several thousands, which are concentrated on a comparatively small area. This results in the aforementioned high density of clients, as well as the mobility of the clients. Visitors of such events tend to move around in the area. The organiser of the event wants to provide additional services through Wi-Fi to enhance the experience of the visitors. Therefore, he places APs throughout the area and the visitors have access to the Wi-Fi network. We cannot assume that visitors will be evenly distributed. It is

more realistic to assume that there will be some areas with higher density and some areas with lower density. APs in higher density areas need to be relieved of too much burden by distributing the clients to other APs. We can also not assume access to client devices, as well as similar behaviour from device to device. We can only assume that clients support the mandatory features of the *IEEE 802.11* standard. Therefore, we can only use standardised messages when we communicate with the client.

4 Network-Driven Multi-AP Load-Balancing

The main idea of the approach is a decentralised score computation algorithm, which has the advantage that we can react faster to the requests of a client, which is necessary if we consider the amount of clients that will try to connect Each AP computes a score for a client when it tries to connect to the AP, as depicted in Figure 2. Beside the computation of its own score, the AP has means to compute the score for its neighbours too. If the AP has the best score, it accepts the client, otherwise it rejects the client and informs the neighbour with the best score. To realise the score computation for neighbouring APs, information needs to be exchanged between the APs. To realise this, we introduce a *neighbour update*. This allows the APs to exchange information about their status. We also consider *NMCs* insofar that we let them connect if they tried more than twice to reconnect. This reduces overall performance, but we need to guarantee connectivity for every client. The score itself is recomputed periodically to check if a client can have a better AP. If this is the case then the client is encouraged to move to the new AP.

4.1 Score Computation

The computation of the score is based on parameters that are broadcasted with the *neighbour update*. We will explain this mechanism later on. The parameters can be arranged in groups according to their interconnectivity. At first we have a group of network related parameters that inform us about the performance of the network, seen from each AP and individual for the AP, which are: {*bandwidth usage (BU), bandwidth per client (BPC), throughput (T)*}. Then we have AP related parameters that give us information about the performance of the AP. These consist of: {*CPU utilisation (CPU), number of clients (NC) (active (NCA), idle (NCI)), supported and actual data rate (SDR, ADR), number of NMC (NON)*}. And at last, we have environment related parameters, which include: {*signal to noise ratio (SNR), location of client and AP* (L_c, L_{ap})}.

All of the information is gathered on each AP. For the purpose of this paper, we will focus on the most important parameters. These encompass the full group of network related parameters {*bandwidth usage, bandwidth per client, throughput*}, part of the AP related parameters {*number of clients (active, idle), number of NMC*} as well as part of the environment parameters {*location of client and AP*} from which we calculate the distance D of the client to the AP.

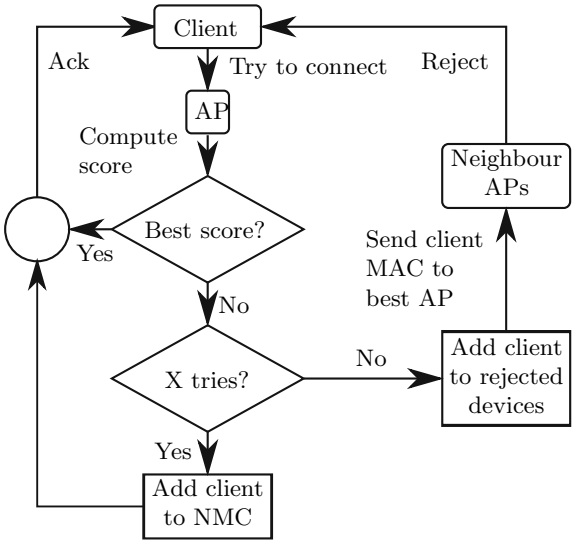

Fig. 2. Client tries to connect to AP. AP computes score to check if the client is allowed to connect and either lets it connect or denies the connection.

We want a fast computation of the score, therefore we prioritise performance above accuracy. To make the comparison of scores easier, we want a single score, but we cannot neglect the dependency of the parameters on to each other. A high number of clients on an AP does not necessarily imply bad performance. If only a very small amount of clients is active and do not need a lot of bandwidth, there is plenty of bandwidth still available. On the other hand, if there are several APs with low bandwidth usage, we still need a way to decide which one is most fitting. In this case the number of connected clients will get important again.

As a general formula for the computation, based on empirical observations, we use

$$\text{AP score} = (NC \times D \times T \times BU \times (1/BPC)) + NC \times D \tag{1}$$

where we choose the AP with the lowest score. In early simulations, we realised that the score can fluctuate very fast and that some clients could not connect due to that. Therefore, we introduced a smoothing interval. If the AP is not the best, but within the interval, then the AP will accept the client nevertheless. Through tests, we determined that 20% is a satisfiable value which prevents fluctuation. Different parameters have a different impact on the score and regarding them exclusively is not sufficient. We weight distance, number of clients and bandwidth per client higher than the others due to their direct influence on the quality of the connection for a client. The bandwidth per client has a high weight because it fulfils mainly two roles. First, it indicates the possible performance for the client on the AP and second it indicates an estimate of the impact on the performance for the client if the throughput is low. Although there is little

to no throughput right now, it could change any moment, because a relatively high amount of clients may be connected to the AP. These connected clients could generate traffic at any time and from this moment on, the AP is a bad choice. To prevent such a sudden impact, the bandwidth per client indicator is of help. The number of clients is another indicator for this scenario, which is the reason for the additional use of it and the distance. One of the parameters, like the throughput, could be equal to zero, for example, because no client is sending anything at that time, but a high amount of clients, compared to the amount of clients serviceable with acceptable quality when they generate traffic, is connected. The clients will then be distributed according to the amount of them on each AP. The distance needs to be considered even when other information is not available, because a longer distance means that it can reduce the bit rate for the AP or that another AP would be a better choice, because it is closer to the client. We call the algorithm with these parameters the *advanced* one.

In the evaluation we compare it with another algorithm, called the *simple* one. In this case we only consider the number of clients on an AP to distribute the clients.

$$\text{AP score} = NC \tag{2}$$

We do not consider any other factor, like throughput or bandwidth per client, in the simple algorithm.

4.2 Client Distribution

The computation is done for each client once when it tries to connect to the AP during the authentication phase. The algorithm itself is illustrated in Figure 2. First, the aforementioned score is computed. Based on this score, the client is then either rejected or accepted. If it is rejected, the best AP will be informed to accept the client immediately, so that the AP does not need to compute the score again. If two APs have the same score, both will be notified and the one where the client tries to connect first accepts the client. We also remember the clients that tried to connect for a time. If the client is trying more than two times to connect to the AP in a short time frame, then it means it is a *NMC*. This client is accepted and marked as such.

As clients move around over time, there might be a better AP to which they should connect. Therefore, a periodic recomputation of the score for each client is done to make sure that the AP is still the best one for the client and that the AP itself does not get cluttered with clients that have a better option. The process is presented in Figure 3. If the client is not active right now, if it does not send traffic, the client will be removed from the AP and prevented to connect to it again. The client will only be prevented to connect again for a specific amount of time, after that it can connect to that AP again. If for some reason the AP becomes the best choice again within that amount of time, the client will not be prevented to connect, because another AP informed that AP that it should let the client connect. If the client is active, it will not be removed, but as soon as it gets inactive, the AP will remove it, so that it can connect to

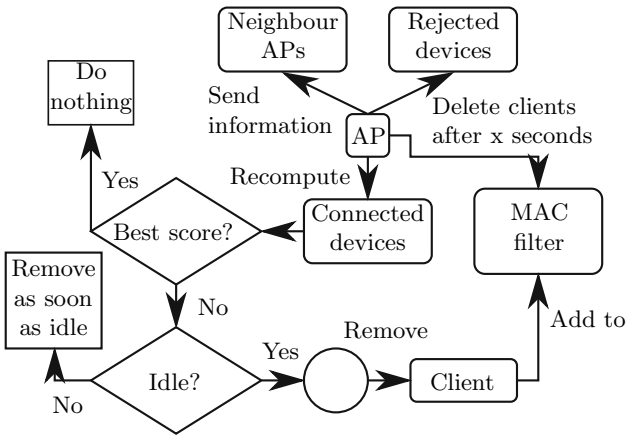

Fig. 3. The AP sends periodic updates to its neighbours and recomputes the score for each connected client

another AP. Through the recomputation, clients are moved around to increase the performance of the AP, but also the performance of the client, because the new AP can offer it a better connection.

Another periodic mechanism is the *neighbour update*. We use it to keep the information on an AP about its neighbours in sync. To be sure to have the same score for each AP-client combination on every AP, we use on each AP only the information that was last received, respectively sent. The update itself is realised with UDP packets. Every AP encodes the information it has to sent into the payload and sends the packet via broadcast in the wired sub-network. The information itself is represented as comma separated values. The clients do not come in contact with it, as it is not broadcasted in the wireless medium. By sending it through the wire the information is available for every AP and they can get the best score available for a client. Additionally to the periodic sending of updates, there is also the possibility to send updates on demand. This is used to inform another AP that it is the best option for a specific client, so that it can immediately accept the client when it tries to connect without computing a score.

4.3 Client Encouragement

To enforce our distribution, we encourage clients to connect to another AP and use only standardised mechanisms for that, so that we do not have to make any changes on the client.

We are using four possibilities that an AP has according to the standard, which are *MAC filter, not answering probe requests, reject at authentication* and *deauthentication*.

The first three are used to prevent a client from connecting to an AP. The rejection at the time of the authentication is used at the time when the client

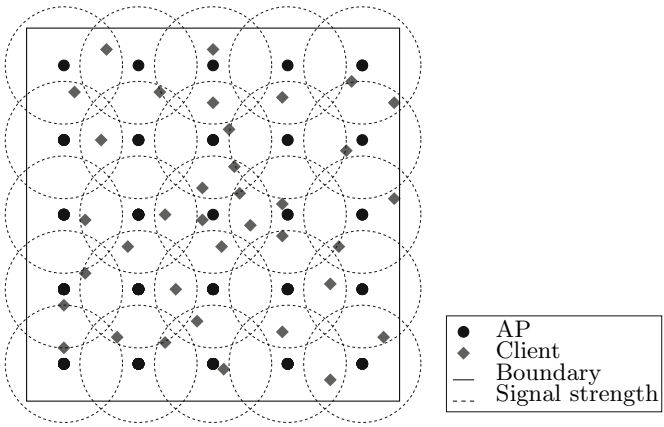

Fig. 4. Topology of simulation. We have 25 distributed APs and randomly distributed and randomly moving clients.

tries to connect for the first time on that AP or enough time has passed that the MAC filter was already removed. The filter is removed after 3 seconds. The MAC filter denies the client to connect to the AP again and the denial of a probe response encourages the client to search for another AP. If the client does not receive a probe response, it will assume that the old AP is not there anymore. Both of those are used when a client was removed from the AP to prevent that the client connects again. The deauthentication is only used to remove a client from the AP and encourage it to choose another AP. After the deauthentication, a MAC filter is used to prevent the client to connect again.

5 Performance Evaluation

5.1 Evaluation Environment

For the evaluation, we set up an *ns-3* simulation. We implemented the algorithm for the APs according to the description above and chose a neighbour update interval of 100 ms. The topology itself is described in Figure 4. 25 APs are evenly distributed in the restricted area of 100 to 100 meters. Their coverage overlaps, so a client can always connect. We used the Wi-Fi standard *IEEE 802.11g* due to the fact that it is the most stable implementation in *ns-3*. The standard propagation model was used with an additional maximum range of 28 meters. We set up the simulation with 500 clients to demonstrate our algorithm. All the clients move around randomly, but they can only move around in the restricted area to achieve a high density of clients at all times. The initial distribution of the clients is according to the random allocation model of the simulator. As a mobility model we chose the random walk model. Traffic is produced by several servers, which send to the clients, which can be seen as a streaming of a video

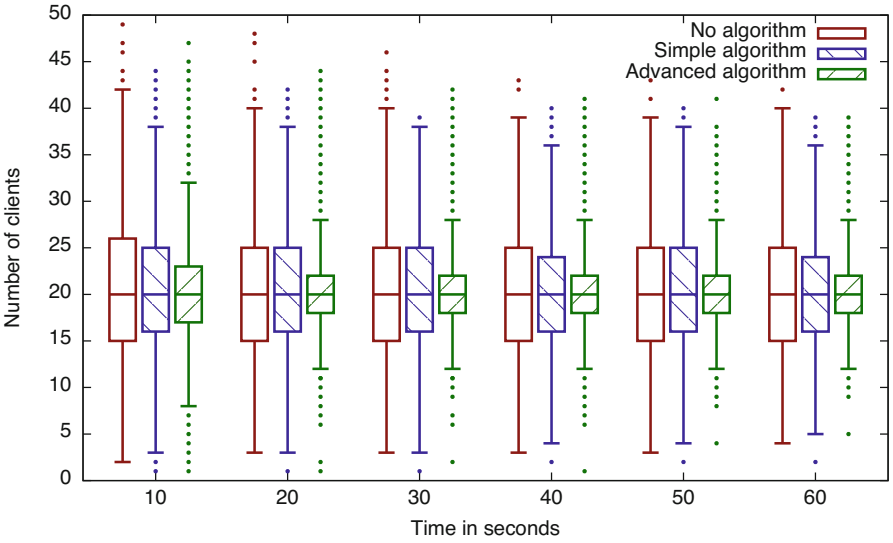

Fig. 5. Distribution of the number of clients over time, captured every 10 seconds. Outliers with the same value are grouped together, their percentage is below 2%.

for example. This is realised through UDP applications with a constant bit rate of 0.25 Mbit/s for each client. The overall simulation time was 70 seconds. The start of the search for an AP for the clients was randomly triggered within the first 5 seconds, to relax the burst a bit and get a more realistic situation. The applications too, were not started immediately, but with 3 ms delay between each other. The APs in this simulation do not have any mechanisms to adapt their settings, like transmission power, transmission range, to avoid collisions and congestion, like some professional APs have. Therefore we chose a setting where we do not have channel saturation, but are close to it. To get meaningful results, we did run the simulation 100 times with different seeds, resulting in 2500 samples for each time stamp. We measured everything directly on the AP.

We will compare our algorithm with the state of the art method, which is basically no control, and the simple algorithm that just considers the number of clients on an AP. We compare the distribution of the clients amongst the APs as well as the throughput distribution amongst the APs.

5.2 Results Description

In all graphs, the whiskers mark the 1.5 interquartile range, while the outliers mark the points outside of that range. Their percentage is very low in each graph and are statistically not significant. Therefore, we will not regard those in our further analysis.

Figure 5 illustrates the evolution of the distribution of clients on all APs for all three scenarios, depicted every 10 seconds. The recomputation is done every 15

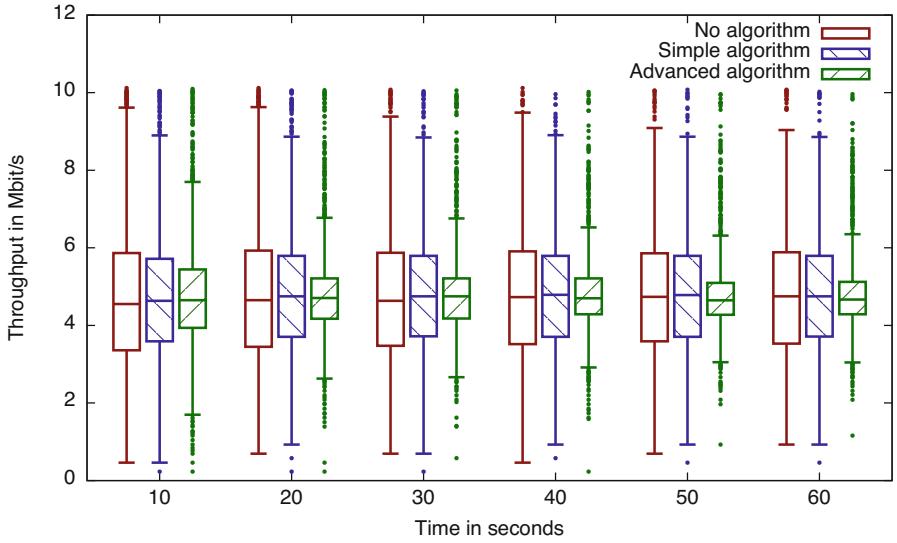

Fig. 6. The distribution of the throughput over time, captured every 10 seconds. Outliers with the same value are grouped together, their percentage is below 1%.

seconds and the clients are moved to another AP and get more evenly distributed with the advanced algorithm than with the other two. The deviation over time is getting lower, until it reaches a stable distribution. Compared to the other two, the advanced algorithm can keep this deviation due to the recomputation and the more complex score computation. The simple algorithm also has an advantage over the state of the art method, as it considers the number of clients as a criterion, but it can not compete with the advanced one. The overall number of clients is stable over all three algorithms, which means all clients are assigned to an AP and can use the connection for services. Overall, the load-balancing algorithms distribute the clients more evenly, where the advanced one is the best amongst them, followed by the simple one and at last the state of the art method.

In Figure 6 we can see the distribution of the throughput of every AP. Similar to the number of clients, the throughput is more evenly distributed with the advanced algorithm than with no algorithm or the simple one. We can see that the minimum value of the throughput is higher for the advanced algorithm due to the better distribution of the clients. The maximum value on the other hand is a bit lower, due to the fact that less clients are connected. Again, the deviation, compared to the other two, is lower and can be kept during the whole time. This means, that there is still throughput available on each AP and the clients experience a better connection with lesser delay and lesser packet loss, by reason of a lesser amount of clients competing for air time.

The better distribution of the number of clients and throughput amongst the APs allows for further capacities on each access point for the advanced algorithm. On the other hand, if we have no algorithm, some APs are already overloaded or

close to it, while others still have capacity. If the number of clients on the APs, which are close to being overloaded, would increase, they can not serve them anymore with an acceptable quality, while the APs with the advanced algorithm still can, because they have spare capacity on each one. In the context of moving clients, the advanced one can handle those more easily as it actively moves them to another AP, while the clients stick to the AP for the state of the art method. This leads to a degeneration of experience for clients over time. The advanced algorithm can avoid this degeneration with the recomputation.

Overall, the simple algorithm does not far as good as the advanced one, because it considers less parameters. But it can still show an improvement over the algorithm with no control. The advanced one considers more parameters and can improve the performance of the network significantly more. The state of the art method can not compete with both of them and shows worse network performance and therefore a worse experience for the client.

6 Conclusion and Future Work

In this paper, we presented a network-driven approach that can improve the performance of the network, regarding throughput distribution and distribution of clients, without any change on the clients. The control over the decisions where a client is connected, is transferred to the AP. The information exchange between the APs allows for a better view on the network and can be used on each AP to independently decide which is the best AP for a client. The decision is enforced through the use of standardised methods.

As future work, our approach can be combined with other methods, such as cell breathing to allow for further improvement [3] [7]. If the standards *IEEE 802.11k/r/v* become more commonly implemented, these can also be used [1] [10]. A crowd movement model, as well as more realistic traffic generation can be added. The placement of the APs can also be researched, as realistic settings do not always allow for evenly distributed APs. As the proposed algorithm might need more messages from clients, the impact on the battery level can be researched.

Acknowledgements. Part of this work has been funded by the iFEST project, cofunded by iMinds, a digital research institute founded by the Flemish Government. Project partners are PlayPass, ID&T, Telenet, Argus Labs, Cozmos and Sendrato, with project support from IWT.

References

1. IEEE Standard for Information technology–Telecommunications and information exchange between systems Local and metropolitan area networks–Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007), pp. 1–2793 (2012)

2. Abusubaih, M., Wiethoelter, S., Gross, J., Wolisz, A.: A New Access Point Selection Policy for Multi-rate IEEE 802.11 WLANs. Int. J. Parallel Emerg. Distrib. Syst. 23, 291–307 (2008)
3. Bahl, P.V., Hajiaghayi, M.T., Jain, K., Mirrokni, S.V., Qiu, L., Saberi, A.: Cell Breathing in Wireless LANs: Algorithms and Evaluation. IEEE Transactions on Mobile Computing 6, 164–178 (2007)
4. Bejerano, Y., Han, S.J., Li, L.: Fairness and load balancing in wireless lans using association control. IEEE/ACM Transactions on Networking 15, 560–573 (2007)
5. Dandapat, S., Mitra, B., Choudhury, R., Ganguly, N.: Smart Association Control in Wireless Mobile Environment Using Max-Flow. IEEE Transactions on Network and Service Management 9, 73–86 (2012)
6. Fukuda, Y., Abe, T., Oie, Y.: Decentralized access point selection architecture for wireless LANs. In: Wireless Telecommunications Symposium, pp. 137–145 (2004)
7. Garcia, E., Vidal, R., Paradells, J.: Cooperative load balancing in IEEE 802.11 networks with cell breathing. In: IEEE Symposium on Computers and Communications, ISCC 2008, pp. 1133–1140 (2008)
8. Gong, H., Kim, J.: Dynamic load balancing through association control of mobile users in WiFi networks. IEEE Transactions on Consumer Electronics 54, 342–348 (2008)
9. Lee, H., Kim, S., Lee, O., Choi, S., Lee, S.J.: Available Bandwidth-based Association in IEEE 802.11 Wireless LANs. In: Proceedings of the 11th International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, MSWiM 2008, pp. 132–139. ACM, New York (2008)
10. Machań, P., Wozniak, J.: On the fast BSS transition algorithms in the IEEE 802.11r local area wireless networks. Telecommunication Systems 52, 2713–2720 (2013)
11. Nicholson, A.J., Chawathe, Y., Chen, M.Y., Noble, B.D., Wetherall, D.: Improved Access Point Selection. In: Proceedings of the 4th International Conference on Mobile Systems, Applications and Services, MobiSys 2006, pp. 233–245. ACM, New York (2006)
12. Scully, T., Brown, K.: Wireless LAN Load-Balancing with Genetic Algorithms. In: Allen, T., Ellis, R., Petridis, M. (eds.) Applications and Innovations in Intelligent Systems XVI, pp. 3–16. Springer, London (2009)
13. Siris, V., Evaggelatou, D.: Access point selection for improving throughput fairness in wireless lans. In: 10th IFIP/IEEE International Symposium on Integrated Network Management, IM 2007, pp. 469–477 (2007)
14. Takeuchi, S., Sezaki, K., Yasuda, Y.: Access point selection strategy in ieee802.11e wlan networks toward load balancing. Electronics and Communications in Japan (Part I: Communications) 90, 35–45 (2007)
15. Vasudevan, S., Papagiannaki, K., Diot, C., Kurose, J., Towsley, D.: Facilitating Access Point Selection in IEEE 802.11 Wireless Networks. In: Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement, IMC 2005, pp. 26–26. USENIX Association, Berkeley (2005)