Applying Genetic Algorithms for Multiradio Wireless Mesh Network Planning

Ting-Yu Lin, Kai-Chiuan Hsieh, and Hsin-Chun Huang

***Abstract*—Two main issues that are involved in the performance of multichannel multiradio wireless mesh networks (WMNs) are channel assignment (CA) and multichannel routing (MCR). The joint CA and MCR problem has been proven to be NP-hard. In this paper, we propose to apply genetic algorithms for the CA problem and tackle MCR using linear-programming techniques. Because CA and MCR tend to interact with each other, to reflect such an interplay property, we evaluate the fitness value of a chromosome (certain CA configuration) in our genetic algorithms (GAs) by computing the linear objective function. Therefore, we successfully decouple the two problems and obtain an optimized CA configuration with a corresponding MCR schedule in polynomial time at the WMN planning stage. We demonstrate the detailed genetic evolution processes for three WMNs with different deployment constraints, which serves as a useful guideline on customizing WMNs under various requirements. Simulation results show that the proposed approach effectively increases the network capacity.**

***Index Terms*—Genetic algorithm (GA), IEEE 802.11, linear programming (LP), multiradio wireless mesh networks (WMNs), network planning.**

I. Introduction

DUE TO the physical attenuation and fading effects of radio signal propagation, the wireless capacity is bounded, no matter how much bandwidth is allocated [10]. In multihop wireless networks, the capacity is further reduced due to intraflow and interflow interferences [12]. Under limited physical capacity, the effective throughput performance that is produced by the link and routing layers plays an important role. Wireless mesh networks (WMNs) are a class of multihop wireless networks that are built on immobile nodes (called *mesh routers*) interconnected through wireless links. Among these mesh routers, some can also access the Internet (called *gateway mesh routers* or *Internet gateways*), as shown in Fig. 1. WMN is a promising network architecture used for last-mile broadband Internet access, enterprise wireless backbone, and community Internet sharing, particularly in places where wired infrastructures are not easily deployable. Fig. 1 illustrates an example WMN that serves as the wireless Internet backbone.[1]

Emerging research works on WMNs investigate the possibility of utilizing multiple channels and equipping mesh routers with multiple radios to increase the network capacity. Multiple nonoverlapping (orthogonal) physical channels enable parallel noninterfering transmissions. However, in most practical cases, the number of equipped radios at each mesh router is less than the number of available orthogonal channels. To exploit all available channels, techniques for allocating appropriate sets of channels to mesh routers while maintaining reasonable network connectivity are necessary and referred to as channel assignment (CA) mechanisms. The resulting multichannel network topology after performing some CA mechanism directly affects the routing strategy, which we refer to as the multichannel routing (MCR) problem. As pointed out in [8], CA and MCR are two main issues in multichannel multiradio (MCMR) WMNs. The joint CA and MCR design has been proven to be an NP-hard problem. Several previous works attempt to maximize network throughput by devising CA algorithms (possibly combined with time-divided transmission scheduling) [5], [15], [19], [23]. However, these proposals mainly focus on minimizing interfering links that are theoretically defined by protocol interference model without considering actual traffic relaying loads/needs distributed over wireless links.

As indicated in [20], WMN capacity is highly related to links with more relaying traffic. Fortunately, the survey in [2] reveals that WMN traffic is infrequently changing and can be measured based on some profiling techniques [7], [17]. In addition, WMN traffic distribution is typically skewed, because the majority of data packets are destined for or originated in the Internet through gateway mesh routers [11]. The aforementioned traffic characteristics permit several WMN optimization models, such as [3], [4], [16], [21], and [22], which were proposed to enhance network performance. Under the constraint of a limited total number of radio interfaces, the optimization tool that was used in [3] and [16] is linear programming (LP), whereas [22] is a tree-based heuristic algorithm. On the other hand, the authors in [4] and [21] perform genetic algorithms (GAs) to optimize routing and transmission scheduling. However, [21] does not impose a constraint on the radio number, and [4] focuses on single-channel environments without the need to address the CA problem.

In this paper, we target on IEEE 802.11–based MCMR WMNs with three different resource constraints (defined in

Manuscript received October 11, 2011; revised January 29, 2012 and February 12, 2012; accepted February 14, 2012. Date of publication March 15, 2012; date of current version June 12, 2012. This work was supported in part by the National Science Council of Taiwan, under Grant 99-2221-E-009-006 and by the Ministry of Education through the Program Aiming for the Top University and Elite Research Center Development Plan. The review of this paper was coordinated by Dr. S. Zhong.

The authors are with the Department of Electrical and Computer Engineering, National Chiao Tung University, Hsinchu 300, Taiwan, (e-mail: ting@cm.nctu.edu.tw; kai.chiuan@gmail.com; sth0301.cm98g@nctu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2012.2191166

[1]We will revisit the WMN architecture and elaborate on the graph notations in Section II-A.

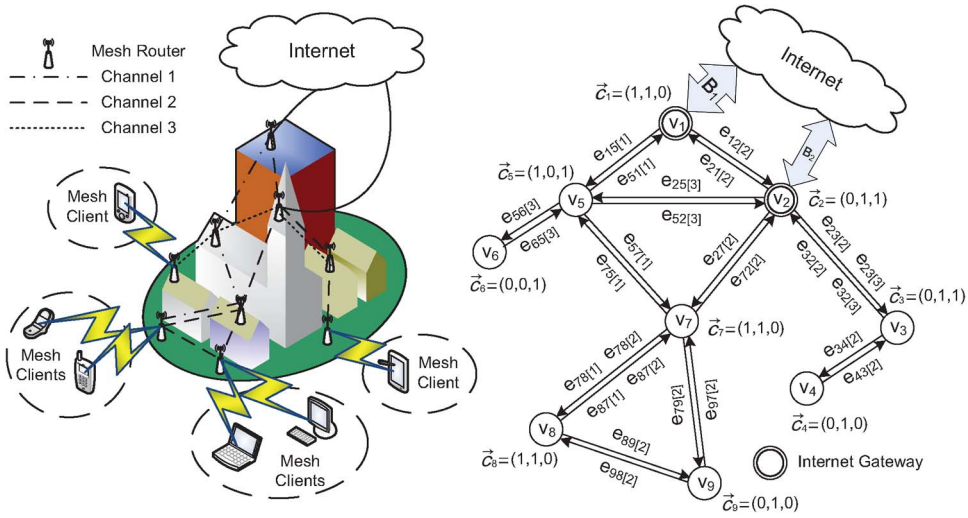

Fig. 1. Example of an infrastructure WMN as the wireless backbone for Internet access and the corresponding network architecture in graph representation.

Section II). Because the joint CA and MCR problem is NP-hard, we do not intend to tackle the whole problem using one single optimization tool. Instead, we observe that CA in MCMR WMNs is a discrete optimization problem that can suitably be modeled by GA.[2] In contrast, the MCR problem, which is involved in dispatching network flows, can be optimized by LP formulations. To reflect the interaction property of the CA and MCR problems, we define the fitness value of a chromosome in GA as the value of a linear objective function. In this manner, we successfully decouple the two problems and obtain an optimized CA configuration with a corresponding MCR schedule in polynomial time at the WMN planning stage. Another contribution that has been made in this paper is that we demonstrate the detailed genetic evolution processes for three WMNs with different deployment constraints, which serves as a useful guideline on customizing WMNs under various requirements.

The remainder of this paper is organized as follows. In Section II, we define three types of WMNs that we plan to optimize, and we also present LP formulations for the MCR problem. Section III provides preliminary knowledge on GAs, which prepares the readers for better comprehension of the following sections. In Sections IV–VI, we detail the encoding, crossover, and mutation operations contained in GAs for three types of WMNs, respectively. Section VII reports the simulative results and performance comparisons. Finally, we conclude this paper in Section VIII.

II. Network Architecture and Linear Formulation

A. Network Architecture

The left side of Fig. 1 illustrates an example operational mesh network. As defined in [2], the network that was constructed by (stationary) mesh routers above the buildings is called an *infrastructure mesh*, whereas the network that was formed by (mobile) mesh clients inside each building is referred to as a *client mesh*. In this paper, we target on the infrastructure mesh network and view the client mesh as an aggregate entity that contributes to the user traffic demands at each mesh router. For brevity, when we mention WMN in this paper, we refer to the infrastructure mesh architecture. Some mesh routers can access the Internet, called gateway mesh routers (or simply gateways). One or multiple gateways may exist and are shared by all mesh routers. Each mesh router is equipped with one or multiple radio interfaces. Multiple nonoverlapping (orthogonal) channels are supported in the network. Together, we define the WMN under study as a MCMR WMN.

We attempt to optimize the network capacity under limited hardware resource constraints at the WMN planning stage. Three types of WMNs with different resource constraints are defined as follows.

- A type-I WMN represents the network where the number of radio interfaces equipped at each mesh router is, respectively, upper bounded. This is the case when WMN deployment occurs in a community where volunteers for hosting the mesh routers already have their own computing devices to act as the mesh routers. Due to inherited variances in computational and communications capabilities that are possessed by mesh routers, the maximum numbers of radios that are supported by distinct mesh routers should be different and separately defined.
- A type-II WMN epitomizes the network where the total number of radio interfaces that are used by all mesh routers is upper bounded. This is the case when all mesh routers have equally powerful capabilities and are possibly made available by the same service provider to deploy a WMN with limited radio resource. In these types of WMNs, the number of radios at each mesh router can be adjusted according to relaying traffic needs and are expected to have better capacity performance than type-I WMNs due to such flexibility.

[2]The adoption of GA for addressing the CA problem in this paper is because GA is a type of heuristic technique that is ideal for handling discrete variables. Our performance results also demonstrate that GA is quite promising for solving complex discrete models that are involved in MCMR WMNs. However, GA may not be the only choice for dealing with such a problem.

- A type-III WMN exhibits the network where the total number of radio interfaces that are used by all mesh routers is upper bounded and the gateway location is undetermined. This is the most flexible type among the three. In the previous two types, we assume wired gateways whose locations are predetermined. Nevertheless, with the advent of various cutting-edge communications technologies such as Worldwide Interoperability for Microwave Access (WiMAX) and Long-Term Evolution (LTE), wireless gateways that can access the Internet become quite feasible. Without the hassle of wired cabling, we further relax the gateway location constraint and attempt to also optimize gateway placement in our type-III GA computations.

To facilitate mathematical modeling, we represent the example network in the form of a graph $G = (V, E)$, where node set V contains all mesh routers, and edge set E includes all wireless links. As depicted in Fig. 1, right, edge $e_{ij[k]}$ indicates the wireless link from node v_i to node v_j over radio channel k. We allow heterogeneous gateways that possess different Internet access capacities. In this figure, nodes v_1 (with gateway capacity B_1) and v_2 (with gateway capacity B_2) act as gateway mesh routers, where $B_1 > B_2$. Define $\vec{c}_i$ as the *channel Boolean vector* that describes the radio configuration at node v_i. Supposing that K nonoverlapping (orthogonal) channels are available in the system, let $\vec{c}_i = (c_{i[1]}, c_{i[2]}, \ldots, c_{i[K]})$, where

$$c_{i[k]} = \begin{cases} 1, & \text{if } v_i \text{ has a radio interface} \\ & \text{operating on channel } k \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Taking mesh nodes in Fig. 1 for example, channel vector $\vec{c}_1 = (1, 1, 0)$ indicates that node v_1 is equipped with two radio interfaces that operate on channels 1 and 2, respectively. Similarly, channel vector $\vec{c}_9 = (0, 1, 0)$ expresses that node v_9 is equipped with one radio that operates on channel 2.

B. Linear Formulation

The determination of channel vector $\vec{c}_i$, where $1 \le i \le |V|$, for the previously defined three types of WMNs is based on GA computations and will be elaborated in Sections IV–VI, respectively. Once channel vector $\vec{c}_i$ is given, the corresponding radio configuration is determined, and we can perform our routing optimization based on the LP model. Our goal is to maximize the network throughput by wisely distributing all user traffic demands among available wireless links (with limited capacities). Several assumptions and definitions that were made in the LP model are provided as follows.

- All user traffic is destined to or originated in the Internet. Each mesh router v_i is associated with an uplink load upper bound u_i^u, a downlink load upper bound u_i^d, an uplink load lower bound l_i^u, and a downlink load lower bound l_i^d. These parameters can be set based on traffic profiling techniques [7], [17] or, perhaps, user-paid prices.
- Define subset $V^g \subseteq V$ of mesh routers as Internet gateways that do not generate user traffic and subset $V^h \subseteq V$ of mesh routers as user hosts with uplink and downlink traffic demands, where $V = V^h \cup V^g$.
- Define data rate $r_{ij[k]}$ as the capacity of link $e_{ij[k]}$, which is obtainable using a channel-probing measurement such as the approach presented in [1]. Due to link asymmetry, it is not necessary that $r_{ij[k]} = r_{ji[k]}$.
- We allow both symmetric and asymmetric gateways. For each gateway router $v_m \in V^g$, let B_m denote the gateway capacity if v_m is symmetric (with capacity shared by both uplink and downlink flows). For an asymmetric gateway router $v_m \in V^g$, use B_m^u and B_m^d to represent the uplink and downlink gateway capacities, respectively.
- Define $\vec{c}_{ij}$ as the connectivity vector that indicates the connection status between nodes v_i and v_j, where $\vec{c}_{ij} = (c_{ij[1]}, c_{ij[2]}, \ldots, c_{ij[K]})$, and $c_{ij[k]} = c_{i[k]} \times c_{j[k]}$. For some channel k, the two nodes can only be connected if both $c_{i[k]}$ and $c_{j[k]}$ are *true* (i.e., both nodes have radios that operate on channel k).
- We adopt the protocol interference model and define set IE_{ij}^k as the set of links in the interfering range of edge $e_{ij[k]}$, where $IE_{ij}^k = \{e_{pq[k]} |$ link $e_{pq[k]}$ interferes with link $e_{ij[k]}\}$. In this paper, we define the interfering range to include all links within two hops of edge $e_{ij[k]}$.

Based on the aforementioned parameters, we now describe the variables that will be computed in our LP model as follows.

- Define λ_i^u as the actual uplink traffic load that was delivered from node v_i and, similarly, λ_i^d as the actual downlink traffic load that is destined to node v_i.
- Define $x_{ij[s,k]}^u$ as the actual uplink traffic that was generated by source node v_s over wireless link $e_{ij[k]}$ and, similarly, $x_{ij[d,k]}^d$ as the downlink traffic that was forwarded to destination node v_d over wireless link $e_{ij[k]}$. Moreover, we define $x_{ij[0,k]}$ as the aggregate traffic load on wireless link $e_{ij[k]}$, where $x_{ij[0,k]} = \sum_{v_s \in V^h} (x_{ij[s,k]}^u \times c_{ij[k]}) + \sum_{v_d \in V^h} (x_{ij[d,k]}^d \times c_{ij[k]})$.
- For each gateway router $v_m \in V^g$, we define the aggregate uplink/downlink traffic through v_m to be g_m^{out}/g_m^{in}, where $g_m^{out} = \sum_{v_s \in V^h} g_{s,m}^{out}$, and $g_m^{in} = \sum_{v_d \in V^h} g_{d,m}^{in}$.

Our ultimate goal is to maximize the mesh network capacity such that the traffic that flows in/out of the set of gateways is the largest, without violating the traffic requirement (upper and lower bounds) of each mesh node. Consequently, the objective function f can be written as

$$f = \text{Maximize} \sum_{v_m \in V^g} (g_m^{\text{out}} + g_m^{\text{in}}) \quad (2)$$

subject to the constraints that were formulated as follows.

1) General Constraints

$$\lambda_i^u \ge l_i^u, \lambda_i^u \le u_i^u, \lambda_i^d \ge l_i^d, \lambda_i^d \le u_i^d$$
$$x_{ij[s,k]}^u \ge 0, x_{ij[d,k]}^d \ge 0$$
$$\sum \lambda_i^u = \sum_{v_m \in V^g} g_m^{\text{out}}, \sum \lambda_i^d = \sum_{v_m \in V^g} g_m^{\text{in}}.$$

2) Gateway Constraint

$$\begin{cases} g_m^{\text{out}} + g_m^{\text{in}} \le B_m, & \text{if uplink and downlink share the bandwidth} \\ g_m^{\text{out}} \le B_m^u, g_m^{\text{in}} \le B_m^d, & \text{otherwise.} \end{cases}$$

3) Link Contention Constraint

$$\sum_{e_{pq[k]} \in IE_{ij}^k} (x_{pq[0,k]}/r_{pq[k]}) \le 1. \tag{3}$$

Table I summarizes the notations that were used in LP constraints and flow conservation equations. Note that we maximize the network throughput by enabling simultaneous transmission/receiving over noninterfering channels. The resulting traffic distributions that were computed by our LP model may split traffic loads over multiple paths (through multiple gateways) for a single flow. Such multipath forwarding with multigateway association behavior can result in better balanced capacity share and be realized by exercising certain traffic engineering mechanisms, as indicated in [14].

TABLE I
SUMMARY OF NOTATIONS THAT WERE USED THROUGHOUT THIS PAPER

Notation	Given Parameters in Our LP Model
u_i^u	Maximum supported uplink traffic load at node v_i (upper bound)
l_i^u	Minimum supported uplink traffic load at node v_i (lower bound)
u_i^d	Maximum supported downlink traffic load at node v_i (upper bound)
l_i^d	Minimum supported downlink traffic load at node v_i (lower bound)
$r_{ij\,[k]}$	Capacity (data rate) of wireless link $e_{ij\,[k]}$ over channel k
B_m	Capacity of (symmetric) gateway node v_m
B_m^u	Uplink capacity of (asymmetric) gateway node v_m
B_m^d	Downlink capacity of (asymmetric) gateway node v_m
K	Number of non-overlapping channels
$\vec{c}_i$	Channel boolean vector of node v_i where $\vec{c}_i = (c_{i[1]}, c_{i[2]}, \ldots, c_{i[K]})$ with $c_{i[k]} = \{0, 1\}$

Notation	Unknown Variables to be Determined by Our LP Computations
λ_i^u	Actual uplink traffic load delivered from node v_i
λ_i^d	Actual downlink traffic load destined to node v_i
$x_{ij[s,k]}^u$	Amount of uplink traffic generated by source node v_s over wireless link $e_{ij\,[k]}$
$x_{ij[d,k]}^d$	Amount of downlink traffic forwarded to destination node v_d over wireless link $e_{ij\,[k]}$
g_m^{out}	Amount of traffic flowing out of gateway node v_m
g_m^{in}	Amount of traffic flowing back into gateway node v_m
f	Objective function defined as $f = Maximize \sum_{v_m \in V^g} (g_m^{out} + g_m^{in})$

Notation	Parameters Used by Our Genetic Algorithm
T	Maximum allowable iterations of performing genetic evolutions
$P(t)$	Population pool at time t $(0 \le t < T)$
Q	Population size (number of chromosomes included in $P(t)$)
q_i	The i^{th} chromosome contained in a population $(1 \le i \le Q)$
L	Length of the chromosome (number of digits contained in a chromosome)
f_i	Fitness value associated with chromosome q_i (value of the objective function output from our LP computations based on the radio and channel configuration defined by q_i)
d_i	The i^{th} random number generated from $(0, \sum_{j=1}^{Q} f_j)$ in the roulette wheel selection procedure
p_c	Probability associated with the crossover operation ($\lceil Q \cdot p_c/2 \rceil$ pairs of parent chromosomes selected to generate offspring chromosomes)
p_m	Probability associated with the mutation operation (likelihood of a gene to mutate)
n_i	Maximum number of radio interfaces allowed at node v_i (Type-I constraint)
N	Total number of radio interfaces allowed in the network (Type-II / Type-III constraint)
$\lvert V^g \rvert$	Number of supported gateway mesh routers (Type-III constraint)

III. GENETIC ALGORITHM PRELIMINARIES

GA is a class of computational mechanism that was inspired by natural evolution, a biological process in which stronger (better) individuals are likely to survive in a competitive environment [6], [9]. GA encodes a potential solution to a specific problem on a chromosome-like data structure.[3] A chromosome is represented by a string of values (genes) to indicate a set of discrete variables (parameters). A *fitness value* that is associated with each chromosome is used to reflect the degree of goodness for solving the problem. In a broader usage, GA is a population-based model that exercises selection, crossover (recombination), and mutation operators to generate new sample points in a search space. During each cycle of genetic evolution, chromosomes with higher fitness values have better chance of surviving in the subsequent generations, emulating the survival-of-the-fittest phenomenon in nature.[4] The basic framework and general applications of GA are outlined in [13] and [18].

In this paper, we formulate a chromosome as one possible radio and channel configuration in a MCMR WMN. The relationship of essential building blocks, including selection, crossover (recombination), and mutation operators, in a cycle of genetic evolution is illustrated in Fig. 2. We define the maximum number of evolution cycles (computational runs) as T. At arbitrary time t, $0 \le t < T$, Q chromosomes (possible

[3]Suppose that substring $str = (1, 3)$ denotes the radio and channel configuration on some mesh node v that indicates two radio interfaces that bind to channels 1 and 3, respectively. Following the same rule on generating substrings for all mesh nodes in a WMN, the GA encoding process basically combines all substrings to form a full string, called a chromosome, that represents the complete radio and channel configuration for the target WMN. More detailed examples can be found in Sections IV-A–VI-A.

[4]In each evolution cycle, the selection routine acts as the process of duplicating chromosomes with quantities proportional to their fitness values. For example, in the roulette wheel selection procedure in Fig. 2, chromosome q_1 is duplicated three times due to its larger occupancy ratio (better fitness value). As a result, $P(t)$ contains three copies of q_1 (biologically better chromosome). Based on $P(t)$, the crossover operation recombines certain chromosomes (determined by probability p_c) to produce a new offspring generation. Finally, the mutation operation performs on individual genes (with a small probability p_m) to imitate the infrequent yet inevitable event of biological genes mutating (genetic changes) in living creatures.

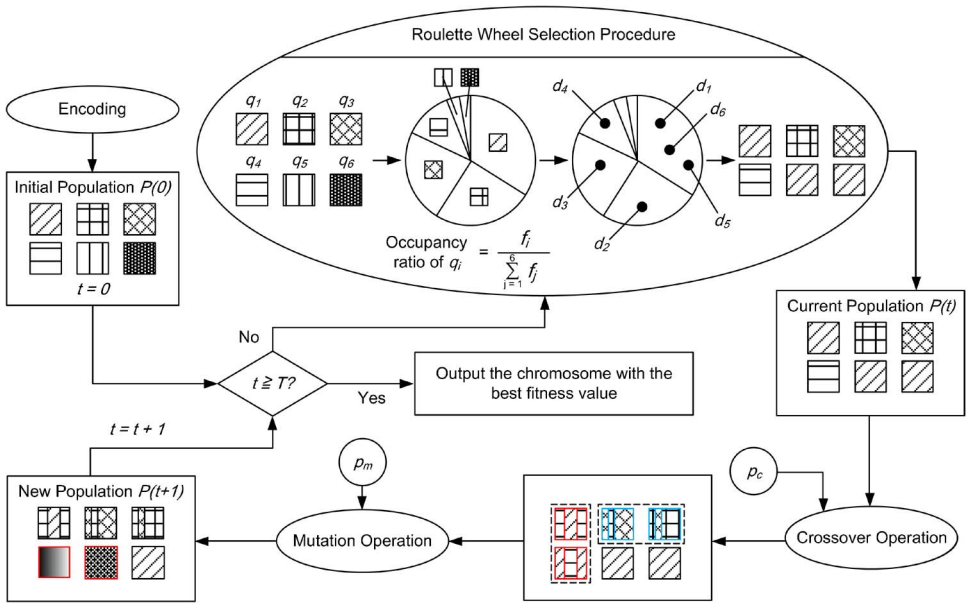

Fig. 2. Genetic evolutions that we adopt in finding the best radio and channel configuration.

solutions) are contained in the population pool $P(t)$. For any population pool, we denote the ith chromosome as q_i associated with a fitness value f_i, and the (string) length of each chromosome is represented by L. After generating the initial population $P(0)$ with Q randomly chosen chromosomes ($Q = 6$ in Fig. 2), we perform the roulette wheel selection procedure. The roulette wheel procedure is a proportionate-selection mechanism that emulates the behavior of shooting darts on a roulette wheel whose space is divided in proportion to the fitness values of chromosomes, as depicted in Fig. 2. Fitness is defined as the value of our LP objective function, given a certain radio and channel configuration specified by a chromosome. Randomly generate a number d_i from the range $(0, \sum_{j=1}^{Q} f_j)$ Q times. Apparently, the chromosomes with larger occupancy ratios will more frequently be selected (with higher probabilities). For all three types of WMNs under investigation, we adopt the same roulette wheel procedure as the selection mechanism. Detailed operations of the roulette wheel selection procedure are provided in Algorithm 1.

Algorithm 1: Roulette wheel selection procedure at time t

1: $sum = 0$;
2: **for** $(i = 1; i \leq Q; i++)$ **do**
3: $\quad sum = sum + f_i$;
4: Randomly generate Q numbers $d_1, \ldots, d_Q$ from the range $(0, sum)$
5: **for** $(i = 1; i \leq Q; i++)$ **do**
6: $\quad index = 0$;
7: $\quad$ **for** $(j = 1; j \leq Q; j++)$ **do**
8: $\quad\quad index = index + f_j$;
9: $\quad\quad$ **if** $(index \geq d_i)$ **then**
10: $\quad\quad\quad$ Chromosome q_j is selected into population $P(t)$;
11: $\quad\quad\quad$ break;

After performing the roulette wheel selection procedure, the crossover operation[5] comes into play. A probability p_c is defined for selecting a subset of chromosomes to generate offspring chromosomes in the crossover operation, where p_c is typically set between 0.6 and 1.0 [18]. In the mutation operation,[6] a small probability p_m (typically less than 0.1) is used to control the likelihood of altering an individual gene after completing the crossover operation. For the three types of WMNs, the required crossover and mutation processes are distinct, which will be, respectively, elaborated in Sections IV–VI. In addition, due to different resource constraints, the encoding scheme also varies in the three types of WMNs. A summary of notations that were used in our GA computations is provided in Table I.

Note that a number of variations with structural modifications to GA building blocks in the evolution cycle are possible. Depending on the desired performance and acceptable computation time, researchers may design their own GA-based evolution mechanisms. In this paper, our goal of GA computations is to produce an optimized radio and channel configuration for the CA problem. Given Q chromosomes, T maximum allowable evolution runs, and $t(LP)$ consumed time in LP computations for calculating the fitness value of each chromosome, our GA computational complexity can be approximated by $Q \times T \times t(LP)$. With the easy availability of current low-cost yet fast-speed small computers, such computation time is quite acceptable at the WMN planning stage. After the WMN has been deployed, the adjustment of radio configuration is not desirable, but adaptations of routing traffic distributions are possible by executing LP recomputations from time to time.

[5]The crossover operation is utilized to generate offspring chromosomes (new sample points) by performing certain genetic shuffling on parent chromosomes. These offspring chromosomes share genetic similarities with their parents but form a different generation.

[6]The mutation operation attempts to bring solutions out of local optimal points of the search space. Similar to the biological gene alteration process, the GA mutation operator manipulates each gene for all chromosomes in $P(t)$.

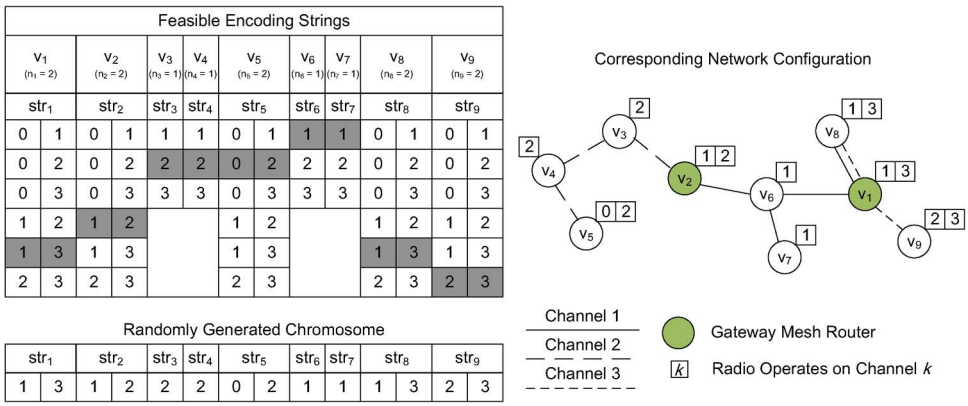

Fig. 3. Encoding process, random generation of a feasible chromosome, and the corresponding radio and channel configuration in a type-I WMN, with $K = 3$.

IV. Wireless Mesh Networks With the Per-Node Radio Number Constraint (Type I)

In a type-I WMN, the maximum number of radio interfaces that can be allowed at each mesh router is restricted. This case generally occurs when WMN deployment takes place in a community where volunteers for hosting the mesh routers already have their own computing devices with varying computational and communications capabilities. Recall that K nonoverlapping (orthogonal) channels can be utilized in the system. For any mesh node v_i, $v_i \in V$, we define parameter n_i to denote the maximum number of radio interfaces allowed at node v_i, where $1 \leq n_i \leq K$. This n_i parameter is referred to as the *per-node radio number constraint*, and the values of $n_1, n_2, \ldots, n_{|V|}$ are given as inputs in our type-I GA computations.

A. Integer-Encoding Process

Fig. 3 shows a sample type-I WMN that comprises nine mesh nodes $(v_1, v_2, \ldots, v_9)$ with respective radio number constraints $(n_1 = 2, n_2 = 2, n_3 = 1, \ldots, n_9 = 2)$. For any mesh node v_i, $v_i \in V$, define substring str_i as the encoding of some feasible radio and channel configuration at node v_i. Substring str_i contains n_i digits with *nonnegative integer* values in the range $[0, K]$. Each digit with value k represents a radio interface that operates on channel k, where $k = 0, 1, 2, \ldots, K$. When k takes on the value 0, it implies that the radio interface is *not* used. For example, when $str_1 = (1, 3)$, as shown in Fig. 3, it indicates that mesh node v_1 is equipped with two radio interfaces that operate on channels 1 and 3, respectively. To achieve feasible radio and channel encodings for any type-I mesh router v_i, the following three general rules need to be followed.

- It is *not* necessary to use up all n_i radios, but *at least* one radio that binds to some valid channel (with nonzero value) should exist. In other words, substring $str_i = (0, 0)$, with both digits set to 0, is an *infeasible* encoding for mesh node v_i.
- To avoid cochannel interference, all nonzero digits should take on different values. In particular, a feasible encoding will *not* bind any two radios equipped at mesh node v_i to the same channel. That is, encoding substrings $str_i = (1, 1)$, $str_i = (2, 2)$, and $str_i = (3, 3)$ are all *infeasible* for node v_i.
- Because the order of arranging radios is irrelevant, we view substrings (1, 3) and (3, 1) as the same encoding. To avoid duplicating options, we only include substrings with digit values arranged in incremental order. As a result, substring $str_1 = (1, 3)$ is included in the pool of feasible encodings for node v_1, whereas $str_1 = (3, 1)$ is regarded as *infeasible* and excluded from the pool.

Consequently, for any mesh node v_i, the total number of feasible radio and channel configurations, based on the aforementioned rules, can be derived by computing $C_1^K + \cdots + C_{n_i}^K = \sum_{j=1}^{n_i} C_j^K$. Taking node v_1 in Fig. 3 for example, there are $\sum_{j=1}^{n_1} C_j^K = \sum_{j=1}^{2} C_j^3 = 6$ feasible combinations for substring str_1 encodings, including (0, 1), (0, 2), (0, 3), (1, 2), (1, 3), and (2, 3). Once all feasible substring encodings are available, we randomly combine these substrings to form a chromosome with length $L = \sum_{i=1}^{|V|} n_i$, as shown on the lower left of Fig. 3.

B. Initial Population Formation

For each constructed chromosome q, the type-I GA obtains the fitness value f based on LP calculations. If f is solvable with a positive returned value, then the chromosome q is selected into the initial population pool $P(0)$. Otherwise, the current chromosome q is dropped, and another chromosome is constructed for selection consideration. This process continues until either Q chromosomes have successfully been selected or the maximum M chromosomes have been tested, where $M \geq Q$. When the process terminates, if the size of $P(0)$ is zero (none of the constructed M chromosomes have positive fitness values), then the type-I GA declares No_Solution. Otherwise, if the size of $P(0)$ is nonzero but smaller than Q (less than Q feasible chromosomes have been selected), then the type-I GA simply keeps duplicating the latest discovered feasible chromosome until $P(0)$ contains *exactly* Q chromosomes. The formed initial population $P(0)$ then acts as the *very first generation* that kicks off the subsequent evolving steps, as shown in Fig. 2.

C. Crossover and Mutation Operations

Two fundamental operations that are involved in the evolution process are crossover and mutation. The purpose of

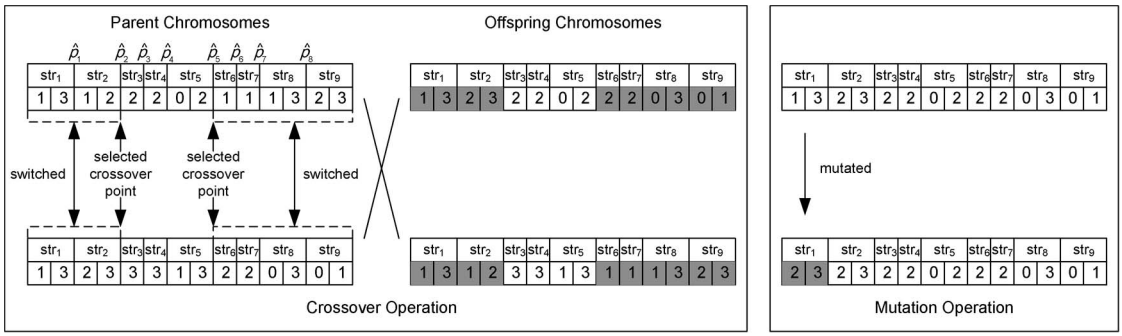

Fig. 4. Type-I crossover operation on a pair of parent chromosomes and type-I mutation operation on a given chromosome (with the shaded parts being the modified genes).

crossover operation is to generate offspring chromosomes (new sample points) by performing certain genetic recombination on parent chromosomes. These offspring chromosomes share genetic similarities with their parents but form a different generation. In our GA model, we define a probability p_c and select $\lceil (Q \cdot p_c)/2 \rceil$ pairs of parent chromosomes from the current population $P(t)$ that is output from the roulette wheel selection procedure. All chromosomes in $P(t)$ are randomly selected with equal probability and cannot repeatedly be selected. For any selected pair of parent chromosomes, we adopt the *two-point crossover* operation, as illustrated on the left side of Fig. 4. To avoid scrambling the different radio number constraints that are imposed on type-I mesh routers, we limit the crossover points to occur only at the joints of any two adjacent substrings in a chromosome, denoted as $\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{|V|-1}$. The two-point crossover operator randomly generates two distinct cutoff points $\hat{p}_a$ and $\hat{p}_b$, where $1 \leq a < b \leq |V| - 1$ and switches the digits before $\hat{p}_a$ (here, $a = 2$) and after $\hat{p}_b$ (here, $b = 5$) of the parent chromosomes to form offspring chromosomes. The created offspring chromosomes then replace their parents in population $P(t)$. Chromosomes that were not selected for the crossover operation remain in $P(t)$.

After the crossover operation, GA enters the mutation phase. The purpose of performing mutation is to avoid solutions being stuck in local optimal points of the search space. In a type-I WMN, we view each substring in a chromosome as a single gene. Similar to the biological gene alteration process, the GA mutation operator examines each substring (gene) for all chromosomes in $P(t)$. Define a small probability p_m as the likelihood of a gene to mutate. If a substring (gene) is determined to mutate, then it will be replaced with a new substring that was randomly chosen among all other feasible encodings. As illustrated in Fig. 4, right, substring $str_1 = (1, 3)$ mutates to a different but feasible substring (gene) $str_1 = (2, 3)$, which is randomly selected from the table of feasible encodings at node v_1, as summarized in Fig. 3, upper left. When the mutation operation completes, a new population $P(t + 1)$ is now formed.

Before the newly generated population further evolves, GA updates the fitness values of all Q chromosomes (possible solutions) contained in $P(t + 1)$. For any chromosome q_i in $P(t + 1)$, GA computes the associated fitness value f_i based on the LP model. In case LP returns No_Solution for q_i, GA simply sets $f_i = 0$ (indicating the infeasibility of q_i). Chromosomes with fitness values set to 0 will be filtered out by the roulette wheel selection procedure in the next iteration. When the fitness updating completes, population $P(t + 1)$ proceeds, and the evolving process starts over again. Algorithm 2 provides the detailed pseudocode on our crossover and mutation operations for type-I WMN planning.

Algorithm 2: Type-I crossover and mutation operations for a given population $P(t)$.

```
// Crossover operation
Define crossover points p̂_1, p̂_2, ..., p̂_{|V|-1} at the joints of any two adjacent substrings in a chromosome;
Randomly select ⌈(Q · p_c)/2⌉ pairs of parent chromosomes from P(t);
for (each pair of parent chromosomes) do
    Randomly generate two crossover points p̂_a and p̂_b, with 1 ≤ a < b ≤ |V| − 1;
    Switch the digits before p̂_a and after p̂_b of the parent chromosomes to form two offspring chromosomes;
    Replace the parent chromosomes with their offspring chromosomes in P(t);
// Mutation operation
for (i = 1; i ≤ Q; i++) do
    for (each substring in q_i) do
        Randomly generate a number x from (0, 1);
        if (x ≤ p_m) then
            Replace the current substring with a new substring that was randomly chosen from all other feasible encodings;
// Now, a new population P(t + 1) is formed and requires postprocessing before entering the next iteration
for (i = 1; i ≤ Q; i++) do
    Compute the fitness value f_i for chromosome q_i in P(t + 1) based on the LP model;
    if (LP returns No_Solution) then
        f_i = 0;
```

Ultimately, the obtained best chromosome (with the highest fitness value) represents an optimized radio and channel configuration, and the mapping of a chromosome onto the corresponding network setting can be found on the right side of Fig. 3. For the obtained best chromosome in a type-I WMN, we can easily transform the chromosome into corresponding channel vectors

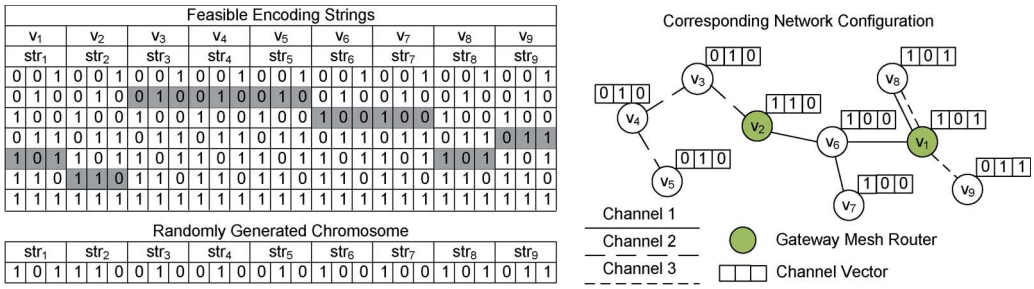

Fig. 5. Encoding process, random generation of a feasible chromosome, and the corresponding radio and channel configuration in a type-II WMN, with $K = 3$.

that were used in our LP model for all mesh routers. Taking node v_1 in Fig. 3 for example, $str_1 = (1, 3)$ translates into channel vector $\vec{c}_1 = (1, 0, 1)$, indicating two radios that operate on channels 1 and 3, respectively. In this manner, the obtained best chromosome can be turned into optimized channel vectors $(\vec{c}_1, \vec{c}_2, \ldots, \vec{c}_{|V|})$ for all type-I mesh routers. These channel vectors are then fed back into our LP model to compute a high-throughput MCR schedule.

V. Wireless Mesh Networks With the Total Radio Number Constraint (Type II)

In a type-II WMN, we relax the per-node radio number constraint but impose an upper bound on the sum of radio interfaces that were used by all mesh routers, referred to as the *total radio number constraint*. This is generally the case when all mesh routers have equally powerful capabilities and are possibly provided by the same service organization for WMN deployment with restricted radio resource. Recall that K nonoverlapping (orthogonal) channels are available in the system. We define parameter N to denote the maximum number of total radio interfaces that are allowed by the network provider, where $|V| \leq N \leq K \cdot |V|$. Without the per-node radio number restriction, the number of radios at the respective mesh router can be adjusted according to their relaying traffic demands. Consequently, type-II WMNs are a more flexible type of deployment model and expected to produce better capacity performance than type-I WMNs. Parameter N is a given argument in our type-II GA computations.

A. Binary-Encoding Process

Fig. 5 shows a sample type-II WMN that contains nine mesh nodes $(v_1, v_2, \ldots, v_9)$. Define substring str_i as the encoding of some feasible radio and channel configuration at node v_i. Because the per-node radio number constraint n_i no longer exists, we let the encoding be represented by a *bit string* with fixed length K. In particular, substring str_i is a string of values in binary form that contains K b. For example, bit string $str_2 = (1, 1, 0)$ is used to indicate that mesh node v_2 is equipped with two radios that operate on channels 1 and 2, respectively. To achieve feasible substring encodings for any type-II mesh router v_i, the following general rule needs to be followed.

- *At least* one radio that binds to some valid channel should exist for a mesh node. This condition implies that at least one among K b in a substring should be set to *true* (value of 1). In other words, $str_i = (0, 0, 0)$, with all three digits (bits) set to 0, is an *infeasible* encoding for node v_i.

Consequently, for any mesh node v_i, the total number of feasible radio and channel encodings can be derived by computing $2^K - 1$ (excluding the one with K 0s). Because all substrings have the same length of K b, for any mesh node in Fig. 5, there are $2^K - 1 = 2^3 - 1 = 7$ feasible combinations for substring encodings, including (0, 0, 1), (0, 1, 0), (1, 0, 0), (0, 1, 1), (1, 0, 1), (1, 1, 0), and (1, 1, 1). We may notice that the substring here directly translates to the channel vector that was used in our LP model. Once all feasible substring encodings are available, we randomly combine these substrings to form a chromosome with length $L = K \cdot |V|$, as shown on the lower left side of Fig. 5.

B. Initial Population Formation

Because the total radios used by all type-II mesh routers cannot exceed the given parameter N, a randomly constructed chromosome is not necessarily a feasible radio configuration. Therefore, for any constructed chromosome q, the type-II GA first checks if the total number of 1s contained in q exceeds N. If yes, chromosome q is discarded, and another chromosome is constructed for consideration. Otherwise, the type-II GA obtains the fitness value f based on LP calculations. If f is solvable with a positive returned value, then the chromosome q is selected into the initial population pool $P(0)$. Otherwise, the current chromosome q is dropped, and another chromosome is constructed for selection consideration. This process continues until either Q chromosomes have successfully been selected or the maximum M chromosomes have been tested, where $M \geq Q$. When the process terminates, if the size of $P(0)$ is zero (none of the constructed M chromosomes are feasible), then the type-II GA declares No_Solution. Otherwise, if the size of $P(0)$ is nonzero but smaller than Q (less than Q feasible chromosomes have been selected), then the type-II GA simply keeps duplicating the latest discovered feasible chromosome until $P(0)$ contains *exactly* Q chromosomes. The formed initial population $P(0)$ then acts as the *very first generation* that kicks off the subsequent evolving steps, as shown in Fig. 2.

C. Crossover and Mutation Operations

Crossover (recombination) and mutation are two key operations that are involved in the population-evolving process. At the crossover stage, the purpose is to generate offspring chromosomes (new sample points) by performing certain

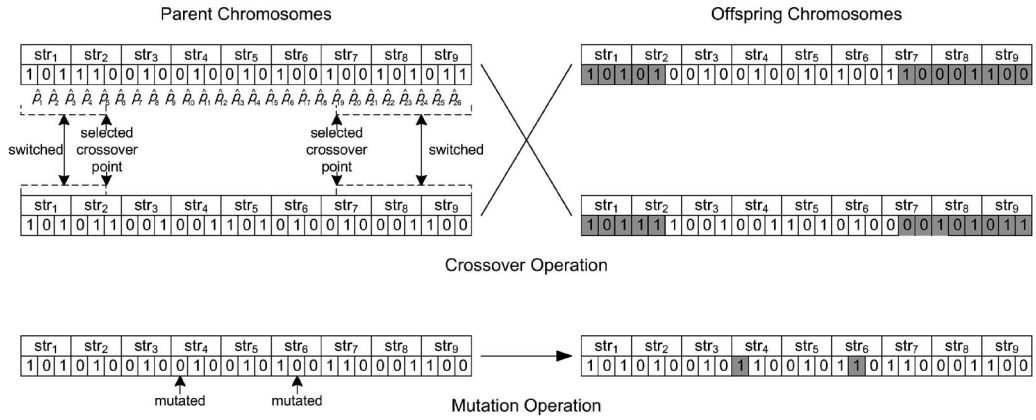

Fig. 6. Type-II crossover operation on a pair of parent chromosomes and type-II mutation operation on a given chromosome (with the shaded parts being the modified genes).

genetic shuffling on parent chromosomes. These offspring chromosomes share genetic similarities with their parents but form a different generation. In our GA model, we define a probability p_c and select $\lceil (Q \cdot p_c)/2 \rceil$ pairs of parent chromosomes from the current population $P(t)$ that is output from the roulette wheel selection procedure. All chromosomes in $P(t)$ are randomly selected with equal probability and cannot repeatedly be selected. For any selected pair of parent chromosomes, we adopt the *two-point crossover* operation, as illustrated on the upper part of Fig. 6. Define crossover points to occur between any two adjacent bits in a chromosome, denoted as $\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{L-1}$, where $L = K \cdot |V|$. The two-point crossover operator randomly generates two distinct cutoff points $\hat{p}_a$ and $\hat{p}_b$, where $1 \leq a < b \leq L-1$ and switches the bits before $\hat{p}_a$ (here, $a = 5$) and after $\hat{p}_b$ (here, $b = 19$) of the parent chromosomes to form offspring chromosomes. The created offspring chromosomes then replace their parents in population $P(t)$. Chromosomes that were not selected for the crossover operation remain in $P(t)$.

Next, the mutation operation is individually applied to each chromosome in $P(t)$. The mutation exercise attempts to bring solutions out of local optimal points of the search space. In a type-II WMN, we view each bit in a chromosome as a single gene with binary values (0 or 1). Define a small probability p_m as the likelihood of a gene to mutate. The type-II GA mutation operator randomly alters each bit (gene) with a low probability p_m for all chromosomes in $P(t)$. If a bit (gene) is determined to mutate, it flips between values of 1 and 0. As illustrated on the lower part of Fig. 6, the first bit of str_4 and the second bit of str_6 mutate to 1 from 0. When the mutation operation completes, a new population $P(t+1)$ is now formed.

For any chromosome q_i in $P(t+1)$, the type-II GA examines if the total number of 1s contained in q_i exceeds parameter N. If yes, GA sets the associated fitness value $f_i = 0$. Otherwise, GA computes fitness value f_i based on the LP model. In case LP returns No_Solution for q_i, GA sets $f_i = 0$. Chromosomes with fitness values set to 0 will be filtered out by the roulette wheel selection procedure in the next iteration. When the fitness updating completes, population $P(t+1)$ proceeds, and the evolving process starts over again. Algorithm 3 provides the detailed pseudocode on our type-II crossover and mutation operations.

Algorithm 3: Type-II crossover and mutation operations for a given population $P(t)$.

```
// Crossover operation
Define crossover points p̂_1, p̂_2, ..., p̂_{L-1} between any two adjacent bits in a chromosome;
Randomly select ⌈(Q · p_c)/2⌉ pairs of parent chromosomes from P(t);
for (each pair of parent chromosomes) do
    Randomly generate two crossover points p̂_a and p̂_b, with 1 ≤ a < b ≤ L − 1;
    Switch the digits (bits) before p̂_a and after p̂_b of the parent chromosomes to form two offspring chromosomes;
    Replace the parent chromosomes with their offspring chromosomes in P(t);
// Mutation operation
for (i = 1; i ≤ Q; i++) do
    for (j = 1; j ≤ L; j++) do
        Randomly generate a number x from (0, 1);
        if (x ≤ p_m) then
            if (bit b_j in chromosome q_i == 0) then
                b_j = 1;
            else
                b_j = 0;
// Now, a new population P(t + 1) is formed and requires postprocessing before entering the next iteration
for (i = 1; i ≤ Q; i++) do
    if (# of 1s in chromosome q_i > N) then
        Let the fitness value f_i for chromosome q_i in P(t + 1) = 0;
    else
        Compute the fitness value f_i for chromosome q_i in P(t + 1) based on the LP model;
        if (LP returns No_Solution) then
            f_i = 0;
```

The obtained best chromosome (with the highest fitness value) represents an optimized radio and channel configuration, and the mapping of a chromosome onto corresponding network setting can be found on the right side of Fig. 5. In a type-II

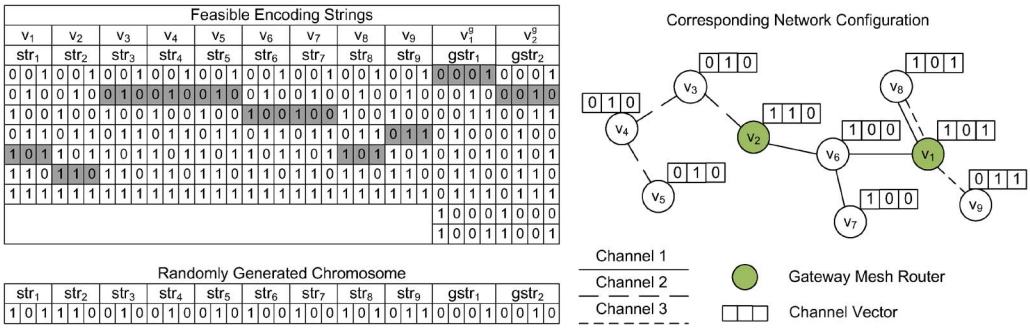

Fig. 7. Encoding process, random generation of a feasible chromosome, and the corresponding radio and channel configuration in a type-III WMN, with $K = 3$.

WMN, a chromosome directly translates into corresponding channel vectors used in our LP model. Consequently, the obtained best chromosome represents optimized channel vectors for respective type-II mesh routers. The channel vectors are then fed back into our LP model for computing high-throughput routing traffic distributions.

VI. Wireless Mesh Networks With the Total Radio Constraint and Unknown Gateway Location (Type III)

A type-III WMN inherits the total radio number constraint from its type-II counterpart while further relaxing the predefined gateway location constraint. This is the most flexible but complex deployment model among the three types of WMNs. Except for configuring radio and channel settings for all mesh routers, we attempt to simultaneously optimize the gateway placement and demonstrate that GA can perform complex optimization that is involved in multiple discrete variables. By placing gateways at appropriate locations, a type-III WMN has a better chance of balancing network traffic and, thus, produce higher system throughput than the previous two types. Recalling the set of gateway nodes V^g in our LP model, we denote the number of available gateways as $|V^g|$ (cardinality of set V^g). Parameter N (total radio number constraint) from the type-II deployment model and the number of supported gateways $|V^g|$ are both given arguments in our type-III GA computations.

A. Binary-Encoding Process

Fig. 7 depicts a type-III WMN with nine mesh routers $(v_1, v_2, \ldots, v_9)$ and two supported gateways with undetermined locations. The encoding process for radio and channel configuration is the same as the process that was adopted in a type-II WMN (see Section V-A). In addition, in a type-III WMN, we define *gateway substrings*, denoted as $gstr_1, gstr_2, \ldots, gstr_{|V^g|}$, to indicate the IDs of mesh routers that act as the gateways in binary form. With a total of $|V|$ mesh routers in the network, we encode a gateway substring using y b, where $y = \lceil log_2|V| \rceil$. Because all mesh routers are gateway candidates, each gateway substring can take on $|V|$ different values. To indicate a mesh node v_i that acts as a gateway, the type-III GA sets the (decimal) value of the corresponding gateway substring to j, where $j = i \ mod \ 2^y$. Although a total of 2^y binary combinations are possible for a gateway substring, some values are considered invalid when we do not have that many mesh routers (i.e., $|V| < 2^y$) for placing gateways. In particular, a feasible gateway substring encoding should satisfy the following rule.

- The (decimal) value j of a gateway substring must indicate a location with a valid mesh router ID. In other words, the inequality $0 \leq (j-1) \ mod \ 2^y \leq |V| - 1$ should hold for any valid gateway substring with value j.

For example, in Fig. 7, because there are nine mesh routers, feasible gateway substring encodings include (0, 0, 0, 1), (0, 0, 1, 0), (0, 0, 1, 1), (0, 1, 0, 0), (0, 1, 0, 1), (0, 1, 1, 0), (0, 1, 1, 1), (1, 0, 0, 0), and (1, 0, 0, 1), whereas other substring encodings, such as (1, 1, 0, 0) and (1, 1, 1, 1), are *infeasible* encodings (invalid gateway substrings). The type-III GA appends the gateway substrings at the end of a chromosome. Once all feasible channel and gateway substring encodings are available, we randomly combine the substrings to form a chromosome with length $L = K \cdot |V| + y \cdot |V^g|$. As shown on the lower left of Fig. 7, the randomly generated chromosome contains $gstr_1 = (0, 0, 0, 1)$ and $gstr_2 = (0, 0, 1, 0)$, indicating that gateways need to be placed at mesh routers v_1 and v_2.

B. Initial Population Formation

For any constructed chromosome q, the type-III GA examines if either of the following conditions occurs.

- The total number of 1s that were contained in q, excluding gateway substrings, is more than N (violating the total radio number constraint).
- There exist duplicate gateway substrings that point to the same location (mesh router ID) among $gstr_1, gstr_2, \ldots, gstr_{|V^g|}$ in chromosome q.

If yes, the current chromosome q is discarded, and another chromosome is constructed for consideration. Otherwise, GA obtains the fitness value f based on LP calculations. In case f is solvable with a positive returned value, the chromosome q is selected into the initial population pool $P(0)$. Otherwise, the current chromosome q is dropped, and another chromosome is constructed for selection consideration. This process continues until either Q chromosomes have successfully been selected or the maximum M chromosomes have been tested, where $M \geq Q$. When the process terminates, if the size of $P(0)$ is zero (none of the constructed M chromosomes are feasible), then the type-III GA declares No_Solution. Otherwise, if the size of $P(0)$ is nonzero but smaller than Q (less than Q

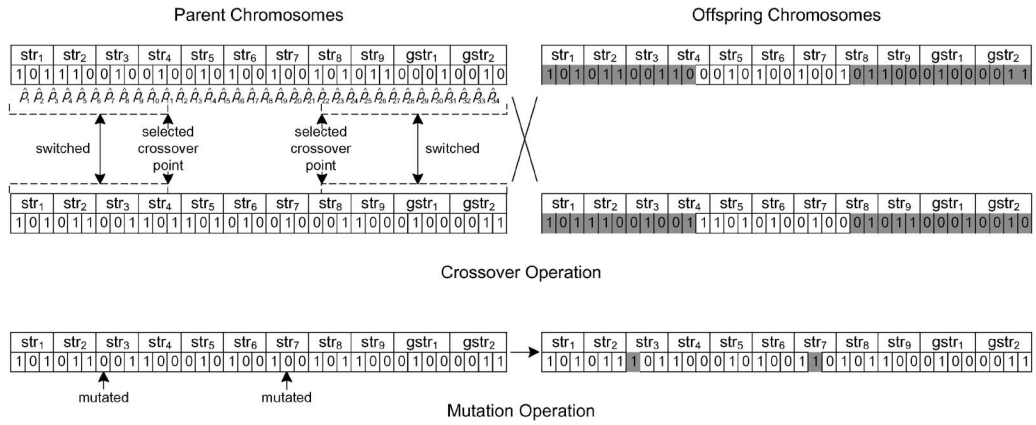

Fig. 8. Type-III crossover operation on a pair of parent chromosomes and type-III mutation operation on a given chromosome (with the shaded parts being the modified genes).

feasible chromosomes have been selected), then the type-III GA keeps duplicating the latest discovered feasible chromosome until $P(0)$ contains *exactly* Q chromosomes. The formed initial population $P(0)$ then acts as the *very first generation* that kicks off the subsequent evolving steps, as shown in Fig. 2.

C. Crossover and Mutation Operations

At the crossover phase, certain genetic recombination on parent chromosomes is performed to generate offspring chromosomes. These offspring chromosomes share genetic similarities with their parents but form a different generation. In our GA model, we define a probability p_c and select $\lceil (Q \cdot p_c)/2 \rceil$ pairs of parent chromosomes from the current population $P(t)$. All chromosomes in $P(t)$ are randomly selected with equal probability and cannot repeatedly be selected. For any selected pair of parent chromosomes, we adopt the *two-point crossover* operation, as illustrated on the upper point of Fig. 8. Define crossover points to occur between any two adjacent bits in a chromosome, denoted as $\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{L-1}$, where $L = K \cdot |V| + y \cdot |V^g|$. The two-point crossover operator randomly generates two distinct cutoff points $\hat{p}_a$ and $\hat{p}_b$, where $1 \leq a < b \leq L-1$ and switches the bits before $\hat{p}_a$ (here, $a = 11$) and after $\hat{p}_b$ (here, $b = 22$) of the parent chromosomes to form offspring chromosomes. The created offspring chromosomes then replace their parents in population $P(t)$. Chromosomes that were not selected for the crossover operation remain in $P(t)$.

Then, the type-II GA mutation operator randomly alters each bit (gene) with a low probability p_m for all chromosomes in $P(t)$. If a bit (gene) is determined to mutate, it flips between values of 1 and 0. As illustrated on the lower part of Fig. 8, the first bit of str_3 and the second bit of str_7 mutate to 1 from 0. When the mutation operation completes, a new population $P(t+1)$ is now formed.

For any chromosome q_i in $P(t+1)$, the type-III GA inspects if either of the following situations occurs.

- The total number of 1s that are contained in q_i' surpasses parameter N, where $q_i' = q_i$, excluding gateway substrings $gstr_1, gstr_2, \ldots, gstr_{|V^g|}$.
- There exist one or more gateway substrings, among $gstr_1, gstr_2, \ldots, gstr_{|V^g|}$, that point to *infeasible* locations (invalid mesh router IDs) as a result of performing crossover and mutation operations.

If yes, GA sets the associated fitness value $f_i = 0$. Otherwise, GA computes the fitness value f_i based on the LP model. In case LP returns No_Solution for q_i, GA sets $f_i = 0$. Chromosomes with fitness values set to 0 will be filtered out by the roulette wheel selection procedure in the next iteration. When the fitness updating completes, population $P(t+1)$ proceeds, and the evolving process starts over again. Algorithm 4 provides the detailed pseudocode on our type-III crossover and mutation operations.

Algorithm 4: Type-III crossover and mutation operations for a given population $P(t)$.

1: // Crossover operation
2: Define crossover points $\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{L-1}$ between any two adjacent bits in a chromosome;
3: Randomly select $\lceil (Q \cdot p_c)/2 \rceil$ pairs of parent chromosomes from $P(t)$;
4: **for** (each pair of parent chromosomes) **do**
5: Randomly generate two crossover points $\hat{p}_a$ and $\hat{p}_b$, with $1 \leq a < b \leq L-1$;
6: Switch the digits (bits) before $\hat{p}_a$ and after $\hat{p}_b$ of the parent chromosomes to form two offspring chromosomes;
7: Replace the parent chromosomes with their offspring chromosomes in $P(t)$;
8: // Mutation operation
9: **for** $(i = 1; i \leq Q; i++)$ **do**
10: **for** $(j = 1; j \leq L; j++)$ **do**
11: Randomly generate a number x from (0, 1);
12: **if** $(x \leq p_m)$ **then**
13: **if** (bit b_j in chromosome $q_i == 0$) **then**
14: $b_j = 1$;
15: **else**
16: $b_j = 0$;
17: // Now, a new population $P(t+1)$ is formed and requires postprocessing before entering the next iteration
18: **for** $(i = 1; i \leq Q; i++)$ **do**
19: Define $q_i' = q_i$ excluding gateway substrings;

			GA I / Hyacinth / PSO Parameter Settings																			GA II / DIM Settings			GA III Settings		
Scenario		K	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}	n_{17}	n_{18}	Gateway ID	K	N	Gateway ID	K	N	# Gateway(s)
			n_{19}	n_{20}	n_{21}	n_{22}	n_{23}	n_{24}	n_{25}	n_{26}	n_{27}	n_{28}	n_{29}	n_{30}	n_{31}	n_{32}	n_{33}	n_{34}	n_{35}	n_{36}							
1	2 × 2	4	2	3	1	2															v_2	4	8	v_2	4	8	1
2	2 × 2	4	3	2	2	1															v_1, v_4	4	8	v_1, v_4	4	8	2
3	3 × 3	4	4	2	2	3	2	1	1	2	1										v_1	4	18	v_1	4	18	1
4	3 × 3	4	4	2	1	2	2	1	3	2	1										v_1, v_7	4	18	v_1, v_7	4	18	2
5	4 × 4	5	4	2	1	1	3	2	1	2	2	1	2	2	1	1	3	4			v_1, v_{16}	5	32	v_1, v_{16}	5	32	2
6	4 × 4	5	3	2	2	3	1	1	1	2	2	2	2	1	4	3	2	1			v_1, v_4, v_{13}	5	32	v_1, v_4, v_{13}	5	32	3
7	5 × 5	5	1	2	2	4	1	1	1	2	3	1	2	1	1	2	1	4	3	2	v_4, v_{21}	5	50	v_4, v_{21}	5	50	2
			1	1	5	3	3	2	1																		
8	5 × 5	5	1	1	2	2	3	2	2	3	1	2	2	2	4	2	2	2	1	3	v_5, v_{13}, v_{21}	5	50	v_5, v_{13}, v_{21}	5	50	3
			2	2	3	2	2	1	1																		
9	6 × 6	5	1	2	1	1	2	1	1	2	3	2	2	1	2	3	5	4	2	2	v_{15}, v_{36}	5	72	v_{15}, v_{36}	5	72	2
			1	2	4	3	2	1	2	2	2	2	2	2	1	2	1	1	2	3							
10	6 × 6	5	5	4	2	1	1	1	3	2	1	1	2	1	2	1	2	2	2	2	v_1, v_{21}, v_{36}	5	72	v_1, v_{21}, v_{36}	5	72	3
			1	2	5	4	2	1	1	2	2	2	2	3	1	1	1	1	2	4							

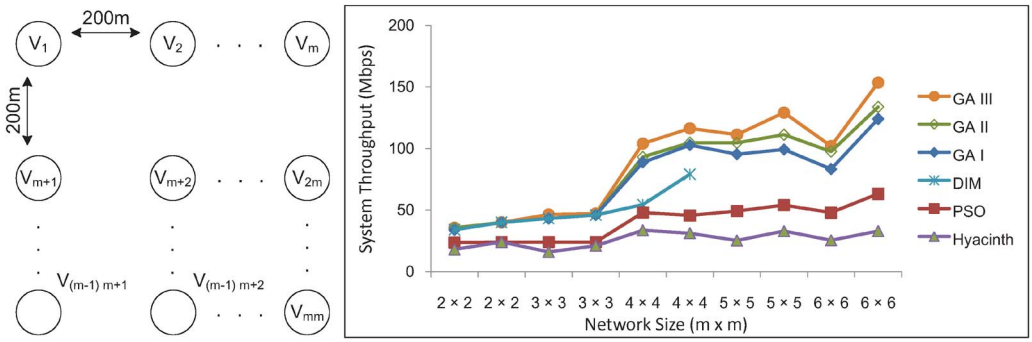

Fig. 9. Obtainable system throughput produced by the respective WMN optimization approaches under different network sizes.

20: **if** (# of 1s in chromosome $q_i' > N$) || (invalid gateway substring occurs) **then**
21: Let the fitness value f_i for chromosome q_i in $P(t+1) = 0$;
22: **else**
23: Compute the fitness value f_i for chromosome q_i in $P(t+1)$ based on the LP model;
24: **if** (LP returns No_Solution) **then**
25: $f_i = 0$;

Ultimately, we obtain the best chromosome (with the highest fitness value), and the mapping of a chromosome onto the corresponding network setting can be found on the right side of Fig. 7. In a type-III WMN, a chromosome directly translates into corresponding channel vectors and gateway location(s) used in our LP model. Consequently, the obtained best chromosome represents optimized channel vectors and suggested gateway(s) placement. The channel vectors and gateway location(s) are then fed back into our LP model to compute the high-throughput MCR schedule.

VII. Performance Evaluation

In this section, we conduct experiments using the ns-2 simulator (version 2.29) with multiradio extension in an IEEE 802.11a networking environment. The two-ray ground interference model and default transmit power are used, leading to around a 250-m transmission distance and a 440-m interference range. The distributed coordination function request-to-send/clear-to-send four-way handshaking is turned on. We denote our three GA-based optimization mechanisms for type-I, type-II, and type-III WMNs as GA I, GA II, and GA III, respectively. We set the population size $Q = 20$, maximum number of tested chromosomes $M = 100$ in forming the initial population $P(0)$, maximum allowable number of evolving iterations $T = 300$, crossover probability $p_c = 0.9$, and mutation probability $p_m = 0.02$. For comparison, we also implement the following three other WMN planning approaches: 1) decremental interface management (DIM) [16]; 2) particle swarm optimization (PSO) [23]; and 3) Hyacinth [22]. DIM performs the optimization based on LP computations under the same total radio number constraint (parameter N) as our GA II, whereas PSO and Hyacinth execute optimization under the same per-node radio number constraint as our GA I. For type-III WMNs, to the best of our knowledge, GA III is the first work to simultaneously optimize radio configuration and gateway placement under the constraints of totally used radios and supported gateways, and thus, no adequate target is available for comparison. In the following paragraphs, we separately describe the designs of three comparative approaches.

The idea of DIM is to initially equip each mesh router with K radios and gradually remove unnecessary or least useful radio interfaces until the total radio number constraint is met. In the first run of LP calculations, several wireless links may be discovered to bear zero routing traffic, and thus, corresponding radios can be removed. In the following runs, when all links

bear nonzero traffic, DIM evaluates each radio and removes the least useful radio interface one by one until the total number of used radios becomes no greater than N [16]. In the LP model, we assume that gateways do *not* generate traffic. For other mesh hosts that generate traffic, we set the user traffic lower bound to $L = 0.2$ Mb/s and the user traffic upper bound to $U = 10$ Mb/s for both uplink and downlink flows. In addition, symmetric gateways with a capacity of $B = 100$ Mb/s and symmetric wireless links with a data rate of $R = 12$ Mb/s are used in the simulations.

PSO encodes a possible radio and channel configuration in a particle (analogous to a chromosome in GAs). Rather than directly dealing with mesh nodes, PSO utilizes a conflict graph that transforms a wireless link into a corresponding node. In the conflict graph, links that interfere are interconnected by edges. Suppose that a total of $|E|$ wireless links exist in the original graph. PSO randomly chooses a channel (from 1 to K), e.g., k, to construct a particle, with all $|E|$ digits set to value k, implying all links using channel k. The initial population $P(0)$ then contains Q randomly constructed particles as the first generation to start the subsequent evolving iterations. A fitness value that is associated with a particle is defined as the *inverse* of interference level that is produced by the corresponding radio configuration. PSO quantifies the interference level as the total number of interfered links in the conflict graph. In the end of each iteration, PSO globally updates the best particle $gBest$ and locally updates the best particle $pBest$ in the current population $P(t)$. When entering a new iteration, PSO performs a crossover operation on every particle by exchanging one (randomly chosen) digit with $gBest$ and another (randomly chosen) digit with $pBest$. In case the particle after the crossover operation becomes invalid (violating the per-node radio number constraint), the particle is restored to the original setting. No mutation operation is carried out in PSO [23]. In the simulations, we set the interference range at two hops around each link for constructing the conflict graph in PSO, the population size $Q = 20$, and the maximum allowable number of iterations $T = 300$ as we do in GA-based approaches. After obtaining the best radio configuration, PSO performs LP computations for the routing schedule.

In Hyacinth, multiple spanning trees that were rooted at respective gateway mesh routers are constructed and called the Hyacinth architecture. Based on the tree structures, each mesh router has exactly one parent node through *up* radio interface and, possibly, multiple child nodes through *down* radio interfaces. To avoid the ripple effect, which is produced by changing channels over links, each mesh node in Hyacinth follows the channel set by its parent over the *up* interface and can only change the channels used by its children over *down* interfaces. When a node joins the network, it favors the tree with the shortest path (minimum hops) to the gateway node and sets the channel following its parent [22]. Suppose that α is the ratio of interference range to the transmission distance. A mesh node in Hyacinth selects the least used channel (with the most residual channel capacity) within α hops for the *down* interface to communicate with a child node on the premise of satisfying the per-node radio number constraint. If the constraint is violated, only the channels that were used by existing radios can be selected. In the simulations, we assume that the global network topology and residual link capacities are obtainable by Hyacinth.

We set up $m \times m$ grid networks of different sizes (from 2×2 up to 6×6, with two scenarios generated for each network size), in which each mesh node is separated by an equal distance of 200 m.[7] The numbering rule for all mesh routers is displayed in Fig. 9, lower left. Ten scenarios with corresponding parameter settings, as shown in Fig. 9, are simulated. Except for GA III, gateway positions are pregiven. To saturate the network, we generate constant-bit-rate traffic, with the sending rate set to 24 Mb/s. The aggregate system throughput produced by different mechanisms under the ten scenarios is included on the lower right of Fig. 9. Our GA-based approaches perform better than other approaches by two to three times on the average. Because tree-based architecture leads to single-path routing without effectively exploiting multiple channels, Hyacinth yields the lowest throughput under most scenarios. For PSO, the performance is slightly better than Hyacinth due to the multipath data delivery enabled by LP-based traffic distributions. However, because PSO is independent of routing traffic demands and equally treats each mesh routers, the throughput improvement is quite limited. The DIM mechanism addresses this drawback by allocating more bandwidth resource to links with more relaying traffic, avoiding those links (closer to gateways) to become the traffic bottleneck, hence achieving better performance than PSO and Hyacinth. However, by performing radio removal one by one, DIM often reaches a local optimum in the solutions space, because the removed radio cannot be plugged back, leading to less throughput achieved compared with GA-based strategies. Moreover, as shown in Fig. 9, DIM reaches its computational limit when the network size grows to 4×4 and cannot produce meaningful solutions beyond that.

Focusing on a 4×4 grid network (the maximum network size that DIM can support), we simulate another nine scenarios, with the corresponding parameter settings summarized in Fig. 10. In scenarios 1–3, one gateway node is available, whereas in scenarios 4–6 and 7–9, two and three gateways are supported, respectively. The system throughput results, as shown in Fig. 10, lower right, provide similar performance insights to the results that were implied in Fig. 9. To further investigate the design implications, Fig. 11 displays the resulting radio and channel configurations that were computed by respective strategies under scenarios 3, 5, and 8 in a 4×4 grid network. Due to the gateway placement flexibility, GA III always performs best, whereas PSO tends to use the least radios, because only one channel can be assigned to each wireless link. When the evolving process in PSO fails to produce better particles that satisfy the per-node radio number constraint, the solution will be stuck in a local point, resulting in an undesirable radio configuration and even yielding less throughput than

[7]In this paper, we target on moderate-size community WMN planning. For a regular community WMN (that contains tens of mesh routers), our GA-based mechanisms work properly. For very large scale WMNs (that consist of hundreds of mesh routers or more), directly applying GA-based algorithms on the whole network is possible but not efficient. We will explore the large-scale WMN planning scalability issues and report possible findings in our future work.

GA I / Hyacinth / PSO Parameter Settings

Scenario	K	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	n_{12}	n_{13}	n_{14}	n_{15}	n_{16}	Gateway ID
1	4	2	1	2	2	1	4	2	2	2	2	2	2	2	2	2	2	v_6
2	4	2	1	1	1	2	4	1	1	2	2	1	1	1	1	1	1	v_6
3	5	5	2	1	1	2	1	2	2	2	2	2	1	2	1	1	1	v_1
4	5	3	1	2	2	2	2	2	2	2	1	4	2	2	2	2	2	v_1, v_{11}
5	4	4	2	1	1	2	1	2	1	2	1	1	2	1	1	2	4	v_1, v_{16}
6	4	1	4	1	1	1	2	2	1	1	2	2	1	1	1	4	1	v_2, v_{15}
7	5	3	1	2	2	2	2	2	2	1	1	4	2	2	1	2	2	v_1, v_{11}, v_{13}
8	5	4	1	1	4	2	1	1	2	2	1	1	1	4	2	1	1	v_1, v_4, v_{13}
9	5	1	1	2	3	1	1	1	2	2	3	2	1	3	1	1	1	v_4, v_{10}, v_{13}

GA II / DIM Settings

K	N	Gateway ID
4	32	v_6
4	23	v_6
5	28	v_1
5	33	v_1, v_{11}
4	28	v_1, v_{16}
4	26	v_2, v_{15}
5	31	v_1, v_{11}, v_{13}
5	29	v_1, v_4, v_{13}
5	26	v_4, v_{10}, v_{13}

GA III Settings

K	N	# Gateway(s)
4	32	1
4	23	1
5	28	1
5	33	2
4	28	2
4	26	2
5	31	3
5	29	3
5	26	3

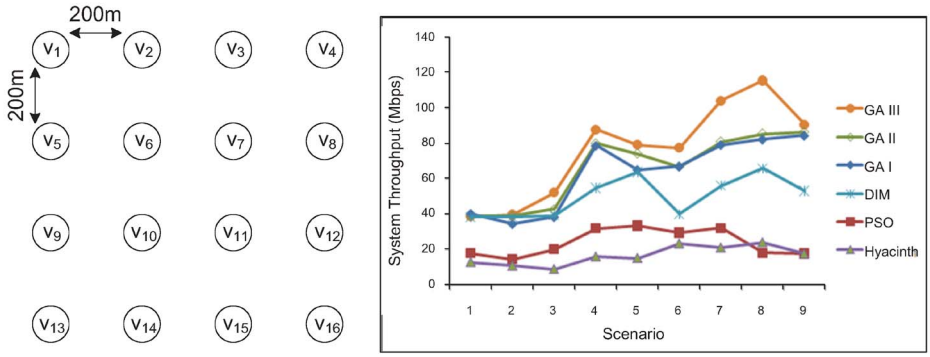

Fig. 10. Nine simulated scenarios with the corresponding parameter settings under respective approaches for planning a 4×4 mesh network and the resulting aggregate system throughput versus different scenarios under the respective WMN optimization approaches.

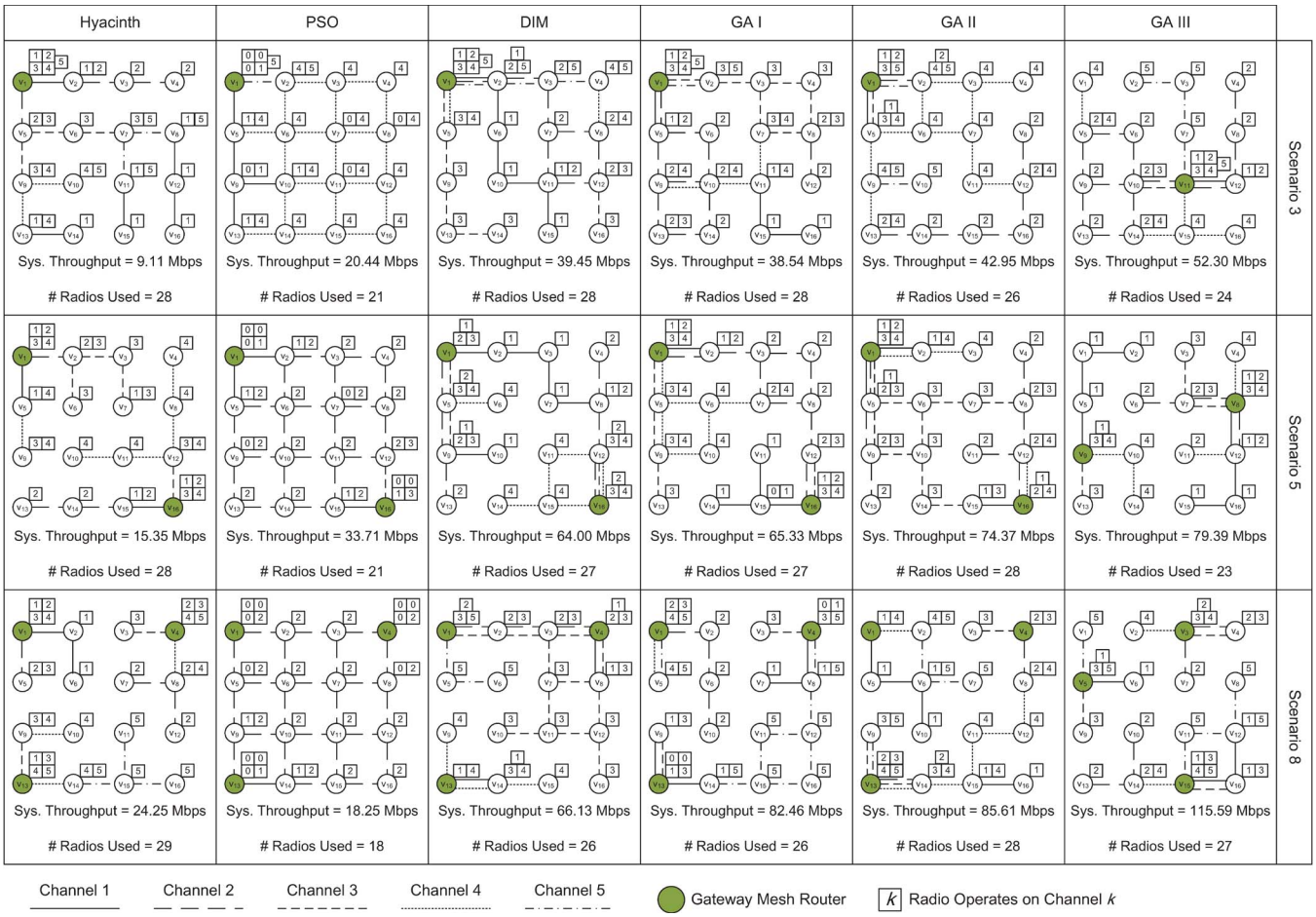

Fig. 11. Display of the resulting radio and channel configurations produced by the respective WMN optimization approaches when one gateway (scenario 3), two gateways (scenario 5), and three gateways (scenario 8) are available among mesh routers.

Hyacinth (scenario 8). Among the three GA-based approaches, GA III tends to generate the most network capacity with fewer required total radios. Based on the aforementioned results, we demonstrate that a load-aware WMN optimization should judiciously distribute radio and channel resources among wireless links in a way that matches their traffic demands.

VIII. Conclusion

In this paper, we have focused on establishing a high-capacity MCMR WMN. Two major problems that pertain to the performance of MCMR WMNs are CA and MCR. We proposed GA-based techniques for CA and LP formulations to obtain the MCR schedule. The interplay of CA and MCR was modeled by connecting the fitness value of a chromosome with the value of linear objective function. At the WMN deployment stage, our approach can compute an optimized CA configuration with the corresponding MCR schedule in polynomial time. Through detailed presentations of genetic evolutions under three different resource constraints, we have demonstrated that GA is quite promising for solving complex discrete models involved in MCMR WMNs. Simulation results showed that GA-based techniques deliver pretty good WMN planning solutions.

References

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *Proc. ACM SIGCOMM*, Oct. 2004, vol. 34, no. 4, pp. 121–131.

[2] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Comput. Netw.*, vol. 47, no. 4, pp. 445–487, Jan. 2005.

[3] M. Alicherry, R. Bhatia, and L. E. Li, "Joint channel assignment and routing for throughput optimization in multiradio wireless mesh networks," in *Proc. ACM Int. Conf. MobiCom*, Aug. 2005, pp. 58–72.

[4] L. Badia, A. Botta, and L. Lenzini, "A genetic approach to joint routing and link scheduling for wireless mesh networks," *Ad Hoc Netw.*, vol. 7, no. 4, pp. 654–664, Jun. 2009.

[5] H. Cheng, N. Xiong, G. Chen, and X. Zhuang, "Channel assignment with topology preservation for multiradio wireless mesh networks," *J. Commun.*, vol. 5, no. 1, pp. 63–70, Jan. 2010.

[6] E. K. P. Chong and S. H. Zak, *An Introduction to Optimization*, 2nd ed. New York: Wiley, Apr. 2004.

[7] K. C. Claffy, H.-W. Braun, and G. C. Polyzos, "A parameterizable methodology for Internet traffic flow profiling," *IEEE J. Sel. Areas Commun.*, vol. 13, no. 8, pp. 1481–1494, Oct. 1995.

[8] J. Crichigno, M.-Y. Wu, and W. Shu, "Protocols and architectures for channel assignment in wireless mesh networks," *Ad Hoc Netw.*, vol. 6, no. 7, pp. 1051–1077, Sep. 2008.

[9] L. D. Davis, *Handbook of Genetic Algorithms*, 1st ed. New York: Van Nostrand, Jan. 1991.

[10] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no. 2, pp. 388–404, Mar. 2000.

[11] P.-H. Hsiao, A. Hwang, H.-T. Kung, and D. Vlah, "Load-balancing routing for wireless access networks," in *Proc. IEEE INFOCOM*, Apr. 2001, vol. 2, pp. 986–995.

[12] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multihop wireless network performance," *Wireless Netw.*, vol. 11, no. 4, pp. 471–487, Jul. 2005.

[13] K.-T. Ko, K.-S. Tang, C.-Y. Chan, and S. Kwong, "Using genetic algorithms to design mesh networks," *Computer*, vol. 30, no. 8, pp. 56–61, Aug. 1997.

[14] S. Lakshmanan, R. Sivakumar, and K. Sundaresan, "Multigateway association in wireless mesh networks," *Ad Hoc Netw.*, vol. 7, no. 3, pp. 622–637, May 2009.

[15] X.-Y. Li, A. Nusairat, Y. Wu, Y. Qi, J. Zhao, X. Chu, and Y. Liu, "Joint throughput optimization for wireless mesh networks," *IEEE Trans. Mobile Comput.*, vol. 8, no. 7, pp. 895–909, Jul. 2009.

[16] T.-Y. Lin, W.-H. Tam, K.-L. Fan, and Y.-C. Tseng, "Resource planning and packet forwarding in multiradio, multimode, multichannel, multirate (M4) wireless mesh networks," *Comput. Commun.*, vol. 31, no. 7, pp. 1329–1342, Feb. 2008.

[17] Y.-J. Lin and M. C. Chan, "A scalable monitoring approach based on aggregation and refinement," *IEEE J. Sel. Areas Commun.*, vol. 20, no. 4, pp. 677–690, May 2002.

[18] K. F. Man, K. S. Tang, and S. Kwong, "Genetic algorithms: Concepts and applications," *IEEE Trans. Ind. Electron.*, vol. 43, no. 5, pp. 519–534, Oct. 1996.

[19] M. K. Marina, S. R. Das, and A. P. Subramanian, "A topology control approach for utilizing multiple channels in multiradio wireless mesh networks," *Comput. Netw.*, vol. 54, no. 2, pp. 241–256, Feb. 2010.

[20] A. Ouni, H. Rivano, and F. Valois, "Capacity of wireless mesh networks: Determining elements and insensible properties," in *Proc. IEEE WCNC*, Apr. 2010, pp. 1–6.

[21] R. Pries, D. Staehle, M. Stoykova, B. Staehle, and P. Tran-Gia, "A genetic approach for wireless mesh network planning and optimization," in *Proc. ACM IWCMC*, Jun. 2009, pp. 1422–1427.

[22] A. Raniwala and T.-C. Chiueh, "Architecture and algorithms for an IEEE 802.11-based multichannel wireless mesh network," in *Proc. IEEE INFOCOM*, Mar. 2005, vol. 3, pp. 2223–2234.

[23] X. Zhuang, H. Cheng, and N. Xiong, "Channel assignment in multiradio wireless networks based on PSO algorithm," in *Proc. IEEE Int. Conf. FutureTech*, May 2010, pp. 1–6.

Ting-Yu Lin received the Ph.D. degree in computer science and information engineering from the National Chiao Tung University, Hsinchu, Taiwan.

From June 2003 to February 2004, she was with Massachusetts Institute of Technology, Cambridge, as a Research Scientist. From March 2004 to August 2005, she was with the Industrial Technology Research Institute of Taiwan as a Software Engineer. From September 2005 to August 2006, she was with the University of Illinois at Urbana-Champaign, as a Postdoctoral Research Associate, under the sponsorship of both the government and the university. She is currently an Assistant Professor with the Department of Electrical and Computer Engineering, National Chiao Tung University. Her research interests include wireless networking, mobile computing, and green communications.

Dr. Lin is a member of the Association for Computing Machinery. She is a recipient of the Phi Tau Phi Scholastic Honor award from the National Chiao Tung University.

Kai-Chiuan Hsieh received the M.S. degree in communications engineering from the National Chiao Tung University, Hsinchu, Taiwan, in October 2011.

He is currently a Software Engineer with ASUSTeK Computer Inc., Taipei, Taiwan. His research interests include wireless mesh networks and genetic algorithms.

Hsin-Chun Huang received the B.S. degree in communication engineering from Yuan-Ze University, Taoyuan, Taiwan, in June 2008. He is currently working toward the M.S. degree with the Department of Electrical and Computer Engineering, National Chiao Tung University, Hsinchu, Taiwan.

His research interests include wireless mesh networks and Linux-based system prototyping.