On Optimization of Wireless Mesh Networks using Genetic Algorithms

Rastin Pries, Dirk Staehle, Barbara Staehle, Phuoc Tran-Gia
University of Würzburg
Institute of Computer Science
Chair of Communication Networks
Würzburg, Germany
Email: {pries, dstaehle, bstaehle, trangia}@informatik.uni-wuerzburg.de

***Abstract*—Next generation fixed wireless networks are most likely organized in a mesh structure. The performance of these mesh networks is mainly influenced by the routing scheme and the channel assignment. In this paper, we focus on the routing and channel assignment in large-scale Wireless Mesh Networks to achieve a max-min fair throughput allocation. As most optimization approaches fail to optimize large wireless mesh network deployments, we investigate the usability of genetic algorithms for this approach. The results show the influence of the genetic operators on the resulting network solution and underline the advantages of a genetic optimization when applied carefully.**

***Keywords*-Wireless Mesh Networks, Planning, Optimization, Routing, Genetic Algorithms**

I. Introduction

The complex structure of Wireless Mesh Networks (WMNs) require a careful network planning and optimization. Our goal is to increase the throughput of the complete WMN while still sharing the resources fairly among the nodes. The planning of WMNs is in contrast to traditional wireless networks much more complex. On the one hand, a WMN consists of a multi-hop structure where not only interference on neighboring paths but also self-interference occurs. On the other hand, each node in the network can be equipped with multiple interfaces operating on different channels. The interference problems are covered by using the concept of collision domains. For the routing and channel allocation, an optimization method is required, which is fast enough to optimize even large WMNs. In our previous papers [1], [2], we evaluated the usability of Genetic Algorithms (GAs) for this optimization approach. GAs are based on the idea of natural evolution by simulating the biological cross of genes. Although GAs are generally not able to find the best solution, they provide near-optimal results in relatively small computation time. In this paper, we extend the evaluation by analyzing the influence of the genetic operators during the evolution and by introducing the concept of local optimization.

Our goal is to increase the throughput of the complete WMN while sharing the resources fairly among the nodes. This is achieved by applying a max-min fair share algorithm presented in [3] and by tuning the genetic parameters. A solution is max-min fair if no rate can be increased without decreasing another rate to a smaller value [4]. A max-min fair share algorithm is used instead of proportional fairness because the main goal is not to optimize the maximum overall throughput on the cost of fairness but to ensure a fair resource distribution between the users.

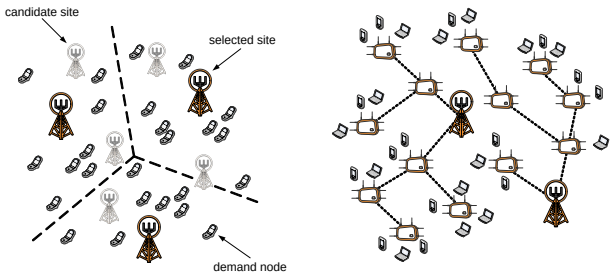
The rest of this paper is structured as follows. In Section II, we first give an introduction to wireless network planning, comparing traditional cellular network planning with wireless mesh network planning. This is followed by a short overview of global optimization techniques, which are applied in the related work section for optimizing WMNs. In Section IV, we describe our genetic algorithms for routing and channel assignment in detail. The performance of different genetic operators is evaluated in Section V and we optimize the genetic algorithm by introducing a local optimization approach in Section VI. Finally, conclusion are drawn in Section VII.

II. Network Planning and Optimization Issues

Network planning and optimization can be done using several techniques. On the one hand, signal quality measurements can be performed, which is very time-consuming and necessitates the access to all areas in which the network should be supported. On the other hand, a demand node concept can be used. This mechanism is often applied to cellular network planning. Furthermore, network planning can be done using an optimization mechanism. Meanwhile, a huge number of optimization techniques have been proposed and we decided to use genetic optimization due to its simplicity and the ability to plan even large networks.

A. Wireless Network Planning

The planning of wireless mesh networks can be applied to a variety of wireless networks, like WiMAX, WLAN, and sensor networks. Although the network technology changes, the planning challenges remain similar. In contrast to traditional cellular network planning, the planning and optimization of WMNs is much more complex. A widely used concept for cellular network planning is the demand node concept introduced by Tutschku [5] and illustrated in Figure 1(a).



(a) Cellular network planning. (b) WMN planning.

Figure 1. Comparison of traditional cellular network planning and wireless mesh network planning.

The algorithm first looks for the demands of cellular services. Therefore, different demographic areas are taken into account. For example, more phone calls occur in urban areas than in rural areas. According to these demographic regions, a different number of demand nodes are set up like shown in Figure 1(a). In addition to the demand nodes, candidate sites for base stations are inserted into the optimization algorithm. As each base station is able to support a fixed amount of users in cellular systems of the second generation, candidate sites are selected for base station placement in such a way that all demand nodes can be served with a certain probability.

In contrast, the planning of WMNs is much more complex. Not only the covered area or number of end users has to be considered, but also the capacity and the interference of the relaying links. The capacity of a link does not only depend on the distance between two mesh points, but also on the interference, which in turn depends on the used channels. Looking again at Figure 1(a), we can see that the channel assignment has to be performed in such a way that neighboring base stations do not use the same channel. In WMNs, such as shown in Figure 1(b), each mesh point can be equipped with multiple interfaces, which can be assigned one channel each.

In addition to the more complex channel assignment in WMNs compared to traditional cellular networks, also the routing has to be considered. In a fixed wireless mesh network where each mesh point is equipped with multiple interfaces, the Modulation and Coding Scheme (MCS), the interference from neighboring nodes, and the number of flows traversing a link have to be taken into account for the routing decision.

B. Global Optimization Techniques

Due to the complexity of routing and channel assignment in WMNs, global optimization techniques are applied. Until now, over 90 different optimization techniques have been proposed, ranging from ant colony optimization to tabu search. We only describe four of them, which are used for the planning and optimization of wireless mesh networks, namely tabu search, branch and bound, simulated annealing, and genetic algorithms.

1) Tabu Search: Tabu search is an extension of the local search technique for solving optimization problems. The algorithm was introduced by Glover in 1986 [6]. It enhances the local search method by using a memory structure. To avoid cycles of the possible solutions found by the algorithm, the solutions are marked as “tabu”. All solutions on the tabu list can not be used for the next iteration step.

The tabu search algorithm starts by using either a random solution or by creating a solution with a heuristic. From this initial solution x, the algorithm iteratively moves to a solution x’ in the neighborhood of x. From all possible solutions in the neighborhood of x, the best one is selected as the new solution if it is not on the tabu list. Afterwards, the tabu list is extended and the algorithm proceeds until a stopping criterion is reached.

2) Branch and Bound: The branch and bound method generally finds the optimal solutions with the disadvantage of being slow. In general, it is a search and comparison of different possibilities based upon partition, sampling, and subsequent upper bounding procedures. The first step, the branch, is used to split a problem into two or more subproblems. The iteration of the branch step creates a search tree. To avoid the calculation of all subtrees, the algorithm uses the bound step. It searches for the first valid solution whose value is the upper bound. All following calculations are canceled if their costs exceed the upper bound. If a new, cheaper solution is found, the upper bound will be set to the value of this new solution. Thus, the branch step increases the search space while the bound step limits it. The algorithm proceeds until either all subtrees have been evaluated or a threshold is met.

3) Simulated Annealing: The goal of simulated annealing is to find a good solution rather than to find the optimal solution like branch and bound. The name of the algorithm comes from metallurgy. Metal is heated up and then cooled down very slowly. The slow cooling allows to form larger crystals, which corresponds to finding something nearer to a global minimum-energy configuration.

When applying simulated annealing for the channel allocation in a WMN, the algorithm starts assigning channels randomly. If a small change in the channel assignment improves the throughput, i.e., lowers the cost or energy, the new solution is accepted and if it does not improve the solution it might be accepted based on a random function. If the change only slightly worsens the solution, it has a better chance to get accepted in contrast to a solution, which heavily decreases the performance. Worse solutions are accepted with a probability given by the Boltzmann factor

$$e^{-\frac{E}{k_B \cdot T}} > R(0,1), \quad (1)$$

where E is the energy, k_B is the Boltzmann constant, T is the temperature, and $R(0,1)$ is a random number in the interval [0,1]. This part is the physical process of annealing. For a given temperature, all solutions are evaluated and then, the temperature is decremented and the entire process repeated until a stable state is achieved or the temperature reaches zero. This means that worse solutions are accepted with a higher probability when the temperature is high. As the algorithm progresses and the temperature decreases, the acceptance criterion gets more and more stringent.

4) Genetic Algorithms: Genetic algorithms are similar to simulated annealing and are also not applied to find the optimal solution but rather good ones. In contrast to the branch and bound method, they are much faster and therefore applicable for the planning and optimization of large wireless mesh networks.

GAs are based on the idea of natural evolution and are used to solve optimization problems by simulating the biological cross of genes. A randomly created population of individuals represents the set of candidate solutions for a specific problem. The genetic algorithm applies a so-called fitness function to each individual to evaluate its quality and to decide whether to keep it in the new population. However, the selection without any other operation will lead to local optima. Therefore, two operators, crossover and mutation, are used to create new individuals. These new individuals are called progenies. Figure 2 shows the influence of crossover and mutation on the fitness landscape of two traits. As mutation is just a swapping of one or two bits, it leads to only small changes in the fitness landscape. The crossover operator instead can lead to complete new solutions as indicated in the figure with the creation of the progeny. Thus, the crossover operator can protect the genetic algorithm from running into local optima, while a mutation is just a small step around a previous solution. Both operators together are used to find a near-optimal solution.

Simulated annealing and genetic algorithms are well suited for the planning of wireless mesh networks. Applying tabu search and the branch and bound algorithm would be too time consuming, especially when considering large WMNs. In the next section, we take a closer look at the work related to WMN planning and optimization where one of the described optimization methods is applied.

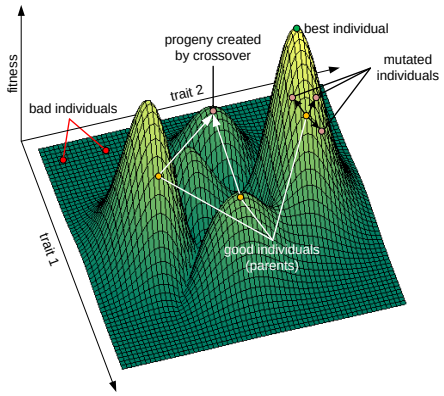


Figure 2. Influences of crossover and mutation in the fitness landscape.

III. Related Work

Wireless mesh networks have attracted the interest of various researchers and Internet providers. Hence, a number of papers have been published on the problem of planning WMNs and estimating their performance. We divide the related work into three parts. The first part shows general WMN planning approaches. In the second part, the work related to channel assignment and routing is presented. Finally, we present papers working with genetic algorithms for planning radio networks.

A. Wireless Mesh Network Planning Using Optimization Techniques

Sen and Raman [7] introduce a variety of design considerations and a solution approach, which breaks down the WMN planning problem into four tractable parts. These subproblems are inter-dependent and are solved by heuristics in a definite, significant order. The evaluations of the presented algorithms show that they are able to generate long-distance WLAN deployments of up to 31 nodes in practical settings.

Other related works [8]–[10] deal with creating a wireless mesh network model, planning its parameters, and evaluating the solutions via linear programming. He et al. [8] propose mechanisms for optimizing the placement of integration points between the wireless and wired network. The developed algorithms provide best coverage by making informed placement decisions based on neighborhood layouts, user demands, and wireless link characteristics. Amaldi et al. [9] propose other planning and optimization models based on linear programming. The aim is to minimize the network installation costs by providing full coverage for wireless mesh clients. Thereby, traffic routing, interference, rate adaptation, and channel assignment are taken into account. Another cost minimizing, topology planning approach is presented by So and Liang [10]. An optimization framework is proposed, which combines a heuristic with Bender's decomposition to calculate the minimum deployment and maintenance cost of a given heterogeneous wireless mesh network. Furthermore, an analytical model is presented to investigate whether a particular relay station placement and channel assignment can satisfy the user demands and interference constraints.

B. Routing and Channel Assignment for Wireless Mesh Networks

One of the first contributions on channel assignment is presented by Raniwala et al. [11], [12]. The channels are assigned according to the expected load evaluated for shortest path and randomized multi-path routing. It is shown

that by using only two network interface cards per mesh point, the throughput increases up to eight times. In contrast to Raniwala et al. [11], [12], Chen et al. [13] do not only consider the expected load for the channel assignment, but also consider the link capacities. Based on the link metrics, called expected-load and expected-capacity, the channel assignment is optimized using simulated annealing.

Further papers based on the paper presented by Raniwala et al. [11] are published by Ramachandran et al. [14] and Subramanian et al. [15]. Both papers take the interference between links into account. The first paper solves the channel assignment using a straightforward approach while the second one uses a tabu search algorithm. Another paper on channel assignment and routing is presented by Alicherry et al. [16]. A linear programming based routing algorithm is shown, which satisfies all necessary constraints for the joint channel assignment, routing, and interference free link scheduling problem. Using the algorithm, the throughput is fairly optimized. The fairness constraint means that for each node the demands are routed in proportion to the aggregate traffic load.

Raniwala and Chiueh [12] and Chen et al. [13] only consider non overlapping, orthogonal channels. Mohsenian Rad and Wong [17], [18] instead also consider partially overlapping channels and propose a congestion-aware channel assignment algorithm. It is shown that the proposed algorithm increases the aggregate throughput by 9.8 % to 11.4 % and reduces the round trip time by 28.7 % to 35.5 % compared to the approach of Raniwala and Chiueh [12].

C. Genetic Algorithms for Radio Network Planning

Genetic algorithms have been used for radio network planning for years [19]–[23]. Calégari et al. [19] apply a genetic algorithm for UMTS base station placement in order to obtain a maximum coverage. It is claimed that the performance of the GA strongly depends on the fitness function. Another paper on UMTS optimization with genetic algorithms was published by Ghosh et al. [20] in 2005. Genetic algorithms are used to minimize the costs and to maximize the link availability of a UMTS network with optical wireless links to the radio network controllers.

Besides Gosh et al. [20], Badia et al. [21] use genetic algorithms for a joint routing and link scheduling for WMNs. The packet delivery ratio is optimized depending on the frame length. It is shown that genetic algorithms solve the studied problems reasonably well, and also scale, whereas exact optimization techniques are unable to find solutions for larger topologies. The performance of the GA is shown for a single-rate, single-channel, single-radio WMN.

Vanhatupa et al. [22], [24] apply a genetic algorithm for the WMN channel assignment. Capacity, AP fairness, and coverage metrics are used with equal significance to optimize the network. The routing is fixed, using either shortest path routing or expected transmission times. An enormous capacity increase is achieved with the channel assignment optimization. Compared to manual tuning, the algorithm is able to create a network plan with 133 % capacity, 98 % coverage, and 93 % costs, while the algorithm needs 15 minutes for the optimization whereas the manual network planning takes hours.

Lee et al. [23] perform an AP assignment for users in smart environments using a genetic algorithm. The AP assignment is optimized in such a way that the load is balanced between the AP and that the bandwidth requirements can be met. The approach is evaluated in a scenario with 16 APs and 70 users. The results show that the load is almost balanced between the APs after 300 generations.

In contrast to the related work, we focus not only on a single-radio or single-rate WMN, but evaluate the performance of a multi-channel, multi-radio, multi-rate WMN using both channel and route assignment. Our genetic algorithm optimizes the throughput while still maintaining a max-min fair throughput allocation between the nodes. In the next section, the complexity of a fair resource allocation in WMNs is described before introducing genetic algorithms and its modifications for the planning and optimization of WMNs.

IV. WMN Planning Using Genetic Algorithms

The objective of this paper is to support the WMN planning process by optimizing the performance of a WMN. With the help of genetic algorithms, near-optimal solutions can be achieved in relatively small computation time. In this section, we show the parameters, which we have to consider and to evaluate in order to achieve a near-optimal WMN solution, meaning that the throughput in the WMN is fairly shared among the mesh points.

A. Problem Formulation

We assume that each mesh point is connected to only one gateway with a fixed routing and we can thus define the mesh network as a directed graph $G(\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is a set of mesh points $n_1, ..., n_V$ and $\mathcal{E} = \mathcal{L}$ is a set of links connecting the mesh points. A subset $GW \subseteq \mathcal{V}$ contains the gateways, which are connected to the Internet. Each mesh point $n_i \in \mathcal{V} \setminus GW$ has a fixed route and gateway to the Internet. The route is denoted as $\mathcal{R}_i$ and consists of a set of links, $\mathcal{R}_i \subseteq \mathcal{L}$. Thus, the mesh points connected to one gateway can be considered as a tree and the complete WMN as a forest.

As we do not have a fully meshed network, a link (i, j) between mesh point i and mesh point j only exists, if a communication between these mesh points is possible within the mesh network. Let $dr_{i,j}$ be the data rate of the link (i, j). The goal is now to optimize the paths from each mesh point $n_i \in \mathcal{V} \setminus GW$ to the gateway so that the throughput in the WMN is fairly shared among the mesh points.

B. Fairness and Capacity in Wireless Mesh Networks

To achieve a fair resource distribution among the mesh points, we use a max-min fair share approach introduced by Bertsekas and Gallager [4]. A solution is max-min fair if no rate can be increased without decreasing another rate to a smaller value. Max-min fairness is achieved by using an algorithm of progressive filling. First, all data rates are set to zero. Then, the data rates of all flows are equally increased until one flow is constrained by the capacity set. This is the bottleneck flow and all other flows have to be faster than this one. Afterwards, the data rates of the remaining flows are increased equally until the next bottleneck is found. This procedure is repeated until all flows are assigned a data rate.

Before assigning the data rates to the flows, the capacity of the network has to be estimated. Therefore, we first have to estimate the link capacities. The capacity of a single link is determined by the pathloss and the Signal to Noise Ratio (SNR). For the pathloss calculation, we use a modified COST 231 Hata [25] pathloss model for carrier frequencies between 2 GHz and 6 GHz. The model is proposed by the IEEE 802.16 working group as the WiMAX urban macrocell model, but is also valid for WLAN mesh networks and is defined as

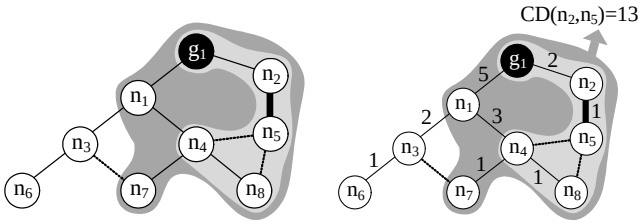
$$PL = 35.2 + 35 \cdot log_{10}(d(n_i, n_j)) + 26 \cdot log_{10}\left(\frac{f}{2}\right). \quad (2)$$

Here, f denotes the operating frequency and d denotes the euclidean distance between mesh points n_i and n_j. The pathloss model is used to calculate the SNR, which is required to determine the maximum achievable throughput. The SNR is calculated as

$$\gamma_{n_i,n_j} = T_x - PL(n_i, n_j, f) - (N_0 + 10 \cdot log_{10}(W)), \quad (3)$$

where T_x is the transmit power, N_0 is the thermal noise spectral density (-174 dBm/Hz), and W is the system bandwidth. Now, the Modulation and Coding Scheme (MCS) is selected with an SNR requirement γ^*_{mcs} that is smaller or equal to the link's SNR γ_{n_i,n_j}. The MCS is chosen in such a way that the frame error rate lies below 1 %. If the SNR requirement for the most robust MCS cannot be met, the two mesh points n_i and n_j are not within communication and interfering range.

Having computed the maximum data rate of each link according to the pathloss, we now have to calculate the capacity of each link taking interference from neighboring mesh points into account. Therefore, we use the concept of Collision Domains (CDs) introduced by Jun and Sichitiu [26]. The collision domain $\mathcal{D}_{i,j}$ of a link (i, j) corresponds to the set of all links (s, t), which can not be used in parallel to link (i, j) because the interference from a transmission on link (s, t) alone is strong enough to disturb a parallel transmission on link (i, j). Figure 3(a) shows the collision domain of link (n_2, n_5). The one-hop collision domain illustrated in light-gray denotes the area for a WLAN-based mesh network without using RTS/CTS. The dark gray area shows the two-hop area where no station can transmit a packet when using RTS/CTS.



(a) One- and two-hop collision domain of link (n_2, n_5).

(b) Link load calculation depending on the carried number of flows.

Figure 3. Collision domain and its link loads.

The nominal load of such a collision domain is the number of transmissions taking place in the collision domain. A transmission $tr_{k,i,j}$ corresponds to the hop from mesh point n_i to mesh point n_j taken by the flow towards mesh point k, i.e., $(i, j) \in \mathcal{R}_k$. The number of transmissions $\lambda_{i,j}$ on link (i, j) corresponds to the number of end-to-end flows crossing it:

$$\lambda_{i,j} = \Big|\{k|(i, j) \in R_k\}\Big|. \quad (4)$$

Figure 3(b) shows the load per link for the same example network as before. Each mesh point on the way to the gateway produces traffic resulting in a traffic load of 5 on the link (n_1, g_1) and a load of 2 on the link (n_2, g_1). Correspondingly, the number of transmissions in collision domain $\mathcal{D}_{i,j}$ is

$$m_{i,j} = \sum_{(s,t)\in\mathcal{D}_{i,j}} \lambda_{s,t}. \quad (5)$$

Thus, the collision domain of link (n_2, n_5) consists of 13 transmissions in total.

In order to fairly supply all mesh points, we share the time resources among all transmissions taking place within the collision domains of the corresponding links. Thereby, we take the rates $dr_{i,j}$ and the number of flows $\lambda_{i,j}$ into account. The throughput $t_{i,j}$ of link i, j is then defined as

$$t_{i,j} = \frac{1}{\sum\limits_{(s,t)\in\mathcal{D}_{i,j}} \frac{\lambda_{s,t}}{dr_{s,t}}}. \quad (6)$$

If we assume that link (n_2, n_5) supports 54 Mbps based on the pathloss and the SNR, the throughput would be 4.15 Mbps due to a collision domain size of 13. However, before setting this throughput to node n_5 we have to follow the principle of max-min fairness.

An algorithm to determine the max-min fair throughput allocation based on the definition of collision domains is given by Aoun and Boutaba [27]. The algorithm iteratively determines the bottleneck collision domain and allocates the

data rates of all flows traversing this domain. If in our example in Figure 3 the link (n_1, g_1) would be the bottleneck, all mesh points traversing the link would be assigned to this throughput, in our case n_3, n_4, n_6, n_7, n_8. As link (n_2, n_5) and link (g_1, n_2) also belong to the collision domain of link (n_1, g_1) but do not transmit over the bottleneck link, the time resources occupied by the bottleneck link are subtracted from the two links.

In the next step of the iteration process, only the remaining collision domains are considered. This way, we calculate the throughput of each flow, which is needed to evaluate the fitness of the WMN. The iteration stops when all flows are assigned. If in our example the next bottleneck collision domain is link (g_1, n_2), the remaining maximum supported rates are assigned to the last two links. Algorithm 1 clarifies the procedure of assigning the rates.

Algorithm 1 Max-min fair resource distribution based on collision domains.

1: $\mathcal{O} = \mathcal{F}$ — all flows are unassigned
2: $\mathcal{L} = \{(i,j) | n_{i,j} > 0\}$ — all active links
3: $p_{i,j} = 1,\ (i,j) \in \mathcal{L}$ — all links have full capacity
4:
5: *Iteration*
6: **for all** links $(i,j) \in \mathcal{L}$
7: $m_{i,j} = \sum_{(s,t) \in \mathcal{D}_{i,j}} \lambda_{s,t}$ — nominal load
8: $t_{i,j} = \frac{1}{\sum_{(s,t) \in \mathcal{D}_{i,j}} \frac{\lambda_{s,t}}{dr_{s,t}}}$ — throughput share per flow
9: **end for**
10: $(u,v) = \arg\min_{(i,j) \in \mathcal{L}} t_{i,j}$ — bottleneck CD
11: $\mathcal{B} = \{k \in \mathcal{O} | \mathcal{R}_k \cap \mathcal{D}_{u,v} \neq \emptyset\}$ — bottleneck flows
12: $b_k = r \cdot t_{u,v}$ for all $k \in \mathcal{B}$ — set bottleneck rates
13: $\mathcal{O} = \mathcal{O} \backslash \mathcal{B}$ — adapt unassigned flows
14: $p_{i,j} = p_{i,j} - \sum_{k \in \mathcal{B}} |\mathcal{R}_k \cap \mathcal{D}_{i,j}| \cdot t_{u,v}$ — adapt free capacity of all CDs
15: $\mathcal{L} = \mathcal{L} \backslash \mathcal{D}_{u,v}$ — adapt active links
16: *Stop criterion:* $\mathcal{O} = \emptyset$

C. Optimization Using Genetic Algorithms

After describing the principle of collision domains and max-min fair throughput allocation, we now explain the workflow of a genetic algorithm in detail. Figure 4 shows the complete procedure of a genetic algorithm for the planning and optimization of WMNs. Firstly, a random population is created with a predefined number of individuals. The fitness of each individual is evaluated using the fitness function and the individuals are ordered according to the fitness value. The best individuals, the elite set, is kept for the new population. Afterwards, the crossover and mutation operator are used to create the remaining number of individuals for the new population. The procedure is repeated until a sufficient solution is achieved. In the following, we explain the steps of our WMN optimization approach in more detail.

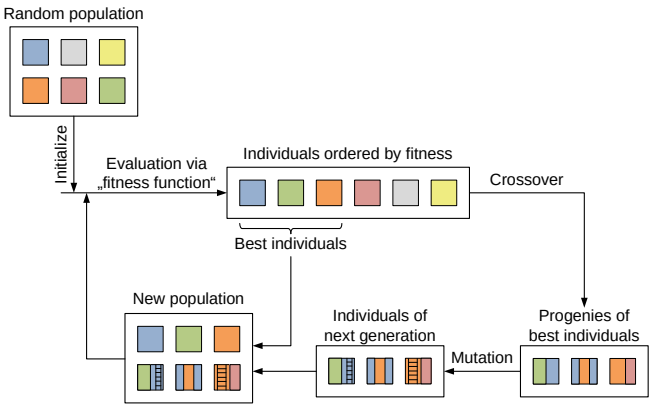
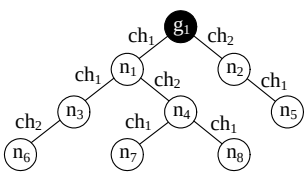


Figure 4. Workflow of a genetic algorithm.

1) Network Encoding: Before going through the steps of the genetic algorithm, the WMN has to be encoded. The encoding must be simple without any redundancy in order not to prolong the runtime of the genetic algorithm. As we assume that each mesh point is connected to only one gateway, the network encoding has to represent a spanning tree with the gateway as root, cf. Figure 5(a). This means that the graph does not contain any cycles and each mesh point has only one route towards the gateway. Such a tree structure can easily be encoded in a list, where the next hop of each mesh point, which the traffic has to take in order to reach the gateway, is stored. This list representation of the example network from Figure 5(a) is shown in Figure 5(b). Considering for example mesh point n_4, the next hop is node n_1 and the next hop of mesh point n_1 is the gateway. Thus, the complete routing of a WMN is handled with a simple list representation.

Besides the routing, we also want to optimize the channel allocation. Although each mesh point can be equipped with several network interface cards, the channel of the link towards the gateway is fixed as shown in Figure 5(a). Thus, the channel allocation can be done in a similar way as the routing. Therefore, the list is extended with one more row, showing the channel of the next hop towards the gateway, cf. Figure 5(b). This simple list represents the tree structure of



(a) Example network.

node	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8
next hop	g_1	g_1	n_1	n_1	n_2	n_3	n_4	n_4
channel	ch_1	ch_2	ch_1	ch_2	ch_1	ch_2	ch_1	ch_1

(b) List representation.

Figure 5. Example network and its list representation.

one gateway and each gateway in the wireless mesh network is encoded in a similar way. The list representation is later used to perform the genetic operations and to evaluate the fitness of the WMN.

2) Evaluation via Fitness Function: The evaluation part of the optimization is the heart of the genetic algorithm. Based on the fitness value, the GA decides, which individuals should be kept in the new population. Hence, it rates the performance of the genes and allows only the best to be replicated.

The fitness of the WMN is estimated using the allocated throughputs of each flow. The fitness function $f(\mathcal{N})$ of the evaluation represents the user satisfaction and the fairness of the resource allocation. Some fitness functions might lead to a complete unfair resource distribution in the WMN. Therefore, we evaluate the performance of several different fitness functions in Section V. Several combinations of the functions $min(\mathcal{R}_\mathcal{N})$, $median(\mathcal{R}_\mathcal{N})$, $mean(\mathcal{R}_\mathcal{N})$, and $var(\mathcal{R}_\mathcal{N})$ are used, which are applied on all routing links of a network solution $\mathcal{N}$. The function $min\,(\mathcal{R}_\mathcal{N})$ calculates for example the minimum throughput of all links used in routing scheme $\mathcal{R}_\mathcal{N}$. We define the following eight different fitness functions:

$$
\begin{aligned}
f_1(\mathcal{N}) &= min(\mathcal{R}_\mathcal{N}) = \text{minimum throughput}(\mathcal{R}_\mathcal{N}) \\
f_2(\mathcal{N}) &= median(\mathcal{R}_\mathcal{N}) = \text{median throughput}(\mathcal{R}_\mathcal{N}) \\
f_3(\mathcal{N}) &= mean(\mathcal{R}_\mathcal{N}) = \text{mean throughput}(\mathcal{R}_\mathcal{N}) \\
f_4(\mathcal{N}) &= min(\mathcal{R}_\mathcal{N}) + \frac{median(\mathcal{R}_\mathcal{N})}{s} \\
f_5(\mathcal{N}) &= mean(\mathcal{R}_\mathcal{N}) - var(\mathcal{R}_\mathcal{N}) \\
f_6(\mathcal{N}) &= min(\mathcal{R}_\mathcal{N}) + \frac{median(\mathcal{R}_\mathcal{N})}{s} + \frac{mean(\mathcal{R}_\mathcal{N})}{|\mathcal{L}|} \\
f_7(\mathcal{N}) &= \sum_{i=0}^{|\tilde{T}|-1} (\left|\tilde{T}\right| - i) \cdot \tilde{T}(i) \\
f_8(\mathcal{N}) &= \sum_{i=0}^{|\tilde{T}|-1} c^{|\tilde{T}|-i} \cdot \tilde{T}(i).
\end{aligned}
$$

The last two functions weight the link throughputs with a factor depending on the corresponding throughput value. Therewith, we aim to achieve a kind of max-min fairness not only with the throughput allocation made by the evaluating algorithm but also with the fitness value from a reasonable fitness function. For this purpose, an ascendingly sorted list $\tilde{T}$ of the throughputs of all routing links in the solution $\mathcal{N}$ is used. Each throughput value from $\tilde{T}$ is weighted with a factor depending on its place in the list, giving more weight to lower positions. This results in a fitness value with which mainly smaller link throughputs are optimized at the expense of higher ones. The parameter c of function $f_8(\mathcal{N})$ is a constant, which we set to 1.5 and s is set to 8 for the experiments in Section V.

3) Selection Principle: After the evaluation of a population, we select a set of solutions, which have the highest fitness of all and keep them in the new generation. This set is called the elite set. In the results section, we vary the size of the elite set in order to see the influence on the solution. As the number of individuals of a population is fixed for all generation steps, the remaining number of individuals are created by crossing and mutating the genes.

The selection of the individuals for applying the genetic operators is thereby based on the fitness and furthermore depends on the number of needed new individuals. Let w be the number of needed new individuals and $s(x)$ be the selection probability for individual x. Then, the number of progenies generated based on individual x are

$$g(x) = \|w \cdot s(x)\|\,. \tag{7}$$

The selection probability $s(x)$ depends on the relation between the fitness of solution x and the sum of all fitness values from the complete population, which means that new individuals are more likely to be created from individuals with a better fitness. This results in

$$s(x) = \frac{f(x)}{\sum_{j=1}^{n} f(j)}. \tag{8}$$

4) Crossover Types: The crossover operator as well as the mutation operator are now applied to the selected number of individuals. For the cross of genes, we use the standard 2-Point Crossover [28] and two other variants, which we especially created for the planning of WMNs, the Cell and the Subtree Crossover.

2-Point Crossover

The 2-Point Crossover is a widely used extension of the 1-Point Crossover. While the 1-Point Crossover changes the list representations of two individuals until a certain point or from a certain point on, the 2-Point Crossover exchanges subsets, which are randomly chosen sublists of the individuals representation, the genotype. Thus, a start and an end point, denoting the range of the sublist, are chosen each time the 2-Point Crossover is applied.

An example of the crossover is shown in Figure 6. The sublists of two individuals should be crossed, namely the routing and channel allocation of mesh points n_2 to n_5. The resulting progenies of the individuals show one characteristic of this reproduction approach. It created solutions, which contain mesh points with no connection to any gateway. This happens due to the unregulated and absolutely arbitrary selection of the gene subset, which is meant to be exchanged.

Looking at the progeny of individual 2, mesh points n_1, n_2, n_6, n_7, n_8 have no connection to any gateway and thus, the crossover results in an unreasonable solution.

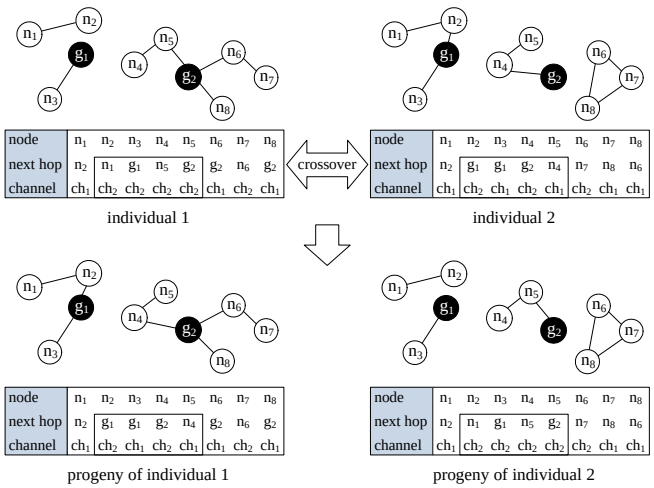


Figure 6. 2-Point Crossover between two individuals.

On the other hand, the 2-Point Crossover has created a reasonable progeny of individual 1.

Since the 2-Point Crossover may lead to unconnected solutions, we have to be careful when evaluating the fitness of the resulting solutions. Thus, we adapt the fitness function to

$$\tilde{f}(\mathcal{N}) = f(\mathcal{N}) - diss(\mathcal{V}), \tag{9}$$

which includes now the $diss(\mathcal{V})$ term denoting the number of nodes with no connection to any gateway. Hence, the throughput contained in $f(\mathcal{N})$ presents the positive costs of the network while $diss(\mathcal{V})$ stands for the penalty costs.

Cell Crossover

In contrast to the 2-Point Crossover, the Cell Crossover does not exchange sublists but complete cells. The crossover operator randomly chooses a gateway and exchanges the entire cell meaning that the routing information as well as the channel allocation is exchanged.

Figure 7 shows an example for the crossover of two solutions. Black nodes denote the network gateways and the light gray areas mark the chosen cell, which is exchanged. In the resulting progenies, the mesh points that have changed their connection are marked dark gray. We can see that not only link connections from mesh points are crossed, but some mesh points are now also connected to other gateways. Mesh points $n_{10}, n_{12}, n_{17}, n_{18}$ are connected to gateway g_2 in the progeny of individual 2 while they were attached to gateway g_1 before. The reason is that the number of mesh points belonging to one cell differ between the individuals. Therefore, we also have to attach unconnected nodes after the Cell Crossover, which can be seen in the progeny of individual 1. In addition to the exchange of routes, the assigned channels are exchanged, which is not shown in the figure for the sake of readability.

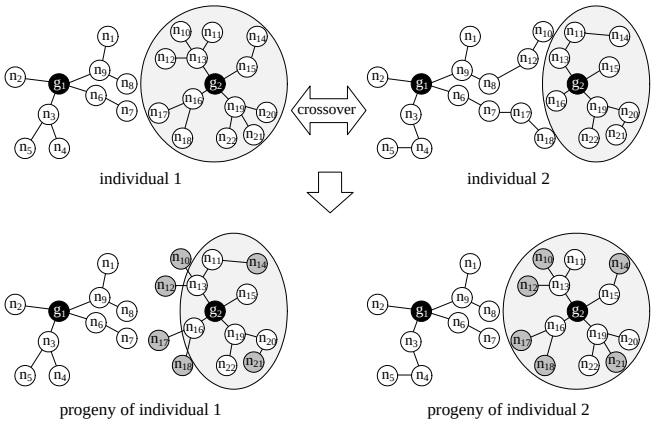


Figure 7. Cell Crossover between two individuals.

Subtree Crossover

The last crossover type is the Subtree Crossover. In contrast to the Cell Crossover, not a complete gateway tree is exchanged but only a subtree. Therefore, the Subtree Crossover chooses mesh points randomly and crosses the entire subtree with the mesh point as root. Similar to the Cell Crossover, the channel allocation is exchanged together with the routing information.

The Subtree Crossover of two subtrees is shown in Figure 8. The chosen mesh points are n_3 and n_{13}. The crossover of subtree n_3 only causes a small change in the tree structure in contrast to the subtree crossover of n_{13}. Here, some nodes of the subtree are connected to different gateways in the two individuals. After the crossover, mesh points n_{10} and n_{12} belong to gateway g_2 in the progeny of individual 2. This reduces the number of long branches of gateway g_1 but there is still potential for further optimization.

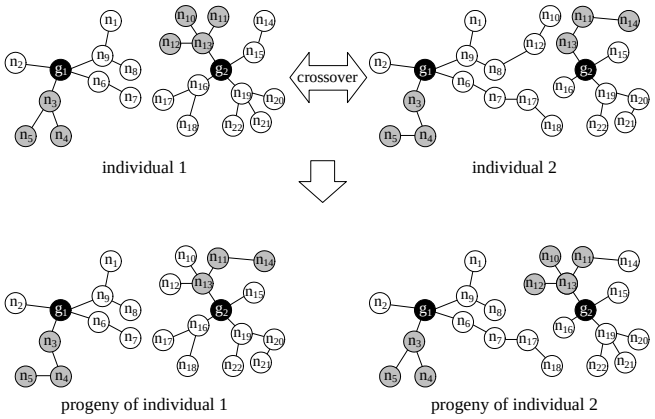


Figure 8. Subtree Crossover between two individuals.

D. Mutation

While the different crossover variants help to avoid running into local optima, the mutation operator increases the

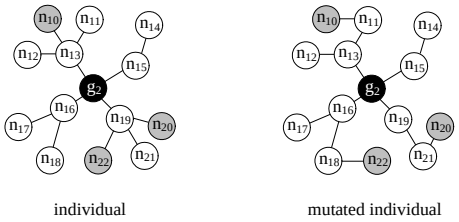


Figure 9. Routing mutation of three mesh points.

performance of WMNs with slightly modifications of the routing structure and channel allocation. For the optimization of WMNs, the number of mutations are chosen based on the scenario size and the mutation of the routing and channel allocation are applied independently from each other.

For the routing scheme, the mutation operator substitutes some randomly chosen positions of the routing code with new information taken from a set of potential neighbors, which would not cause the creation of cycles and would not harm the tree structure of the solution. An example for the mutation of the routing scheme from three nodes is shown in Figure 9. Here, the links towards the gateway of the three gray nodes are mutated. For the channel allocation, the mutation operator randomly chooses a channel from a list of possible channels and substitutes randomly chosen links from the WMN.

According to the workflow diagram shown in Figure 4, the mutation operator is applied after the crossover on the progenies of the crossover. The mutated individuals together with the elite set form then the new population and close the circle of the genetic algorithm.

V. Performance Evaluation

After introducing genetic algorithms in detail and showing our modifications and extensions for wireless mesh networks, we now want to evaluate the performance of the genetic algorithm. The influence of every part of the genetic algorithm's workflow is thereby evaluated separately. First, we take a look at the influence of the fitness function on the resulting solution. Afterwards, the size of the elite set is investigated followed by the population evolution for the three different crossover types. Finally, we show the influence of the two genetic operators crossover and mutation on the resulting network solution.

A. Simulation Settings

For the creation of the results presented in this section, we use the two scenarios introduced in Table I. Although we evaluated a large number of different scenarios, we highlight only the two most different ones here. The first one consists of 2 gateways and 71 mesh points distributed over an area of 2 km x 1.2 km. Thereby, the minimal distance between mesh points is 60 m and between the two gateways it is 700 m. For the sake of readability, we call this scenario G2MP71.

Table I
Simulation Scenarios.

Parameter	Scenario S1	Scenario S2
Topology	G2MP71	G6MP38
Population size	150	
Elite set size	50	
Number of generations	400	
Crossover type	Subtree Crossover Cell Crossover 2-Point Crossover	
Number of crossed subtrees	rand(0,7)	rand(0,5)
Number of mutations	rand(0,20)	rand(0,10)
Fitness function	$f_1(\mathcal{N})$	

The second scenario contains a smaller number of mesh points and a larger number of gateways. We choose this clearly different topology in order to show the influence of the crossover operators depending on the number of mesh points. The 38 mesh points and 6 gateways of the second scenario are allocated in an area of 1.5 km x 1 km. The minimal distance between users is 60 m and between gateways 450 m. We call this scenario G6MP38.

The differences in the settings of the two configurations depend on the used topology of the corresponding scenario. Due to the larger number of mesh points contained in G2MP71, we configure Scenario S1 with more mutations and more exchanged subtrees than Scenario S2. Thereby, we keep the relation between crossover and mutation at a fixed level suitable for the investigation of the genetic operators.

Besides the parameters of the genetic algorithm, the general parameter settings are shown in Table II. These parameters only affect the characteristics of the network connections. The parameters carrier frequency, channel bandwidth, and available channels decide to some extent the performance of the mesh point connections in a network solution but they do not have an impact on the effectiveness of the genetic algorithm. Therefore, we do not consider their impact on the resulting solutions.

Table II
General Parameter Settings.

Parameter	Value
Carrier frequency	3500 MHz
Channel bandwidth	20 MHz
Maximum throughput	67.2 Mbps
Available channels	3500 MHz, 3510 MHz
Antenna power	25 dBm
Pathloss model	WiMAX urban macrocell model

B. Influence of Fitness Function

As the fitness function is the heart of the genetic algorithm, we first take a look on the influence of different fitness functions on the resulting solution. Therefore, eight different fitness functions, described in Section IV, are applied.

Figure 10 shows the throughputs of the mesh points of the best individual after 400 generations of Scenario S1. For the sake of readability, the curves of the eight different fitness functions are shown in two separate subfigures. The x-axis shows the normalized flow IDs, meaning the 71 mesh points sorted by throughput, and the y-axis lists the throughput in Mbps of the flows.

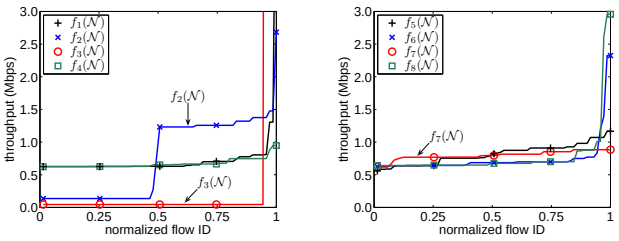
A curve completely parallel to the x-axis would mean a perfect fairness between all flows and a curve whose minimum throughput is above $f_1(\mathcal{N})$ would mean that the solution is max-min fair. This allows to see that the unfairest resource distributions are achieved with the fitness functions $f_2(\mathcal{N})$ and $f_3(\mathcal{N})$.

Optimizing only the median with $f_2(\mathcal{N})$, we do not pay attention to the rest of the throughput allocation. This is why the left part of the $f_2(\mathcal{N})$ curve stays very low. The distribution of $f_2(\mathcal{N})$ also shows that some mesh points have a very high throughput compared to others. This happens accidentally because the fitness function does not control their behavior as it focuses just on the throughput of the median.

Fitness function $f_3(\mathcal{N})$, optimizing only the mean throughput, also results in a very unfair solution. Here, the number of hops towards the gateway are minimized in order to get some nodes with very high throughput, which boost the mean value. In this scenario, four mesh points have a throughput of over 24 Mbps while the throughput of all other flows is about 0.05 Mbps.

All other fitness functions result in a max-min fair resource distribution with a maximized minimal throughput. In the resulting solutions of $f_1(\mathcal{N})$, $f_6(\mathcal{N})$, and $f_8(\mathcal{N})$, some flows have a very high throughput but not at the costs of other flows.

The fairest solution is achieved with fitness function $f_7(\mathcal{N})$ where all flows have a similar throughput of about 0.7 Mbps. The fitness function weights the throughputs of the mesh points. Thereby, smaller throughputs have a stronger influence on the fitness than higher throughputs. This is achieved by multiplying the throughputs with the inverse of the ascendingly sorted flow ID.



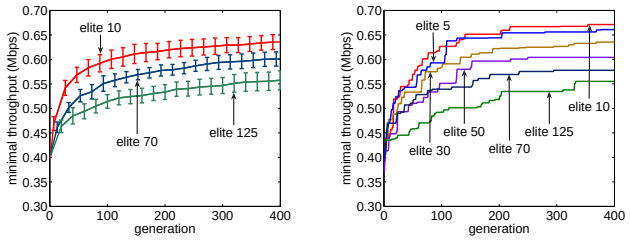
(a) $f_1(\mathcal{N})$ to $f_4(\mathcal{N})$. (b) $f_5(\mathcal{N})$ to $f_8(\mathcal{N})$.

Figure 10. Throughput allocation of the best individual.

C. Elite Set Size

In this section, we examine the impact of the elite set size on the progress of the evolution using Scenario S1 and applying the Subtree Crossover only. Figure 11(a) illustrates the minimal throughput of three different elite set sizes averaged over 15 different initial populations. This time, the x-axis shows the generation number while the y-axis lists the minimal throughputs.

From the figure it can be observed that the best performance is achieved with a small elite set size. On the one hand, a large elite set includes a number of bad individuals, which are kept in the next generation and decrease the minimal throughput. On the other hand, with an elite set size of 125, only 25 new progenies are generated. With this small number of new unexplored genes, the progress of the genetic algorithm slows down, which can be seen on the left side of the figure. Similar solutions compared to an elite set size of 10 might be achieved after several more generations. This means that the larger the elite set size is, the slower is the progress of the genetic algorithm. To prove this statement, we performed the optimization of the same scenario for more different elite set sizes. The results are shown in Figure 11(b).



(a) Three elite set sizes averaged over 15 seeds. (b) Performance of six elite set sizes.

Figure 11. Comparison of different elite set sizes.

The figure reveals almost the same behavior as the previous one. Smaller elite sets cause faster evolution and lead to better solutions. However, a too small elite set size is also bad as the figure shows for an elite set size of 5. With a too small elite set, there might be a discrepancy between the fitness of the elite set and the fitness of new progenies. Thus, the elite set size should be chosen in dependence of the population size.

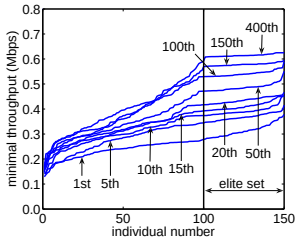
D. Population Evolution

Examining the evolution of the population is an important consideration needed to demonstrate the effectiveness of the genetic algorithm. Observing the evolution of the population with every generation step helps to decide when to terminate the algorithm. When the fitness is not increasing after an additional number of generations, the genetic algorithm can be stopped because either a near-optimal solution is found

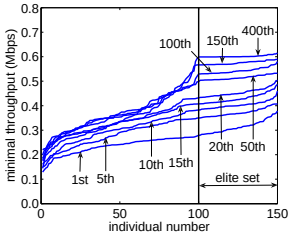
or the genetic algorithm is stuck in a local optimum. As the crossover operator helps to get out of a local optimum, we take a look at the population evolution for all three introduced crossover types.

The results shown in Figure 12 are generated with Scenario S1 from Table I. The x-axis shows the individuals sorted by fitness and the y-axis displays the minimal throughput of each individual. The different curves illustrate the generation progress during the genetic optimization. The elite set size is chosen to be one third of the complete population size.

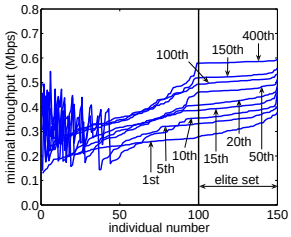
In order to compare all three crossovers, we did not plot the fitness but the minimal throughput on the y-axis. As the penalty costs are included in the fitness function of the 2-Point Crossover, cf. Section IV, the fitness values would be much lower for the 2-Point Crossover. Hence, we consider only the minimal throughputs, which only represent the positive costs. This is also the reason for the strongly varying curves on the left side of Figure 12(c). The individuals have a large minimal throughput but there are a lot of unconnected nodes, which result in a lot of penalty costs and thus in lower fitness.



(a) Subtree Crossover.



(b) Cell Crossover.



(c) 2-Point Crossover.

Figure 12. Generations progress using Subtree, Cell, and 2-Point Crossover.

In all subfigures, we can observe that the higher the generation number is, the smaller is the fitness growth. This slowdown is caused by the similarity of individuals. After several generations, the individuals are quite similar, which means that the crossover does not generate new, unexplored genes. The only possibility to find better solutions is to apply the mutation operator only. Therefore, we introduce the concept of local optimization in Section VI.

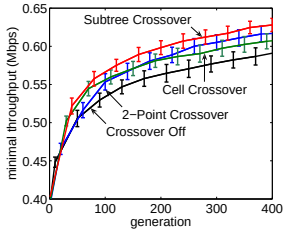
Evaluating the population evolution in other scenarios has shown that it highly depends on the topology structure but a good solution is always found after 400 generations. We tested the performance of Scenario S1 also after 1000 and 1500 generations, but the performance increase was negligible compared to the throughput after 400 generations.

A comparison of the three crossover types shows that the highest minimal throughput after 400 generations is achieved with the Subtree Crossover, followed by the results of the Cell Crossover. The network solution with the worst performance is achieved when applying the 2-Point Crossover. In the next subsection, we want to see if this is an exception or if the Subtree Crossover always leads to the best solutions.

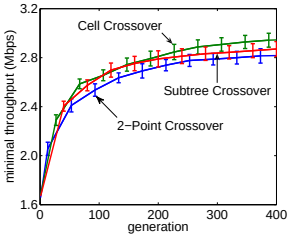
E. Effectiveness of Crossover

In order to show the effectiveness of the crossover type, we compare the performance of the three crossover operators depending on the number of mesh points and gateways in the network. Furthermore, we want to find out if there is an interaction between the efficiency of the crossover types depending on the topology.

The results for both scenarios from Table I are presented in Figure 13. Figure 13(a) shows the evolution of the best individual during 400 generations with different crossover types and for not using the crossover operator at all for Scenario S1. It illustrates the average results of 20 seeds while applying a 95% confidence interval.



(a) Scenario S1.



(b) Scenario S2.

Figure 13. Effectiveness of the crossover operator.

This scenario includes a high number of mesh points, which are distributed in the coverage areas of only two gateways. This results in deep tree structures with long ways over multiple hops towards the corresponding gateway. Such network structures seem to be crucial for the effectiveness of the crossover types. We can observe that the Subtree Crossover leads to a better solution than the other two crossover types. The better performance of the subtree approach is the result of the exchange of small connectivity components, which causes reasonable gene variations without disturbing the tree structure. The other two crossover types show a lower performance whereby the unregulated 2-Point Crossover even outperforms the intelligent Cell Crossover approach. This results from the small number of gateways, which causes the cross of only one cell per new progeny and quickly leads to similar individuals.

The results from Scenario S2 are shown in Figure 13(b). In contrast to the previous scenario, the higher number of available gateways leads to a better efficiency of the Cell Crossover. Moreover, the small number of nodes belonging to one gateway allows a larger variety of individuals. This is due to the fact that small changes in the routing structure cause higher changes in the network performance than in Scenario S1. However, the Cell and Subtree Crossover, which exchange only connectivity components have a better performance than the 2-Point Crossover.

The comparison of the crossover types shows that the crossover operator should be selected based on the considered topology to achieve the best solutions. In the next subsection, we take a look at the influence of the mutation operator on the evolution of the population.

F. Effectiveness of Mutation

The mutation operator causes small changes in the fitness landscape and normally does not help to get out of local optima. However, in the last subsection we have seen that applying only the mutation operator almost increases the performance of the wireless mesh network to the same level as compared to a scenario where both, crossover and mutation are applied. To investigate the influence of the mutation operator, Scenario S1 is considered. Both mutation operations, the routing and the channel mutations, are applied on the progenies of the crossed individuals. The number of routing and channel mutations on each individual are chosen randomly in the interval [0,20]. Figure 14 shows the minimal throughputs during the progress of the genetic algorithm for all three crossover types.

Surprisingly, the performance of the genetic algorithm without mutation is generally low and the genetic algorithm runs into a local optimum after a few generations. For the Cell Crossover, the reason is simple because only 2 gateways are placed in the scenario. The minimal throughputs for the other two crossover variants are higher compared to the Cell Crossover but still way below the throughputs achieved when mutation is used together with crossover. This is because after a few generations, the created individuals are quite similar and thus, the genetic algorithm gets stuck in a local optimum. In contrast, when activating the mutation operator, the fitness of the solution grows even after 400 generations and there is still potential for further evolution.

This shows how crucial the mutation operator is for the evolution of the genetic algorithm. Without using the mutation operator, similar individuals are created by the three different crossovers. The best performance is here seen for the Subtree Crossover as the Subtree Crossover has the largest possibilities to create new genes. The mutation operator instead ensures the creation of new unexplored genes with slight changes in the routing scheme and channel allocation, which fosters the evolution.

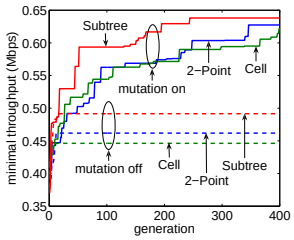


Figure 14. Mutation ON/OFF in combination with three crossover types tested on Scenario S1.

VI. Optimization of the WMN Planning Approach

In the last section, we have seen the influence of the genetic operators on the performance of the resulting wireless mesh network. In this section, we take a look at the influence of the genetic operators in dependence of the GA progress and introduce a local optimization technique to quickly improve the performance of the wireless mesh network.

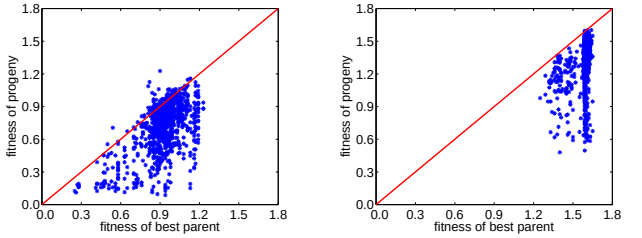
A. Influence of the Crossover on the GA Progress

As crossover operations are very time consuming, we want to see if the crossover types lead to better network solutions during all generations. Therefore, we compare the fitness of the best parent with the fitness of the resulting progeny for early generations as well as for late generations. The genetic optimization runs for 500 generations and the results in Figure 15 show the fitness of the Cell Crossover and Subtree Crossover of 2000 samples.

Looking at Figure 15(a), we can see that about 10 % to 20 % of all crossover operations lead to better progenies. Although this amount seems to be very low, we have to take a look at the exact improvements. One early Cell Crossover increases the fitness from 0.9 to 1.2. This might be a step out of two local optima in the fitness landscape. However, performing a Cell Crossover in the late stages of the genetic algorithm always leads to worse progenies. The reason is simple as a Cell Crossover of two near-optimal solutions are likely to create unreasonable progenies.

When applying the Subtree Crossover, the results are a little bit different as shown in Figure 15(c) and Figure 15(d). Although the percentage of better progenies is similar to the Cell Crossover, the improvements are lower. The reason is that the Subtree Crossover performs only small variations by exchanging subtrees, whereas the Cell Crossover changes two complete cells. However, these small changes also have a bad influence when performing them at the end of the generation process and only one or two progenies are better than their parents.

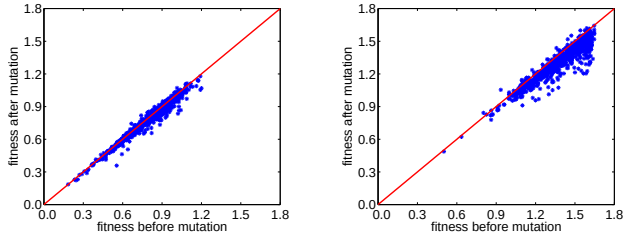
Thus, the amount of crossovers can be reduced with increasing number of generations. Before doing this, we take a look at the influence of the mutation operator in dependence of the number of generations.



(a) Early generation Cell Crossover. (b) Late generation Cell Crossover.

(c) Early generation Subtree Crossover. (d) Late generation Subtree Crossover.

Figure 15. Influence of the crossover on the fitness of the resulting progenies.



(a) Early generation routing mutation. (b) Late generation routing mutation.

(c) Early generation channel mutation. (d) Late generation channel mutation.

Figure 16. Influence of the mutation operator on the GA progress.

B. Influence of the Mutation Operator Depending on the GA Progress

The mutation operator conducts only small modifications of the individuals and it is thus expected that the fitness only slightly changes after the mutation is performed. Furthermore, we want to evaluate if, in contrast to the crossover, the mutation also leads to better results when applied on late generation steps. The results for the two mutation operators, routing and channel allocation, are shown in Figure 16. The plots are generated based on 2000 samples taken at the beginning and at the end of a 500 generation run.

As expected, the change in the fitness value is only small after the mutation is applied. However, the number of improved individuals is larger for both mutation operators compared to the crossover operations. The channel mutation even yields better results in 50 % of all mutations. Although the performance of both mutation operators decreases with an increasing number of generations, still better individuals are achieved in 5 % to 10 % of all mutations, cf. Figure 16(b) and Figure 16(d).

Thus, the mutation operator should be applied during the complete generation process. However, when reducing the number of crossover operations with an increasing number of generations, the number of performed mutations are also decreased. In order to keep the number of mutations, the following mechanism is applied. Firstly, the elite set size is increased with each generation, which means that increasingly more individuals are kept for the following population. This reduces the number of crossover and mutation operations. Secondly, in order to apply the mutation operator during the complete generation process, both mutation operations are performed with each individual of the elite set. If the fitness after the mutation is higher than the fitness before the mutation, the new individual is taken for the next population instead of the old one. If the fitness is worse, the new individual is discarded.

In Figure 17, we compare the fitness values of the ten best individuals in a scenario with an enlargement of the elite set size with increasing generation number and without an enlargement. The values are averaged over 10 simulation runs with 500 generations. Except for the worst of all 10 individuals, the enlargement of the elite set has a positive influence on the fitness. On average, the fitness is increased by 8 %.

Summarizing, a reduction of the number of crossover operations achieved by a stepwise enlargement of the elite set size has a positive effect on the fitness value. In addition, the runtime of the genetic algorithm is reduced due to the smaller number of complex calculations of the crossover operations.

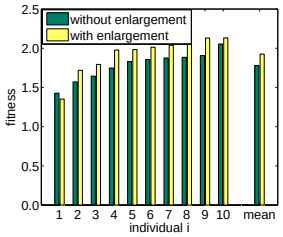


Figure 17. Influence of the enlargement of the elite set.

C. Local Optimization and Population Size

As we have seen in the previous figures, applying the crossover operator on late generations almost always results in worse individuals. However, the mutation operator might improve the individuals because it only slightly changes the individuals. To take advantage of this, we introduce the concept of local optimization. After the normal genetic algorithm finishes, we take the five best individuals of the last generation, copy them three times, and perform several mutations with them. Similar to the previous improvement, the resulting individual is only kept if its fitness value is higher compared to the fitness value before the mutation, else it is discarded. This can be repeated more than a thousand times because the computation time for mutating 15 individuals is negligible.

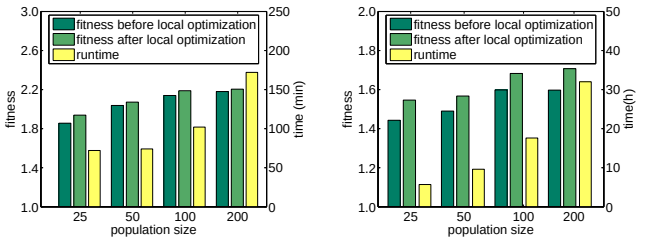
In order to investigate the effect of the local optimization, we take a look at the influence of the population size. The larger the population size, the more new individuals are created per generation resulting in a larger number of good individuals. This means that a large population size has the potential to get to the optimal solution but requires more computation time. In order to find a good population size, we need to look at the fitness of the best individual for a variety of population sizes and compare it to the runtime.

To see the influence of the population size as well as the local optimization, a genetic algorithm run with 500 generations is performed with an additional local optimization of 2500 generations. We investigate the influence on two different scenarios, with different average numbers of mesh points per gateway, and increase the population size from 25 to 200. The results are shown in Figure 18.

The fitness values are averaged values of the best individual over ten runs of the genetic algorithm. The runtime shows the minimal total runtime. In Figure 18(a) the local optimization only slightly increases the fitness of the best individual. However, in a scenario with a larger number of mesh points, the local optimization increases the fitness between 5 % and 7 % depending on the population size, cf. Figure 18(b). The reason is that such a scenario offers more possiblities to assign the routes and channels, which are evaluated in the local optimization process.

Taking a look at the population size, we want to point out that the performance increase is only visible up to a population size of 100. When increasing the population size to 200, a run takes twice as long as a run with a population size of 100, while the fitness increases only by 1.5 % at most. Thus, a population size of 100 is a good compromise between the runtime of the genetic algorithm and the fitness of the resulting individuals.

Summarizing, we want to point out that a local optimization of the best individuals is a good means to get to better solutions without significantly prolonging the runtime of the genetic algorithm. A similar result might be achieved after an additional 500 or 1000 generations but this would take much more time. Also the performance increase by enhancing the population size is negligible and almost doubles the runtime.



(a) Scenario with an average of 18.6 MPs per mesh gateway.

(b) Scenario with an average of 23.6 MPs per mesh gateway.

Figure 18. Relationship between population size, fitness, and runtime.

VII. Conclusion

In this paper, we investigated the usability of genetic algorithms for optimizing wireless mesh networks. Thereby, we showed that the performance of the genetic algorithm depends on the applied fitness function. The fitness function is used to evaluate the resulting network solution. We investigated eight different fitness functions optimizing for example the minimum, mean, and maximum throughput. The results show that the fitness function should be chosen with care because some functions lead to an unfair share of resources. Using a fitness value built on weighted throughputs of all network flows results in the best solutions. In addition to choosing a good fitness function, we illustrated that it is also important to choose the elite set size according to the population size. A small population with a large elite set size often results in a local optimum. The elite set size also has an impact on the required number of generations to get to a good solution. We showed that with an elite set size of one third of the population size, a near-optimal solution is achieved after 400 generations.

Besides the fitness function and the size of the elite set, the genetic operators crossover and mutation have to be carefully applied. We adapted the operators to the requirements of wireless mesh networks and introduced two new crossover variants called Cell and the Subtree Crossover. The evaluation of the influence of these operators revealed that the WMN-specific Cell and Subtree Crossover lead to better solutions compared to the well-known 2-Point Crossover. However, they have to be applied according to the network topology. The Subtree Crossover shows the best performance in scenarios with a large number of mesh points per gateway whereas the Cell Crossover leads to the best solutions in scenarios with a small number of mesh points per gateway.

During the progress of the genetic algorithm, the contribution of the crossover operator to find the optimal solution decreases. After several generation steps, almost no better solutions are achieved by applying the crossover operator.

Here, only mutation leads to a better fitness of the solution. We have shown that a reasonable network optimization is only possible by using mutation. The influence of the mutation operator in combination with all crossover types was tested and it was proven that in all cases it strongly fosters the evolution. Even in late generation steps, the fitness of the resulting solution improved.

In order to benefit from the crossover operator to get out of local optima at the beginning of the evolution process and to still get to better solutions at the end of the genetic optimization, we introduced the concept of an elite set increase and a local optimization. With every generation of the genetic algorithm, the elite set is increased, which decreases the number of crossover and mutation operations. In order to still mutate the individuals, the mutation operator is applied to the elite set and if a better solution is found, it is taken to the next generation. The local optimization is done after the normal generations procedure finishes. Thereby, several mutations of the five best individuals are performed and the resulting individuals are only kept in the new generation of the local optimization if the fitness value is higher compared to the fitness value before the mutation. Using these concepts, the performance of the WMN can be significantly increased with a minimal computational overhead.

Thus, we showed that genetic algorithms are well-suited for the optimization of wireless mesh networks. While other optimization techniques like linear programming fail to optimize large WMNs, genetic algorithms solve the complex structure of WMNs in relatively small computation time. However, the parameters of the genetic algorithm have to be carefully chosen and adapted to the applied topology.

References

[1] R. Pries, D. Staehle, M. Stoykova, B. Staehle, and P. Tran-Gia, "Wireless Mesh Network Planning and Optimization through Genetic Algorithms," in *The Second International Conference on Advances in Mesh Networks (MESH)*, Athens/Glyfada, Greece, June 2009.

[2] ——, "A Genetic Approach for Wireless Mesh Network Planning and Optimization," in *First International Workshop on Planning and Optimization of Wireless Communication Networks (PlanNet2009) in conjuction with the 5th International Wireless Communications and Mobile Computing Conference (IWCMC)*, Leipzig, Germany, 6 2009.

[3] D. Staehle, B. Staehle, and R. Pries, "Max-Min Fair Throughput in Multi-Gateway Multi-Rate Mesh Networks," in *IEEE VTC Spring 10*, Taipei, Taiwan, May 2010.

[4] D. P. Bertsekas and R. G. Gallager, *Data networks*. Prentice-Hall, 1987.

[5] K. Tutschku, "Demand-Based Radio Network Planning of Cellular Communication Systems," in *IEEE Infocom 1998*, San Francisco, CA, USA, March 1998.

[6] F. Glover, "Future paths for integer programming and links to artificial intelligence," *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, 1986.

[7] S. Sen and B. Raman, "Long distance wireless mesh network planning: problem formulation and solution," in *WWW '07: Proceedings of the 16th international conference on World Wide Web*, New York, NY, USA, 2007, pp. 893–902.

[8] B. He, B. Xie, and D. P. Agrawal, "Optimizing deployment of Internet gateway in Wireless Mesh Networks," *Computer Communications*, vol. 31, no. 7, pp. 1259–1275, 2008.

[9] E. Amaldi, A. Capone, M. Cesana, I. Filippini, and F. Malucelli, "Optimization models and methods for planning Wireless Mesh Networks," *Computer Networks*, vol. 52, no. 11, pp. 2159–2171, August 2008.

[10] A. So and B. Liang, "Minimum Cost Configuration of Relay and Channel Infrastructure in Heterogeneous Wireless Mesh Networks," in *Networking*, Atlanta, GA, USA, May 2007, pp. 275–286.

[11] A. Raniwala, K. Gopalan, and T. Chiueh, "Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 2, pp. 50–65, April 2004.

[12] A. Raniwala and T. Chiueh, "Architecture and Algorithms for an IEEE 802.11-based multi-channel wireless mesh network," in *IEEE Infocom 2005*, Miami, FL, USA, March 2005, pp. 2223–2234.

[13] Y.-Y. Chen, S.-C. Liu, and C. Chen, "Channel Assignment and Routing for Multi-Channel Wireless Mesh Networks Using Simulated Annealing," in *IEEE Globecom 2006*, San Francisco, CA, USA, November/December 2006.

[14] K. N. Ramachandran, E. M. Belding, K. C. Almeroth, and M. M. Buddhikot, "Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks," in *IEEE Infocom 2006*, Barcelona, Spain, April 2006.

[15] A. P. Subramanian, H. Gupta, S. R. Das, and J. Cao, "Minimum Interference Channel Assignment in Multiradio Wireless Mesh Networks," *IEEE Transactions on Mobile Computing*, vol. 7, no. 12, pp. 1459–1473, December 2008.

[16] M. Alicherry, R. Bhatia, and L. E. Li, "Joint channel assignment and routing for throughput optimization in multi-radio wireless mesh networks," in *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, Cologne, Germany, 2005, pp. 58–72.

[17] A. H. Mohsenian Rad and V. W. S. Wong, "Joint Channel Allocation, Interface Assignment and MAC Design for Multi-Channel Wireless Mesh Networks," in *IEEE Infocom 2007*, Anchorage, AK, USA, May 2007, pp. 1469–1477.

[18] ——, "Congestion-aware channel assignment for multi-channel wireless mesh networks," *Computer Networks*, vol. 53, no. 14, pp. 2502–2516, September 2009.

[19] P. Calégari, F. Guidec, P. Kuonen, and D. Wagner, "Genetic Approach to Radio Network Optimization for Mobile Systems," in *IEEE VTC Spring 97*, Phoenix, AZ, USA, May 1997.

[20] S. Ghosh, P. Ghosh, K. Basu, and S. K. Das, "GaMa: An Evolutionary Algorithmic Approach for the Design of Mesh-Based Radio Access Networks," in *LCN '05: Proceedings of the The IEEE Conference on Local Computer Networks 30th Anniversary*, Washington, DC, USA, November 2005, pp. 374–381.

[21] L. Badia, A. Botta, and L. Lenzini, "A genetic approach to joint routing and link scheduling for wireless mesh networks," *Elsevier Ad Hoc Networks Journal*, vol. Special issue on Bio-Inspired Computing, p. 11, April 2008.

[22] T. Vanhatupa, M. Hännikäinen, and T. D. Hämäläinen, "Performance Model for IEEE 802.11s Wireless Mesh Network Deployment Design," *Journal of Parallel and Distributed Computing*, vol. 68, no. 3, pp. 291–305, March 2008.

[23] J.-H. Lee, B.-J. Han, H.-J. Lim, Y.-D. Kim, N. Saxena, and T.-M. Chung, "Optimizing Access Point Allocation Using Genetic Algorithmic Approach for Smart Home Environments," *The Computer Journal*, 2008.

[24] T. Vanhatupa, M. Hännikäinen, and T. D. Hämäläinen, "Genetic Algorithm to Optimize Node Placement and Configuration for WLAN Planning," in *4th International Symposium on Wireless Communication Systems, 2007. ISWCS 2007*, Trondheim, Norway, October 2007, pp. 612–616.

[25] E. Damosso and L. M. Correia, *Digital Mobile Radio Towards Future Generation Systems, COST 231 Final Report*. European Commission, 1999.

[26] J. Jun and M. L. Sichitiu, "The Nominal Capacity of Wireless Mesh Networks," *IEEE Communications Magazine*, vol. 10, no. 5, pp. 8–14, October 2003.

[27] B. Aoun and R. Boutaba, "Max-Min Fair Capacity of Wireless Mesh Networks," in *IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS)*, Vancouver, BC, Canada, October 2006, pp. 21–30.

[28] J. H. Holland, *Adaptation in natural and artificial systems*. Cambridge, MA, USA: University of Michigan Press, 1975.