Knowledge-Based Systems 22 (2009) 529–534

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Wireless LAN load balancing with genetic algorithms

Ted Scully *, Kenneth N. Brown

Centre for Telecommunications Value-chain Research, Cork Constraint Computation Centre, Department of Computer Science, University College Cork, Ireland

ARTICLE INFO

Article history:
Available online 9 January 2009

Keywords:
Optimization
Genetic algorithms
Micro-genetic algorithms
WLANs

ABSTRACT

In recent years IEEE 802.11 wireless local area networks (WLANs) have become increasingly popular. Consequently, there has also been a surge in the number of end-users. The IEEE 802.11 standards do not provide any mechanism for load distribution and as a result user quality of service (QoS) degrades significantly in congested networks where large numbers of users tend to congregate in the same area. The objective of this paper is to provide load balancing techniques that optimise network throughput in areas of user congestion, thereby improving user QoS. Specifically, we develop micro-genetic and standard genetic algorithm approaches for the WLAN load balancing problem, and we analyse their strengths and weaknesses. We also compare the performance of these algorithms with schemes currently in use in IEEE 802.11 WLANs. The results demonstrate that the proposed genetic algorithms give a significant improvement in performance over current techniques. We also show that this improvement is achieved without penalising any class of user.

1. Introduction

The uptake in popularity of IEEE 802.11 wireless local area networks (WLANs) in recent years has been remarkable. WLANs are now the most popular technology used to provide broadband access to IP networks such as extended home networks and internet access in public locations [17]. The proliferation of WLANs has resulted in an ever-increasing number of end-users with heterogeneous quality of service (QoS) requirements. In addition these users tend to congregate in certain areas of the network for various reasons such as availability of favourable network connectivity, proximity to power outlets and coffee shops [1]. Such behaviour leads to congestion at particular areas within the network. Such congestion creates an unbalanced load in the network and reduces overall network throughput.

A WLAN typically provides a number of Access Points (APs) that provide service to users in a particular geographical area. Users select access points based on the strongest received signal strength indicator (RSSI) [17]. Thus although a congested area may be offered service by several APs, if the users are clustered together, they will tend to be connected to the same AP. The more users that are connected to a single AP, the less bandwidth they will receive. For example, in the simple scenario depicted in Fig. 1, all users are connected to AP *B* because it has the strongest signal strength for each user. The resulting system imbalance can be easily rectified if users 1 and 3 migrate to AP *A* and users 5 and 6 migrate to AP *C*. For the sake of illustration, we assume that the users depicted in Fig. 1 have homogeneous demands.

The objective of this paper is to provide efficient algorithms for solving the WLAN load balancing problem: distribute users amongst a set of APs to maximise the average bandwidth per user. Therefore, the algorithms will assign each user to an AP as opposed to each user making that choice independently. Since users connect to and disconnect from the network in real time, we are also interested in efficiency with which the algorithms deliver effective solutions.

We propose two genetic-based load balancing algorithms. The first is a standard genetic algorithm (GA), which we refer to as *MacroGA*, while the second is a micro-genetic algorithm, which we refer to as *MicroGA*. In the context of the WLAN load balancing problem, GAs are attractive as candidate solutions because of their ability to discover good solutions rapidly in difficult high dimensional problems. We evaluate, via simulations, the performance of the GAs and demonstrate that they provide significant enhancements over the standard RSSI approach and other popular load balancing mechanisms. Further, we demonstrate that they do not achieve this by penalising any obvious class of user. The rest of this paper is structured as follows. Section 2 discusses background knowledge and motivates the use of genetic algorithms as potential solutions. Section 3 provides a problem description. Section 4 presents the implementation details of *MicroGA* and *MacroGA*. Section 5 empirically analyses the performance of the proposed solutions. Finally, conclusions are drawn and future areas of research are identified in Section 6.

* Corresponding author. Tel.: +353 87 6391184.
E-mail addresses: Ted.Scully@lit.ie (T. Scully), K.Brown@cs.ucc.ie (K.N. Brown).

0950-7051/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2008.10.008

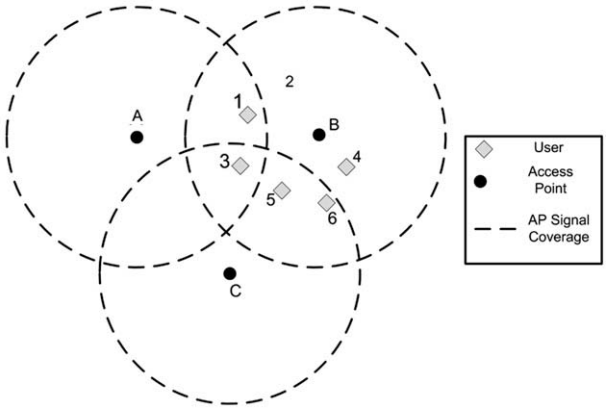

Fig. 1. A basic IEEE WLAN configuration.

2. Motivation and background knowledge

GAs are population-based meta-heuristic optimization algorithms based on an analogy to biological evolution and have been successfully applied to a broad range of real-world NP-Hard problems such as scheduling [13] and data-mining [11]. Standard GAs generate a relatively large population of candidate solutions (there may be several hundred) and iteratively evolve these solutions over time. In contrast, a micro-GA algorithm has a small population size that is periodically reinitialized. The idea of utilising a small population GA was first proposed by Goldberg [9]. He evolved the population using normal genetic operators until it reached a nominal convergence, that is until each individual in the population had the same or similar genotype. When convergence occurred, the fittest individual from the population was copied into a new empty population; the remaining places in the population were filled by randomly generated individuals. The first comparison between standard GAs and micro GAs was performed by Krishnakumar [12]. A micro GA similar to that proposed by Goldberg [9] was proposed and compared with a standard GA. The result demonstrated that the micro GA actually outperformed the standard GA on a number of problem sets. Subsequently, many other researchers have developed applications of micro-GAs ranging from multi-objective optimization [6] to constraint satisfaction problems [7]. However, to the best of our knowledge, the current paper is the first attempt to apply a micro-GA or even a standard GA to the WLAN load balancing problem.

As previously mentioned the ability of GAs to rapidly discover good solutions in difficult high dimensional problems make them attractive as potential solutions to the load balancing problem, which is an *NP*-hard problem [3]. Unlike many other local search algorithms, GAs are intrinsically parallel, which allows them to simultaneously explore different areas of the solution space. This enables GAs to quickly identify good solutions and exploit synergies between solutions. It is this ability to quickly produce good results that makes GAs an attractive prospect from a network operators perspective, where calculating the optimal user/AP configuration is often a time critical operation. This is particularly evident in dynamic networks that exhibit a high degree of user mobility, which causes the optimal user/AP configuration to rapidly change over time. In an effort to satisfy end-user QoS requirements in such an environment, operators sacrifice solution optimality in favour of the more practical option of obtaining good solutions quickly.

Previous work on the WLAN load balancing problem can be subdivided into three categories: (i) user-controlled, (ii) network-centric and (iii) cell breathing. The user-controlled approach to load balancing allows the end-user the autonomy to choose the AP to which it wishes to connect. As mentioned in Section 1 the current default method of association is RSSI. Some vendors have addressed the limitations of RSSI by incorporating load balancing features into network drivers and firmware for APs and wireless cards [5]. Each AP broadcasts the number of users which it currently serves to all users within its signal range. A user within range of several APs will subsequently connect to the AP with the least number of users. This technique is commonly referred to as least loaded first (LLF). We use LLF as one of the benchmark algorithms against which we evaluate the proposed algorithms.

A number of other user-controlled techniques have been proposed in literature that aim to improve load balancing in WLANs. Typically, these techniques consist of a (weighted) function that incorporates multiple metrics such as the RSSI, LLF and other measures of link quality [8,15]. The distributed nature of user-controlled techniques make them attractive from a load balancing perspective. There is no single point of failure. However, user-controlled techniques are limited in that each user views the connection problem from its own perspective. Consequently, an optimal network configuration cannot be guaranteed. This assertion is supported by the empirical results in Section 5.

Network-centric load balancing searches for the set of user/AP connections that optimises some measure of network performance (typically network throughput). This category of load balancing allows for complete control of user assignment and so it can realistically pursue an optimal solution. A number of network-centric algorithms have been proposed in the literature [3,4,16]. Each of these techniques demonstrated significant improvements over the standard RSSI approach. However, they are computationally heavy and unlike the anytime genetic algorithms these approaches are not applicable to the time critical network environments addressed in this paper. The algorithms proposed in this paper fall into the network-centric load balancing category. The application of evolutionary techniques to the WLAN load balancing problem has not yet been investigated.

Finally, cell breathing techniques modify the dimensions of an AP cell to control user/AP association. An AP can change the power at which it broadcasts its signal, thus increasing or decreasing the number of users that may select it. It is an interesting and worthwhile approach to load balancing that has received significant research attention over the last few years [2,10]. Unlike many user-controlled and network-centric load balancing techniques, cell breathing does not require clients to possess appropriate wireless cards. Client side software does not need to be modified and users can continue to utilise the standard RSSI approach. The drawback of cell breathing is that it does not provide the level of fine-grained control that is offered by network-centric algorithms. For example, an AP cannot exclude a user at distance d from the AP without also excluding all users that are at a distance of d or greater from the AP.

3. Problem description

We consider a problem environment consisting of an ordered sequence of m access points $A = \langle a_1, a_2, \ldots, a_m \rangle$ and n users $U = \langle u_1, u_2, \ldots, u_n \rangle$. Each AP transmits an omni-directional signal with a maximum transmission range of 150 m. The APs are arranged in a grid and are separated by a uniform distance of 100 m. A number of factors affect the bit rate that a user u_i receives from an AP a_t. One of the primary factors is the distance between u_i and a_t. The further u_i's distance from a_i, the smaller its achievable bit rate. We assume that the maximum bit rate of a user within 50 m of the AP is 11 Mbps and we refer to this area as *zone1*. A user between 50 and 80 m is in *zone2* and can obtain a maximum of 5.5 Mbps. Users located between 80 and 120 m of the AP are in

zone3 and can offer a maximum bit rate of 2 Mbps. Finally, *zone4* is between 120 and 150 m and a user located in this zone can obtain a maximum bit rate of 1 Mbps from the AP. These figures are commonly adopted in literature [3].

The final bit rate received by u_i is also influenced by the type of service it requires. For example, end-users running streaming media, voice or data applications all require different levels of bandwidth. Therefore, we incorporate heterogeneous QoS requirements into our problem description by defining different categories of users. We describe the category of a user $u_i \in U$ by a weight w_{u_i}, that specifies its service requirement. The weight is used to determine the bit rate allocation b_{u_i} that user u_i is allowed to receive compared to the other users within the same zone. For example, a user $u_i \in U$ is entitled to have a bandwidth $b_{u_i} = w_{u_i}/w_{u_k}$ of any other user $u_k \in U$ that occupies the same zone and is connected to the same AP. This technique of categorising users was also used in Bejerano et al. [3].

The following describes the mechanism used to distribute bit rates amongst users connected to an AP. The initial step is to assign an available bit rate to each of the zones. Initially, we identify the number of connected users in each zone. We refer to a zone with one or more connected users as an active zone. The maximum bit rate of the AP, which was set to 10 Mbps, is divided amongst the active zones so that each zone receives an available bit rate that is proportional to the zones maximum bit rate. For example, consider a problem with two active zones: *zone1* and *zone2*, which have a maximum bit rate of 11 and 5.5 Mbps, respectively. Given that the bit rate of the AP is 10 Mbps, *zone1* and *zone2* would be allotted 6.6 and 3.3 Mbps, respectively.

The available bit rate that is assigned to each zone must now be divided amongst the users that populate the zone. The bit rate received by a user depends on the user's weight and the weight and number of other users within the zone. Consider the scenario where 3 Mbps is the available bit rate in a zone populated by the users, u_i, with $w_{u_i} = 2$ and u_j, with $w_{u_j} = 4$. The user u_i will receive a bit rate of 1 Mbps and u_j will receive a bit rate of 2 Mbps.

To incorporate congestion into our problem model we randomly locate users based on their polar coordinates generated uniformly at random (a user's distance from the centre of the AP grid and the polar angle are uniformly distributed between $(0, 150)$ and $(0, 2\pi)$, respectively). Hence, user density at the centre of the AP grid is higher than it is near the periphery. Again this is in keeping with previous work [3,4].

The objective of the GAs proposed in this paper is to assign each user u_i to an AP a_t so that the sum of all user bit rates is maximised.

4. Genetic load balancing algorithms

Section 4.1 presents the implementation details of *MicroGA*, while Section 4.2 describes *MacroGA*.

4.1. The MicroGA algorithm

The problem of representation in GAs can be described as determining a mapping from the phenotypes to the corresponding genotypes. In *MicroGA* and *MacroGA* a phenotype is a set of user/AP connections, such that each user is connected to a single AP. We map a given phenotype to an integer based genotype. We represent a genotype as an ordered sequence of n integers $\langle a_{g1}, a_{g2}, \ldots, a_{gn} \rangle$ such that $\forall i \in 1, \ldots, n$, $a_{gi} \in A$ and represents the AP to which user u_i is assigned. An example of the representation used is depicted in Fig. 2, which shows a phenotype and its corresponding genotype.

The *MicroGA* algorithm presented in this paper is based on the micro-GA described by Krishnakumar [12]. The initial stage of the algorithm involves the random generation of a micro-population of 5 individuals. We denote the population for generation number x as p_x. An individual is constructed by randomly assigning each user to an AP within the user's range. Given the user/AP connection defined by an individual, fitness assignment involves calculating the sum of all individual user bit rates. The process of determining the bit rate for each user, based on their AP connection, is described in Section 3. Other fair scheduling algorithms have been proposed in Ni et al. and Buddhikot et al. [3,14]. *MicroGA* employs a form of elitist strategy; upon completion of the fitness assignment phase the fittest individual is copied to p_{x+1}, the population for the next generation.

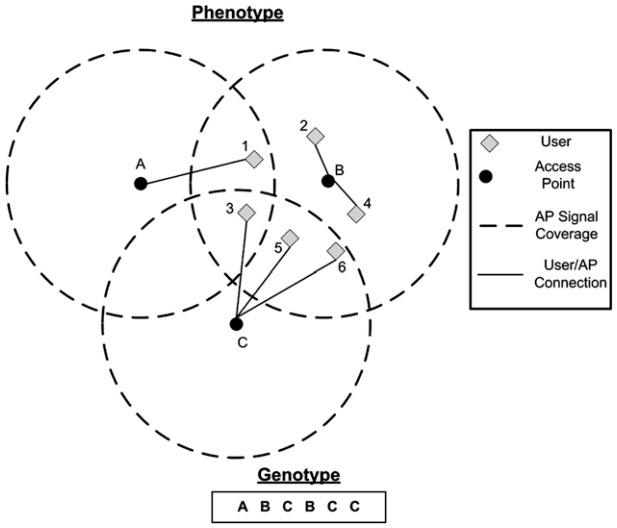

Fig. 2. A example phenotype and its corresponding genotype for a WLAN user/AP configuration.

GAs have a tendency to converge toward a single solution. The smaller the population size the quicker the algorithm will converge. Therefore, ensuring population diversity in a micro-genetic algorithm is crucially important. Micro-genetic algorithms achieve diversity by reinitializing the population if it is deemed to have converged. While there are a number of possible methods of checking for convergence, *MicroGA* monitors the gradual improvement in overall fitness over multiple generations. Convergence occurs if α number of generations expire without any improvement in fitness value. That is, if the fittest individual in generation x has the same fitness level as the fitness individual in generation $x + \alpha$. If the population does converge *MicroGA* randomly generates four individuals and inserts them into the population p_{x+1}. The algorithm subsequently commences the next generational loop. However, if the algorithm has not converged, normal execution continues.

The next stage of the *MicroGA* algorithm involves the selection of four parent individuals for the mating pool. This is achieved through binary tournament selection of the population p_x. That is, two parents are picked randomly from p_x and the fitter of the two is copied into the mating pool. The subsequent phase of the algorithm is crossover. Two parent individuals are randomly selected from the mating pool. The crossover operator iterates through each gene in the parents' genotypes. If the probability of crossover is true, the gene at index i of one parent is copied to index i of the second parent and vice versa. This method of crossover was chosen because experimental evaluation showed that it consistently outperformed the standard n-point crossover for various values of n. The resulting individuals are copied into the population p_{x+1}. Mutation was not performed on the population because reinitialization adds enough diversity to the population. Also the inclusion of a mutation operator did not demonstrate any improvement in performance in the experimental evaluation.

4.2. The MacroGA algorithm

Much of the functionality included in *MicroGA* is replicated in the *MacroGA* algorithm. Therefore, to avoid repetition the following description provides a brief overview of the *MacroGA* genetic algorithm. Through each generational loop the population will first undergo fitness assignment. The *MacroGA* utilises the same mechanism of fitness assignment as the *MicroGA* algorithm. The fittest individual in the current population is copied to the population for the next generational loop. Binary tournament selection is used to populate the mating pool with parents. The recombination phase consists of the two stages. The first is crossover as described for *MicroGA*; the second is mutation. While convergence in the *MacroGA* algorithm occurs at a slower rate than in the *MicroGA* it is still necessary to ensure some sort of diversity. The mutation operator in *MacroGA* performs this function. It picks each individual in turn and iterates through their constituent genes. If the probability of mutation is true then the gene at index i is replaced by a random gene. The gene at index i represents the AP to which user i is attached. Mutation picks another random AP within signal range of the user and assigns it to index i.

5. Empirical evaluation

The objective of the empirical evaluation is to investigate the performance of *MicroGA* and *MacroGA* compared to other common load balancing techniques and also to perform a comparative analysis of the proposed algorithms. Section 5.1 presents the experimental methodology. Section 5.2 presents the network throughput achieved by each algorithm over time while Section 5.3 analyses the distribution of bandwidth amongst users by the proposed algorithms.

5.1. Experimental methodology

To evaluate the *MicroGA* and *MacroGA* algorithms, we compare their performance with that of three other popular mechanisms: RSSI, LLF and HLB (hybrid load balancing). HLB combines the RSSI and LLF techniques. It connects each user to the least-loaded AP within the user's signal range, and in the event of a tie, it connects the user to the AP with the highest signal strength.

The experimental evaluation considers a network populated by 20 APs, which are arranged in a 5 by 4 grid where the distance between adjacent APs is set to 100 m. The parameters for the two GAs were selected during an initial evaluation period. The *MicroGA* convergence parameter α has a value of 5 and its crossover probability is 50%. The population size of *MacroGA* is 200. Its crossover and mutation probability are 50% and 0.5%, respectively.

The three results presented in Section 5.2 analyse the load balancing capability of all algorithms for the following scenarios: (i) a relatively lightly loaded network of 50 users, (ii) a moderately loaded network of 100 users and (iii) a heavily loaded network of 250 users. Each experimental result presented in this section is derived from the average of 30 independent experimental runs, where a single run executes each algorithm for a fixed period of time on the same network/customer configuration.

5.2. Network throughput results

The initial result obtained from running the algorithms in a network populated by 50 users is depicted in Fig. 3. Each algorithm was allowed to run for a total of 10 s and every 0.1 s it reported its best result. LLF exhibits the poorest performance, while the two GAs significantly outperform all other approaches.

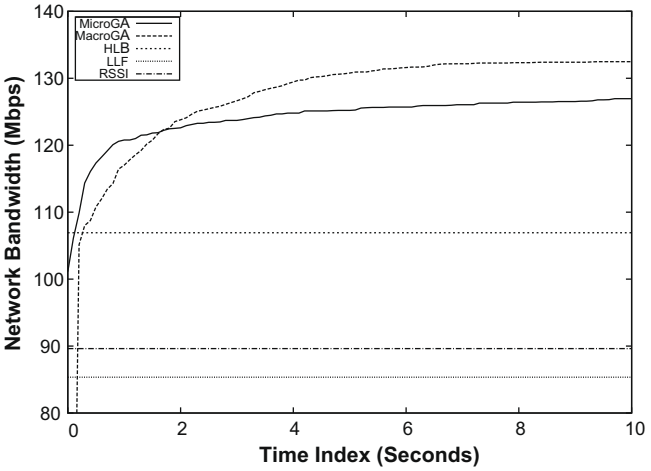

Fig. 3. Performance of all algorithms for a WLAN populated by 50 users.

On average *MacroGA* has achieved a network throughput that constitutes an improvement of: (i) 36% over LLF, (ii) 33% over RSSI, (iii) 20% over HLB and (iv) 4% over *MicroGA*. Therefore, for these settings it represents a very significant upgrade in performance. It is interesting to note that *MicroGA* outperforms *MacroGA* for the first 2 s. After this point *MacroGA* is dominant. The *MicroGA* has a small population size and can evolve it very quickly. Therefore, it can produce good results very quickly. In contrast, the *MacroGA* has a very large population, which evolves at a much slower rate.

The second result, depicted in Fig. 4, compares algorithm performance in a moderately loaded network of 100 users. Again the GAs significantly outperform all other techniques with the *MacroGA* dominant after a few seconds. In contrast to the previous experiment LLF performs better than RSSI. This can be attributed to the fact that LLF is better at distributing the load across APs. As the number of users increase the ability to distribute users amongst APs become critically important. On the other hand RSSI suffers because users connect to the AP with the strongest signal and in an area of user congestion this will result in a large concentration of users all connected to the same AP. The *MacroGA* algorithm outperforms: (i) RSSI by 31%, (ii) LLF by 30%, (iii) HLB by 18% and *MicroGA* by 2%. It is interesting to observe that an increase in user numbers

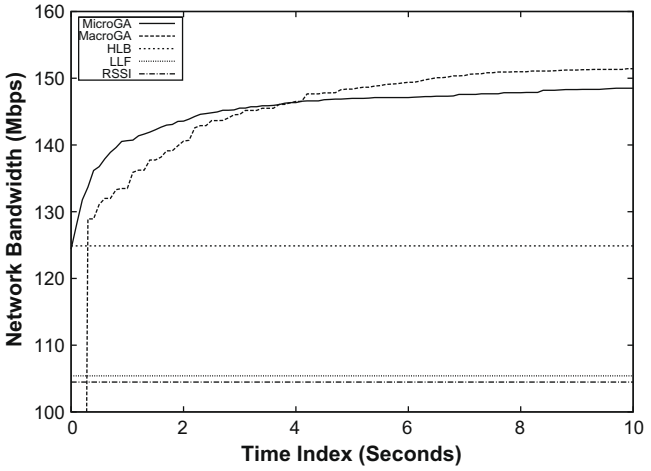

Fig. 4. Performance of all algorithms for a IEEE 802.11 WLAN populated by 100 users.

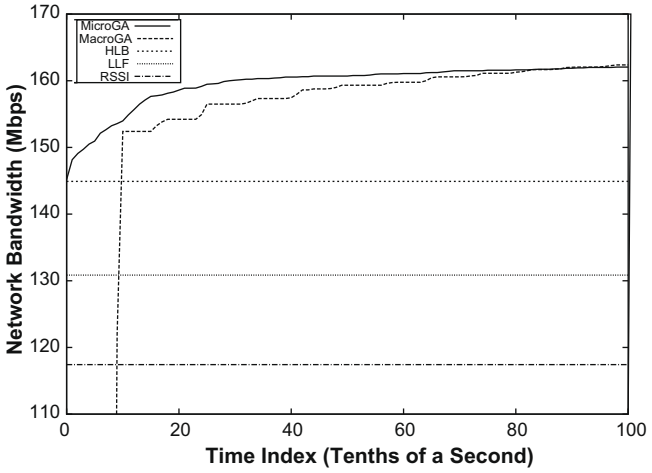

Fig. 5. Performance of all algorithms for a IEEE 802.11 WLAN populated by 250 users.

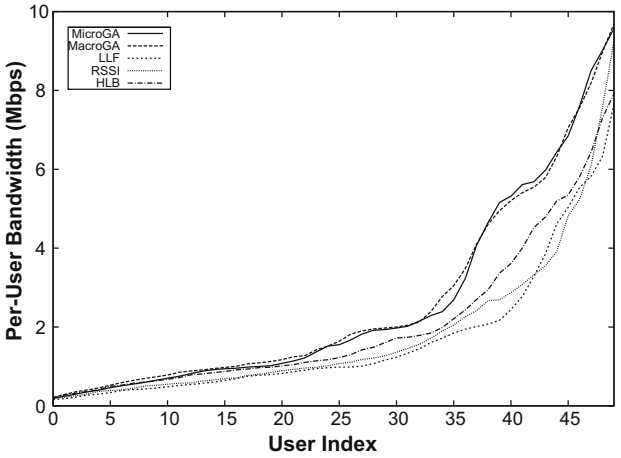

Fig. 6. Bandwidth Distribution of all algorithms for IEEE 802.11 WLAN with 50 users.

improved the performance of *MicroGA* relative to *MacroGA*. This increase leads to a heavier computational load for both of the GAs. However, having such a large population, *MacroGA* is more severely affected. For example, notice that as user numbers increase, *MacroGA* takes progressively longer to produce an initial result. Consequently, *MicroGA* becomes more competitive relative to *MacroGA*.

The genetic algorithms are dominant yet again in Fig. 5, which shows the results obtained for a congested network with 250 users. Also the patterns observed in Fig. 4 are even more pronounced in Fig. 5. The *MicroGA* outperforms *MacroGA* for, on average, the first 8 s. The results also show that the LLF approach significantly outperforms the RSSI approach. Therefore, from the above experiments we can conclude that the proposed genetic load balancing algorithms do represent a very significant improvement over techniques currently used in IEEE 802.11 WLAN networks. Our comparative analysis of the GAs reveal that *MacroGA* has a clear advantage over *MicroGA* as it provides better results for smaller numbers of users and its performance is equal with that of *MicroGA* for large user numbers. However, in a time-critical setting, *MicroGA* has a significant advantage since it outperformed the MacroGA in the early stages of each run. In fact as user numbers increased this advantage became even more pronounced, to the point where it outperformed the *MacroGA* algorithm for over 8 s in a congested WLAN network of 250 users.

5.3. Bandwidth distribution results

This section investigates the distribution of user bandwidth by the proposed algorithms. The objective of the analysis is to check that the GAs do not achieve a high network throughput (as demonstrated in Section 5.2) by increasing the bandwidth of certain categories of users while severely restricting the bandwidth of other categories of users. For example it may reduce the bandwidth of users that already have low bandwidth in order to increase higher end-users. In this section we compare the distribution of bandwidth amongst users by the proposed algorithms with that of the RSSI, LLF and HLB techniques. As in the previous section this evaluation was carried out for population settings of 50, 100 and 250 to simulate varying degrees of network congestion. Due to space limitations we only present the results for a WLAN with 50 users. However, in the context of bandwidth distribution the results obtained for a WLAN with 100 or even 250 users is consistent with that of 50 users.

The distribution of user bandwidth by all algorithms is depicted in Fig. 6. The *Y-axis* represents the per-user bandwidth and the *X-axis* represents the bandwidth sorted in increasing order. The user locations are different at each run, and therefore the bandwidth of the user with the same *x* index actually indicates the average bandwidth of the *x*-lowest bandwidth user in each experimental run. Notice that the distribution of bandwidth by the proposed genetic algorithms represents a improvement over RSSI, LLF and HLB and thus the *i*th ranked user in terms of bandwidth under either GA receives more bandwidth than the *i*th ranked user under the other approaches, for all ranks. For the users with a lower bandwidth index the HLB technique is competitive with the genetic algorithms. However, the HLB's performance degrades and is significantly outperformed by the genetic algorithms for users with a medium to high bandwidth index. Although this is a favourable result for the GAs, it should not be interpreted as meaning that the GAs guarantee an improvement in performance for all users in each experimental run. There may still be a significant percentage of users who experience a loss in bandwidth when using the GA. This of course may not be revealed in Fig. 6 if the percentage increase in performance experienced is much greater than the percentage decrease.

Therefore, we also analysed the percentage of users that experienced an increase and decrease in performance for each experimental run. These figures were then averaged over all 30 experimental runs. Again for reasons of space we confine our analysis to a comparison between *MicroGA* and RSSI, LLF and HLB. The *MicroGA* results are comparable with those of *MacroGA*. The RSSI comparison showed that 57% of users experienced an average increase in bandwidth of 270% when using *MicroGA* over RSSI. Also, 40% of users experienced an average bandwidth decrease of 45%. The remaining 3% retained exactly the same bandwidth. A total of 54% of users experienced an average increase in bandwidth of 473% when using *MicroGA* instead of LLF; 36% of users experienced an average decrease of 54%. The bandwidth of the remaining 10% was unchanged. When HLB is contrasted with *MicroGA* it reveals that 46% of users experienced an average increase in bandwidth of 372%, while 43% of users experience an average decrease in bandwidth of 56%. The remaining 11% had the same bandwidth. The results reinforce the view that HLB is the best of the standard strategies. However, *MicroGA* still outperforms HLB and the average bandwidth increase experienced by a user opting for *MicroGA* is vastly superior to the average decrease.

We were also interested in identifying patterns of behaviour by the *MicroGA*. For example, were there patterns in the way the *Mic-*

roGA treated particular weighted users? Did it reward users of a certain weight more than others? Was there a pattern evident in the way that *MicroGA* distributed users of a certain weight amongst APs? We performed a comprehensive analysis of the experimental data. We looked for patterns and correlations based on user bandwidth, user weight and user locations. In terms of user locations we looked at user distance to connected AP, as well as ratios such as distance to nearest AP relative to distance to second nearest AP.

We were able to educe two main patterns from the experimental data. The first, which was expected, is that *MicroGA* consistently pushes users from the populated APs at the centre of the AP grid to the less utilised peripheral APs. For example, we found that in environments consisting of 50 users the RSSI technique utilised an average of 9 APs, while the *MicroGA* utilised an average of 19 APs. We found that there was no bias in this distribution of users by *MicroGA*; that is, it did not tend to push users of a certain weight out to the periphery more than others.

The other pattern identified was in relation to the percentage of weighted users that could expect to receive an increase in bandwidth if the WLAN operator used *MicroGA* instead of RSSI. While an operator who switches to using the *MicroGA* would see the majority of users receive an increase in bandwidth, we also found that the percentage of users that experience an increase varied depending on the user's weight. A larger percentage of low weighted users receive an increase in bandwidth compared to high weighted users. For example, we found that a larger percentage of users with weight 2 would receive an increase in bandwidth compared with users of weight 3. This was consistent across all weights. However, this pattern did not occur when *MicroGA* was contrasted with LLF or HLB.

6. Conclusions

The objective of this paper is to provide load balancing techniques that improve network throughput and consequently provide customers with a better QoS. Towards that end we proposed two genetic load balancing algorithms called *MicroGA* and *MacroGA*. Empirical evaluation demonstrated that both algorithms outperform the current WLAN techniques in terms of total network throughput and distribution of user bandwidth. We show that on average a user can expect to receive an increase in bandwidth if it uses one of the proposed GAs. The GA performance improvement is achieved without penalising any particular category of user. A comparative analysis of the GAs reveals that *MicroGA* is more applicable to a time-critical load balancing scenario. In future, we aim to investigate the performance of the proposed algorithms when supplemented with greedy initialisation strategies.

Acknowledgement

This work is funded by Science Foundation Ireland under Grant No. 03/CE3/1405 as part of the Centre for Telecommunications Value-chain Research (CTVR).

References

[1] A. Balachandran, G.M. Voelker, P. Bahl, P.V. Rangan, Characterizing user behavior and network performance in a public wireless lan, SIGMETRICS Performance Evaluation Review 30 (1) (2002) 195–205.
[2] Y. Bejerano, S.J. Han, Cell breathing techniques for load balancing in wireless lans, in: INFOCOM 2006, 25th IEEE International Conference on Computer Communications, 2006, pp. 1–13.
[3] Y. Bejerano, S.J. Han, L.E. Li, Fairness and load balancing in wireless lans using association control, in: MobiCom'04: Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, ACM, New York, USA, 2004, pp. 315–329.
[3] M. Buddhikot, G. Chandranmenon, S. Han, Y.W. Lee, S. Miller, L. Salgarelli, Integration of 802.11 and third-generation wireless data networks, in: IEEE INFOCOM, 2003.
[4] J.K. Chen, T.S. Rappaport, G. de Veciana, Iterative water-filling for load-balancing in wireless lan or microcellular networks, in: Vehicular Technology Conference, 2006, pp. 117–121.
[5] Cisco Systems Inc. Data Sheet for Cisco Aironet 1200 Series, 2004.
[6] C.A.C. Coello, G.T. Pulido, A micro-genetic algorithm for multiobjective optimization, in: EMO'01: Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, Springer-Verlag, London, UK, 2001, pp. 126–140.
[7] G. Dozier, J. Bowen, D. Bahler, Solving small and large scale constraint satisfaction problems using a heuristic-based microgenetic algorithm, in: Proceedings of the First IEEE Conference on Evolutionary Computing, 1994, pp. 306–311.
[8] Y. Fukuda, T. Abe, Y. Oie, Decentralized access point selection architecture for wireless lans, in: Wireless Telecommunications Symposium, 2004, pp. 137–145.
[9] D.E. Goldberg, Sizing populations for serial and parallel genetic algorithms, in: Proceedings of the Third International Conference on Genetic Algorithms, 1989, pp. 70–79.
[10] M.T. Hajiaghayi, S.V. Mirrokni, A. Saberi, Cell breathing in wireless lans: algorithms and evaluation, IEEE Transactions on Mobile Computing 6 (2) (2007) 164–178.
[11] Janaki Gopalan, Reda Alhajj, Ken Barker, Discovering accurate and interesting classification rules using genetic algorithm, in: International Conference on Data Mining, 2006, pp. 389–395.
[12] K. Krishnakumar, Micro-genetic algorithms for stationary and non-stationary function optimization, Intelligent Control and Adaptive Systems 1196 (1990) 289–296.
[13] D. Levine, Application of a hybrid genetic algorithm to airline crew scheduling, Computer Operations Research 23 (6) (1996) 547–558.
[14] Q. Ni, L. Romdhani, T. Turletti, I. Aad, Qos issues and enhancements for ieee 802.11 wireless lan, Technical Report INRIA, 2002.
[15] I. Papanikos, M. Logothetis, A study on dynamic load balance for ieee 802.11b wireless lan, in: 8th International Conference on Advances in Communication Control, 2001.
[16] H. Velayos, V. Aleo, G. Karlsson, Load balancing in overlapping wireless lan cells, in: IEEE International Conference on Communications, vol. 7, 2004, pp. 3833–3836.
[17] E. Villegas, R.V. Ferr, J.P. Aspas, Load balancing in wireless lans using 802.11k mechanisms, in: IEEE Symposium on Computers and Communications, 2006, pp. 844–850.