Optimal Resource Allocation for Increasing Strictly Concave Utility Functions in Wireless Networks

Tsern-Huei Lee, *Senior Member, IEEE*, Yu-Wen Huang, *Student Member, IEEE*, and Chien-Nan Chen

***Abstract*—Utility functions are widely used to model user-perceived service quality. For elastic traffic, the utility function is often concave. An elastic allocation (EA) algorithm has recently been proposed to maximize the total utility obtained by users, assuming resource is infinitesimally divisible and queues are constantly backlogged. We found that the EA algorithm is not always optimal. In fact, the solution it obtains can be infeasible. In this paper, we present a modified EA algorithm that is guaranteed to find the optimal solution under the same assumptions. The result is generalized for a system where queues are generally backlogged. In a real system, there is normally a basic unit for resources. Therefore, we further extend the designs to such a system for both constantly backlogged and generally backlogged queues. To reduce computational complexity, we also propose suboptimal resource allocation algorithms. Simulations are conducted to evaluate the proposed algorithms in terms of utility sum and execution time. Results show that our proposed algorithms perform better than previous works. Moreover, the performances of the proposed suboptimal algorithms are close to those of the optimal algorithms.**

***Index Terms*—Elastic traffic, resource allocation, utility, wireless network.**

I. Introduction

MAXIMIZING system throughput and achieving fairness among users in a wireless network are, in general, conflicting goals [1]. As a compromise, utility functions are widely used to model user satisfaction, and the goal becomes to maximize the total utility obtained by users.

It is reasonable to assume that utility function is increasing in the amount of allocated resources. In this paper, a function $f(x)$ is said to be increasing (or decreasing) if $b > a$ implies $f(b) > f(a)$ (respectively, $f(b) < f(a)$). In general, utility functions adopted for different types of applications are likely to be different. For example, in [2] and [3], the sigmoid and step functions are used, respectively, as the utility functions of soft and hard quality-of-service (QoS) traffic flows. For elastic traffic, the utility function is often an increasing strictly concave function [3]–[6].

In [4], a necessary and sufficient condition to maximize the utility function was derived to achieve cross-layer optimization for orthogonal frequency-division-multiplexing-based wireless networks. The same authors presented efficient dynamic subcarrier and adaptive power allocation algorithms in [5]. In [3], a similar necessary and sufficient condition was stated for a simpler system. Two types of traffic, namely, hard QoS and elastic traffic, were studied. Efficient algorithms were provided to find the near-optimal solution for hard QoS traffic and the optimal solution for elastic traffic. The elastic allocation (EA) algorithm presented in [3], which was designed to find the optimal solution for elastic traffic, is related to our work and will be reviewed in Section III. All the works presented in [3]–[5] assumed that queues are constantly backlogged and that resources are infinitesimally divisible.

In [6], the assumption of constantly backlogged queues is relaxed. In other words, queues are assumed to be generally backlogged such that the data buffered in a queue can be completely served if sufficient resource is allocated. The assumption of infinitesimally divisible resource remains intact. The resource allocation problem was formulated to maximize the total utility obtained by users subject to the constraints provided by Kleinrock's conservation law [7], which fully captures the behavior of a general queueing system. Since one crucial parameter of Kleinrock's conservation law is not easy to compute, the authors suggested to manage the resource by the famous packetized general processor sharing [8] or weighted fair queueing [9], with the weights determined by the Lagrangian multiplier method. In addition to the assumption of infinitesimally divisible resource, average data arrival rate and packet size of each user are required, which further limits the applicability of the work presented in [6]. For convenience, we call a resource allocation algorithm fluid flow based if it assumes that the resource is infinitesimally divisible.

In a real system, there is a basic unit for resource. Such a unit will be referred to as resource block in this paper. For example, in IEEE 802.16 WiMAX [10] and long-term evolution (LTE) [11], a resource block is normally constituted by several transmission symbols on one subchannel. The granularity can be selected to trade performance with complexity. To distinguish from fluid-flow-based schemes, an algorithm that allocates resource blocks is called a resource-block-based algorithm.

In [12], a resource-block-based algorithm called PF-MUX was proposed to achieve proportional fairness for orthogonal frequency-division multiple-access (OFDMA) systems. The concept of proportional fairness was introduced in [13], and according to [14], the utility function corresponding to proportional fairness is $\log(x)$, where x represents the average rate obtained by a user. It is not hard to see that $\log(x)$ is an increasing strictly concave function of x for all $x > 0$. The PF-MUX algorithm assumes queues are constantly backlogged and

Manuscript received June 14, 2011; revised September 27, 2011; accepted December 27, 2011. Date of publication January 23, 2011; date of current version March 21, 2012. The review of this paper was coordinated by Prof. J. Tang.

The authors are with the Institute of Communication Engineering, National Chiao Tung University, Hsinchu 30010, Taiwan (e-mail: tlee@banyan.cm.nctu.edu.tw; vicent@banyan.cm.nctu.edu.tw; brian28914@gmail.com).

Digital Object Identifier 10.1109/TVT.2012.2185719

performs resource allocation based on some heuristic. It will be briefly reviewed in Section III because we will compare its performance with those of our proposed resource-block-based algorithms.

The purpose of this paper is to present optimal fluid-flow-based and resource-block-based resource allocation algorithms for increasing strictly concave utility functions. The queues can be constantly backlogged or generally backlogged. For a fluid-flow-based system where queues are constantly backlogged, we found that the EA algorithm presented in [3] does not obtain the optimal solution for some cases. In fact, the solution it obtains can be infeasible. A modified EA (MEA) algorithm is proposed. We show that the MEA algorithm guarantees to find an optimal solution. We then generalize the results to a generally backlogged system. Given queue statuses, a necessary and sufficient condition for optimal resource allocation is developed. Moreover, an optimal resource allocation algorithm, called generalized EA (GEA), is designed based on the necessary and sufficient condition. Finally, we turn to consider resource-block-based systems, where queues are either constantly or generally backlogged. A necessary and sufficient condition for optimal resource allocation is stated for both constantly backlogged and generally backlogged systems. A straightforward resource allocation algorithm, called sequential allocation (SA), which generates optimal solutions for both constantly backlogged and generally backlogged systems, is presented. To speed up resource allocation, a resource-block-based EA (RBEA) and a generalized RBEA (GRBEA) are proposed, respectively, for constantly backlogged and generally backlogged systems. To further reduce the computational complexities of RBEA and GRBEA, we propose suboptimal algorithms called MEA+SA and GEA+SA for constantly backlogged and generally backlogged systems, respectively. The basic idea of MEA+SA (or GEA+SA) is to perform MEA (respectively, GEA) to get a preliminary solution and then execute SA for the rest of resource blocks. Simulations are conducted to evaluate the performances of the proposed resource allocation algorithms for various sizes of resource block and user numbers. Results show that the performances of the proposed suboptimal algorithms are close to those of the optimal algorithms. Moreover, our proposed algorithms perform better than previous works.

The rest of this paper is organized as follows. Problem formulation is described in Section II. In Section III, we review related works. Optimal fluid-flow- and resource-block-based resource allocation algorithms are presented in Sections IV and V, respectively. Simulation results are provided and discussed in Section VI. Finally, we draw our conclusion in Section VII.

II. Problem Formulation

Consider the downlink transmission of a wireless network that consists of a base station (BS) and a group of users. Let Ω represent the set of users. When the BS needs to perform resource allocation, it is assumed that both queue and channel statuses of users are known. This is an acceptable assumption because the BS maintains the queues and channel statuses can be obtained through status report or estimation. For each user,

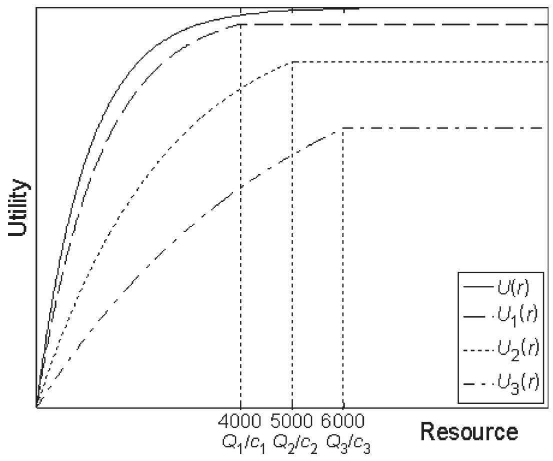

Fig. 1. Example of $U(r)$ and $U_i(r)$, where $U(r) = 1 - e^{(-r/1000)}$, $(c_1, c_2, c_3) = (0.8, 0.4, 0.2)$, and $(Q_1, Q_2, Q_3) = (3200, 2000, 1200)$.

an adaptive modulation and coding (AMC) scheme is adopted according to its channel status to maximize transmission rate while keeping the probability of transmission error under an acceptable threshold. For user i, let Q_i and c_i denote, respectively, the amount of data buffered in its queue (called queue i) and the channel quality parameter. In this paper, Q_i is expressed in terms of required resource assuming that data are transmitted by the maximum achievable rate among the available AMC schemes, and c_i is defined as the ratio of the adopted AMC scheme to the maximum achievable rate and, therefore, is in the range [0, 1]. Note that we have $Q_i = \infty$ if queue i is assumed to be constantly backlogged and $0 \leq Q_i \leq \infty$ if it is assumed to be generally backlogged. The utility function is denoted by $U(r)$, where r is the allocated resource, assuming that the adopted AMC scheme yields the maximum achievable rate. We assume that the utility function $U(r)$ is an increasingly strictly concave function so that the marginal utility function, which is defined by $u(r) = (U(r)/dr)$, exists for all r and satisfies $u(r) > 0$ and $u'(r) = (du(r)/dr) < 0$. Let $U_i(r)$ and $u_i(r)$ represent, respectively, the utility and marginal utility functions for user i. We have

$$u_i(r) = \begin{cases} c_i \cdot u_i(r), & \text{if } r \leq \frac{Q_i}{c_i} \\ 0, & \text{otherwise} \end{cases}$$

$$U_i(r) = \int_0^r u_i(s) ds. \tag{2}$$

It is obvious that $U_i(r)$ is a concave utility function and strictly concave in the range $[0, (Q_i/c_i)]$. Examples of $U(r)$ and $U_i(r)$ are depicted in Fig. 1.

Given Q_i and c_i for all $i \in \Omega$, our objective is to maximize $\sum_{i\in\Omega} U_i(r_i)$ subject to $\sum_{i\in\Omega} r_i \leq r_{\text{total}}$ and $r_i \geq 0$ for all $i \in \Omega$, where r_i and r_{total} represent, respectively, the amount of resource allocated to user i and the total resource. A resource allocation $\pi(\Omega) = \{r_i | i \in \Omega\}$ is said to be feasible if it satisfies $\sum_{i\in\Omega} r_i \leq r_{\text{total}}$ and $r_i \geq 0$ for all $i \in \Omega$. Furthermore, $\pi(\Omega) = \{r_i | i \in \Omega\}$ is said to be an optimal resource allocation if and only if (iff) 1) it is feasible, and 2) it holds that $\sum_{i\in\Omega} U_i(r_i) \geq \sum_{i\in\Omega} U_i(r'_i)$ for any feasible resource allocation $\pi'(\Omega) = \{r'_i | i \in \Omega\}$. We assume that time is divided into

TABLE I
LIST OF MAJOR NOTATIONS

Ω	User set.
r_{total}	The amount of total resource.
B	Resource block size.
x_{total}	The total number of resource blocks (i.e., $x_{total}=\frac{r_{total}}{B}$).
$U(r)$	Utility function, assuming the adopted AMC scheme yields the maximum achievable rate.
$u(r)$	Marginal utility function, assuming the adopted AMC scheme yields the maximum achievable rate.
r_i	The amount of resource allocated to user i.
x_i	The number of resource blocks allocated to user i.
c_i	Channel quality parameter which is defined as the ratio of the adopted AMC scheme to the maximum achievable rate.
Q_i	The amount of buffered data belonging to user i.
$U_i(r)$	Utility function of user i which is shown in equation (2).
$u_i(r)$	Marginal utility function of user i which is shown in equation (1).
$u_i^{-1}(x)$	Inverse marginal utility function of user i.
$u_{\sum}^{-1}(x;\Omega)$	Aggregate inverse marginal utility function for all users in Ω (i.e., $u_{\sum}^{-1}(x;\Omega)=\sum_{i\in\Omega}u_i^{-1}(x)$).
$u_{\sum}(r;\Omega)$	Aggregate marginal utility function for all users in Ω which can be obtained by taking the inverse of $u_{\sum}^{-1}(x;\Omega)$.
$\Delta U_i[j]$	The utility increment of user i after receiving its j^{th} resource block.

frames, and resource allocation is performed at the beginning of each frame.

The above formulation is applicable to resource allocation in OFDMA-based systems such as WiMAX and LTE. For better comprehensibility, major notations used in this paper are summarized in Table I.

III. RELATED WORKS

In this section, we review the EA [3] and the PF-MUX algorithms [12].

A. EA Algorithm

The EA algorithm was designed for fluid-flow-based systems, where queues are constantly backlogged. As stated in [3], a resource allocation $\pi(\Omega)=\{r_i|i\in\Omega\}$ is optimal if it is marginally fair and satisfies $\sum_{i\in\Omega} r_i = r_{\text{total}}$. Here, marginal fairness means $u_i(r_i)=u_j(r_j)\geq u_k(r_k)$ for all users i, j, and k belonging to Ω with $r_i>0, r_j>0$, and $r_k=0$. Based on this necessary and sufficient condition, the following EA algorithm was proposed in [3].

Algorithm 1: EA [3]

```
begin
1) Derive the inverse marginal utility function of each user (i.e., $u_i^{-1}(x)$, $i\in\Omega$).
2) Compute the aggregate inverse marginal utility function $u_{\sum}^{-1}(x;\Omega)=\sum_{i\in\Omega}u_i^{-1}(x)$.
3) Find the aggregate marginal utility function $u_{\sum}(r;\Omega)$ by taking the inverse of $u_{\sum}^{-1}(x;\Omega)$.
4) Determine the aggregate marginal utility with respect to $r_{\text{total}}$ (i.e., $u_{\sum}(r_{\text{total}};\Omega)$).
5) The EA $\pi_{EA}(\Omega)=\{r_i^*|i\in\Omega\}$ is given by
   $$r_i^*=\begin{cases} u_i^{-1}\left(u_{\sum}(r_{\text{total}};\Omega)\right), & \text{if } u_i(0)\geq u_{\sum}(r_{\text{total}};\Omega)\\ 0, & \text{otherwise}\end{cases}$$
end
```

The basic idea of the EA algorithm is to first determine the aggregate marginal utility, i.e., the value x that satisfies $\sum_{i\in\Omega}u_i^{-1}(x)=r_{\text{total}}$ or $u_{\sum}(r_{\text{total}};\Omega)$, and then compute the share of resource for each user based on $u_{\sum}(r_{\text{total}};\Omega)$. Unfortunately, its solution is infeasible if there exists user i such that $u_i(0)<u_{\sum}(r_{\text{total}};\Omega)$. The reason is as follows. The assumption that the utility function is increasing strictly concave implies both $u_i(r)$ and $u_i^{-1}(x)$ are decreasing for all $i\in\Omega$. Consequently, it is true that $u_k^{-1}(u_{\sum}(r_{\text{total}};\Omega))<0$ if $u_k(0)<u_{\sum}(r_{\text{total}};\Omega)$. According to the operation of the EA algorithm, we have $u_i(u_{\sum}^{-1}(r_{\text{total}};\Omega))=u_j(u_{\sum}^{-1}(r_{\text{total}};\Omega))$ (in particular, $u_i(u_{\sum}^{-1}(r_{\text{total}};\Omega))=u_j(u_{\sum}^{-1}(r_{\text{total}};\Omega))$ for all i, $j\in\Omega$, and $\sum_{i\in\Omega}u_i^{-1}(u_{\sum}(r_{\text{total}};\Omega))=r_{\text{total}}$). Assume that user k is the only one that satisfies $u_k(0)<u_{\sum}(r_{\text{total}};\Omega)$. By simply setting $r_k^*=0$ makes $\sum_{i\in\Omega}r_i^*=\sum_{i\in\Omega}u_i^{-1}(u_{\sum}(r_{\text{total}};\Omega))-u_k^{-1}(u_{\sum}(r_{\text{total}};\Omega))=r_{\text{total}}-u_k^{-1}(u_{\sum}(r_{\text{total}};\Omega))>r_{\text{total}}$, which is a contradiction to the constraint $\sum_{i\in\Omega}r_i^*\leq r_{\text{total}}$. The following Lemma 1 states a more general case. Proofs of all lemmas and theorems are provided in Appendix A.

Lemma 1: Assume that $r_{\text{total}}>0$ and $Q_i=\infty$ for all $i\in\Omega$. Given the EA $\pi_{EA}(\Omega)=\{r_i^*|i\in\Omega\}$, it holds that $\sum_{i\in\Omega}r_i^*>r_{\text{total}}$ if $\Lambda\neq\emptyset$, where $\Lambda=\{i|i\in\Omega, u_i(0)<u_{\sum}(r_{\text{total}};\Omega)\}$.

B. PF-MUX Algorithm

In [12], an OFDMA system consisting of M subchannels is considered. Each subchannel comprises S subcarriers and is divided into T slots. The basic resource unit is equal to one slot on a subchannel. Queues are assumed to be constantly backlogged. The resource allocation problem was formulated in [12] as an optimization problem that maximizes $\sum_{i\in\Omega}\log R_i$, where R_i is the average rate of user i. A heuristic algorithm called PF-MUX was proposed to allocate subchannels to users.

Assume that the PF-MUX algorithm is to allocate the T slots of subchannel m to users. All users are sorted according to $\rho_{i,m}=(c_{i,m}/R_i')$, where $c_{i,m}$ denotes the channel quality parameter for user i on subchannel m, and R_i' represents the past average rate of user i. Let Ω_a, $a = 0, 1, 2, \ldots, \log_2 T$ contain the top 2^a users in $\rho_{i,m}$. The PF-MUX algorithm allocates to every user in $\Omega_a (T/2^a)$ slots for the maximum a that satisfies the following: 1) The average transmission rate of the users in Ω_a is greater than or equal to that of users in Ω_{a-1}, and 2) the increment of utility sum if subchannel m is allocated to users in Ω_a is greater than or equal to that if it is allocated to users in Ω_{a-1}. The process of subchannel allocation is performed for $m = 1$ to M.

IV. Resource Allocation for Fluid-Flow-Based Systems

In this section, we present optimal resource allocation algorithms for fluid-flow-based systems. The cases of constantly backlogged queues and generally backlogged queues are studied in sections A and B, respectively.

A. Constantly Backlogged Queues

As discussed in Section III, if $\Lambda = \{i | i \in \Omega, u_i(0) < u_{\sum}(r_{\text{total}}; \Omega)\} \neq \emptyset$, then $\pi_{EA}(\Omega) = \{r_i^* | i \in \Omega\}$ is not a feasible allocation because the constraint $\sum_{i\in\Omega} r_i^* \leq r_{\text{total}}$ is violated. The remedy is to remove every user $k \in \Lambda$ out of Ω and repeatedly execute the EA algorithm until a feasible solution is obtained. This algorithm, called MEA, is subsequently described.

Algorithm 2: MEA

```
Data: $\Lambda = \Gamma = \Omega$
Result: $r_i$ for all $i \in \Omega$
begin
    while $\Lambda \neq \emptyset$ do
        Perform Steps 1)-4) of the EA algorithm for user set $\Gamma$.
        Let $\Lambda = \{i | i \in \Gamma, u_i(0) < u_{\sum}(r_{\text{total}}; \Gamma)\}$
        $\Gamma = \Gamma - \Lambda$
    end
    $r_i = u_i^{-1}(u_{\sum}(r_{\text{total}}; \Gamma))$ for all $i \in \Gamma$.
    $r_i = 0$ for all $i \in \Omega - \Gamma$
end
```

A property regarding $\pi_{EA}(\Omega) = \{r_i^* | i \in \Omega\}$ and $\pi_{EA}(\Omega - \Lambda) = \{s_i | i \in \Omega - \Lambda\}$ is stated in Lemma 2. Note that r_i^* and s_i represent the amount of resource allocated to user i if the EA algorithm is performed for users in Ω and $\Omega - \Lambda$, respectively. This property is useful in proving the optimality of the proposed MEA algorithm stated in Theorem 3.

Lemma 2: Assume that $r_{\text{total}} > 0$ and $Q_i = \infty$ for all $i \in \Omega$. Given $\pi_{EA}(\Omega) = \{r_i^* | i \in \Omega\}$, $\Lambda = \{i | i \in \Omega, u_i(0) < u_{\sum}(r_{\text{total}}; \Omega)\} \neq \emptyset$, and $\pi_{EA}(\Omega - \Lambda) = \{s_i | i \in \Omega - \Lambda\}$, it holds that $u_i(s_i) > u_i(r_i^*) > u_k(0)$ for any $i \in \Omega - \Lambda$ and $k \in \Lambda$.

Theorem 3: Assume that $r_{\text{total}} > 0$ and $Q_i = \infty$ for all $i \in \Omega$. The allocation $\pi_{\text{MEA}}(\Omega) = \{r_i | i \in \Omega\}$ determined by the MEA algorithm is marginally fair and satisfies $\sum_{i\in\Omega} r_i = r_{\text{total}}$ and, thus, is optimal.

Note that Lemma 2 implies $s_i < r_i^*$ for all $i \in \Omega - \Lambda$. As a result, it is possible that $\{i | i \in \Omega - \Lambda, u_i(0) < u_{\sum}(r_{\text{total}}; \Omega - \Lambda)\} \neq \emptyset$. Moreover, the resource allocated to the same user decreases iteration by iteration. Let Γ be the user set of a particular iteration and $i \in \Gamma$, $k \in \Omega - \Gamma$. Lemma 2 implies that $u_i(s_i) > u_k(0)$ if user i is allocated resource $s_i \geq 0$ in the considered iteration.

It is interesting to analyze the time complexity of the MEA algorithm. In Appendix B, we provide the analysis of time complexities of all the algorithms proposed in this paper.

B. Generally Backlogged Queues

In this section, we consider a system where queues are generally backlogged. It is clear that if $\sum_{i\in\Omega}(Q_i/c_i) \leq r_{\text{total}}$, then $\pi(\Omega) = \{r_i | r_i = (Q_i/c_i), i \in \Omega\}$ is an optimal allocation. In fact, any feasible resource allocation $\pi'(\Omega) = \{r'_i | r'_i \geq (Q_i/c_i), i \in \Omega\}$ is optimal. Assume that $\sum_{i\in\Omega}(Q_i/c_i) > r_{\text{total}}$. Let Γ be a subset of Ω such that $i \in \Gamma$ iff $Q_i > 0$. The set Γ can be partitioned into three disjoint subsets, namely, Γ_Z, Γ_P, and Γ_A, such that user i is contained in Γ_Z, Γ_P, or Γ_A iff $r_i = 0$, $0 < r_i < (Q_i/c_i)$, or $r_i = (Q_i/c_i)$, respectively. The definition of marginal fairness is generalized as follows.

Definition: A resource allocation $\pi(\Omega) = \{r_i | i \in \Omega\}$ is said to be generalized marginally fair if 1) $r_i = 0$ if $Q_i = 0$ and 2) for any user $z \in \Gamma_Z$, p, $p' \in \Gamma_P$, and $a \in \Gamma_A$, it is true that

$$u_z(r_z) \leq u_p(r_p) = u_{p'}(r_{p'}) \leq u_a(r_a). \tag{3}$$

Theorem 4 states a necessary and sufficient condition for an optimal allocation.

Theorem 4: Assume that $r_{\text{total}} > 0$, $0 \leq Q_i \leq \infty$ for all $i \in \Omega$ and $\sum_{i\in\Omega}(Q_i/c_i) > r_{\text{total}}$. A feasible resource allocation $\pi(\Omega) = \{r_i | i \in \Omega\}$ is optimal iff it is generalized marginally fair and satisfies $\sum_{i\in\Omega} r_i = r_{\text{total}}$. Moreover, there is exactly one optimal resource allocation.

In the rest of this section, we present an optimal resource allocation algorithm, called GEA, which meets the necessary and sufficient condition stated in Theorem 4. The basic idea of GEA is to first place users to their appropriate subsets (Γ_Z, Γ_P, or Γ_A) and then calculate their resource shares. To accomplish this, we set $r_i = 0$ for all $i \in \Gamma$, $\Gamma_Z = \Gamma$, and $\Gamma_P = \Gamma_A = \emptyset$ initially. There are three possibilities in each iteration: 1) A user is moved from Γ_Z to Γ_P; 2) a user is moved from Γ_P to Γ_A; and 3) all users are in their appropriate subsets and the optimal solution is obtained. As an example, assume that in some iteration user j is moved from Γ_Z to Γ_P. For this event to happen, the conditions to be met are the following: 1) $\sum_{i\in\Gamma_P} u_i^{-1}(u_j(0)) + \sum_{i\in\Gamma_A}(Q_i/c_i) < r_{\text{total}}$ (not all users are in their appropriate subsets); 2) $u_j(0) = \max_{i\in\Gamma_Z} u_i(0)$ (no user other than user j is moved from Γ_Z to Γ_P); and 3) $u_j(0) \geq \max_{i\in\Gamma_P} u_i(Q_i/c_i)$ (no user is moved from Γ_P to Γ_A). The conditions for other events to happen can be determined similarly. After all users are placed in the correct subsets, the solution can be obtained by solving (3) and $\sum_{i\in\Omega} r_i = r_{\text{total}}$. To summarize, we repeatedly check if it holds that

$$\sum_{i\in\Gamma_P} u_i^{-1}(u) + \sum_{i\in\Gamma_A} \frac{Q_i}{c_i} \geq r_{\text{total}} \tag{4}$$

where

$$u = \max(\max_{i\in\Gamma_Z} u_i(0), \max_{i\in\Gamma_P} u_i\left(\frac{Q_i}{c_i}\right). \tag{5}$$

If not, user i^*, chosen according to the following equation, is moved from one set to another:

$$i^* = \arg\max_{i\in\Gamma_Z\cup\Gamma_P}\left(\max_{i\in\Gamma_Z} u_i(0), \max_{i\in\Gamma_P} u_i\left(\frac{Q_i}{c_i}\right)\right). \tag{6}$$

All users are placed in their appropriate subsets if inequality (4) holds. Once inequality (4) holds, the solution can be obtained as follows. Set $r_i = 0$ for all user $i \in \Gamma_Z \cup (\Omega - \Gamma)$, $r_i = u_i^{-1}(u_{\sum}(r_{\text{total}} - \sum_{i\in\Gamma_A}(Q_i/c_i); \Gamma_P))$ for all user $i \in \Gamma_P$, and $r_i = (Q_i/c_i)$ for all user $i \in \Gamma_A$. The pseudocode of GEA is provided in the following.

Algorithm 3: GEA

```
Data:
    1) $\Gamma = \{i | Q_i > 0, i \in \Omega\}$
    2) $\Gamma_Z = \Gamma$
    3) $\Gamma_P = \Gamma_A = \emptyset$
Result: $r_i$ for all $i \in \Omega$
begin
    if $\sum_{i\in\Omega}(Q_i/c_i) \leq r_{\text{total}}$ then
        $r_i = (Q_i/c_i)$ for all $i \in \Gamma$
        $\Gamma_A = \Gamma$
    else
        while (1) do
            $u = \max(\max_{i\in\Gamma_Z} u_i(0), \max_{i\in\Gamma_P} u_i(Q_i/c_i))$
            if $\sum_{i\in\Gamma_P} u_i^{-1}(u) + \sum_{i\in\Gamma_A}(Q_i/c_i) \geq r_{\text{total}}$ then
                $r_i = 0$ for all user $i \in \Gamma_Z \cup (\Omega - \Gamma)$
                $r_i = u_i^{-1}(u_{\sum}(r_{\text{total}} - \sum_{i\in\Gamma_A}(Q_i/c_i); \Gamma_P))$
                 for all user $i \in \Gamma_P$
                (User $i$ is moved from $\Gamma_P$ to $\Gamma_A$ if $r_i = (Q_i/c_i)$.)
                $r_i = (Q_i/c_i)$ for all user $i \in \Gamma_A$
                exit
            else
                $i^* = \arg\max_{i\in\Gamma_Z\cup\Gamma_P}$
                      $(\max_{i\in\Gamma_Z} u_i(0), \max_{i\in\Gamma_P} u_i(Q_i/c_i))$
                if $i^* \in \Gamma_Z$ then
                    $\Gamma_Z = \Gamma_Z - \{i^*\}$
                    $\Gamma_P = \Gamma_P \cup \{i^*\}$
                else
                    $\Gamma_P = \Gamma_P - \{i^*\}$
                    $\Gamma_A = \Gamma_A \cup \{i^*\}$
                end
            end
        end
    end
end
```

V. Resource Allocation for Resource-Block-Based Systems

In this section, we consider resource allocation for resource-block-based systems. Let B denote the size of a resource block. We assume that r_{total} is divisible by B, and let $x_{\text{total}} = (r_{\text{total}}/B)$ denote the total number of resource blocks. An additional constraint, i.e., $r_i = x_i \cdot B$ for all $i \in \Omega$, is added to the problem formulation. Here, x_i and r_i represent, respectively, the number of resource blocks and the amount of resource allocated to user i. Again, the cases of constantly backlogged queues and generally backlogged queues are considered separately in two different sections.

A. Constantly Backlogged Queues

Let

$$\Delta U_i[j] = U_i(j \cdot B) - U_i((j-1) \cdot B) \tag{7}$$

be the utility increment of user i after receiving its jth resource block. Since the utility function is strictly concave, we have $U_i[j] > U_i[j+1]$, $1 \leq j \leq x_{\text{total}} - 1$. The definition of marginal fairness for resource-block-based systems is described as follows.

Definition: A resource allocation $\pi(\Omega) = \{x_i | i \in \Omega\}$ is said to be resource-block-based generalized marginally fair if, for any users i, $m \in \Omega$ and $x_i > 0$, it holds that $\Delta U_i[x_i] \geq \Delta U_m[x_m + 1]$.

Theorem 5 states a necessary and sufficient condition for a resource-block-based allocation to be optimal for constantly backlogged queues.

Theorem 5: Assume that $x_{\text{total}} > 0$ and $Q_i = \infty$ for all $i \in \Omega$. A feasible resource-block-based allocation $\pi(\Omega) = \{x_i | i \in \Omega\}$ is optimal iff it is resource-block-based marginally fair and satisfies $\sum_{i\in\Omega} x_i = x_{\text{total}}$.

An optimal allocation algorithm called SA finds $i^* = \arg\max_{i\in\Omega} \Delta U_i[x_i + 1]$, allocates one resource block to user i^*, and updates $x_{i^*} = x_{i^*} + 1$. The process repeats until no more resource block is available. Of course, the initial value of x_i is set to zero for all $i \in \Omega$. It is clear that the SA algorithm obtains an optimal solution because the resource-block-based marginal fairness is maintained after each resource block is allocated.

It is possible to allocate more than one resource block in each iteration. The allocation algorithm works as follows. Again, the initial value is set to zero for all x_i. If $\Omega = \{i\}$, we simply set $x_i = x_i + x_{\text{total}}$. Assume that $|\Omega| > 1$. A user is removed from Ω if it will not receive any more resource block. Let

$$j^* = \arg\min_{j\in\Omega} \Delta U_j[x_j + 1] \tag{8}$$

and define

$$\mu_i[x_i] = \max\{m \geq 0 | \Delta U_i[x_i + m] \geq \Delta U_{j^*}[x_{j^*} + 1]\} \tag{9}$$

for all $i \in \Omega$. To maintain resource-block-based marginal fairness, we allocate to user i $x_i + \mu_i[x_i]$ resource blocks and to user j^* x_{j^*+1} resource blocks. To calculate $\mu_i[x_i]$, we define a function $g_i(x)$, for all $x \geq 0$, as

$$g_i(x) = U(x \cdot B \cdot c_i) - U((x-1) \cdot B \cdot c_i). \tag{10}$$

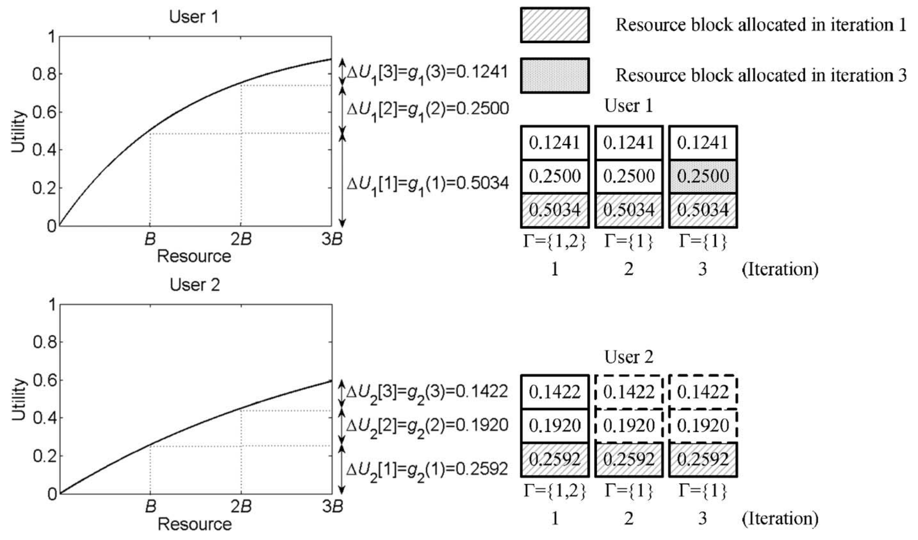

Fig. 2. Example of RBEA for $U(r) = 1 - e^{(-r/1000)}$, $x_{\text{total}} = 3$, $B = 1000$, and $(c_1, c_2) = (0.7, 0.3)$.

It is clear that $g_i(x)$ is a continuous and decreasing function of x for all $x \geq 0$. As a result, $g_i^{-1}(u)$, the inverse function of $g_i(x)$, exists, and $\mu_i[x_i]$ can be computed by

$$\mu_i[x_i] = \lfloor g_i^{-1}\left(\Delta U_{j^*}[x_{j^*}+1]\right)\rfloor - x_i. \qquad (11)$$

Note that we need to check whether such an allocation is feasible. If $\sum_{i\in\Omega}\mu_i[x_i] \leq x_{\text{total}}$, then it is feasible, and therefore, we update $x_i = x_i + \mu_i[x_i]$ for all $i \in \Omega$ and $x_{\text{total}} = x_{\text{total}} - \sum_{i\in\Omega}\mu_i[x_i]$. Otherwise, user j^* will not receive any more resource block and thus is removed from Ω. This process repeats until $x_{\text{total}} = 0$. We call the preceding algorithm RBEA and subsequently summarize its procedure.

Algorithm 4: RBEA

```
Data:
    1) $\Gamma = \Omega$
    2) $x_i = 0$ for all $i \in \Gamma$
**Result**: $x_i$ for all $i \in \Gamma$
**begin**
    **while** $x_{\text{total}} > 0$ **do**
      **if** $|\Gamma| > 1$ **then**
        $j^* = \arg\min_{j\in\Gamma} \Delta U_j[x_j+1]$
        $\mu_i[x_i] = \lfloor g_i^{-1}(\Delta U_{j^*}[x_{j^*}+1])\rfloor - x_i$ for all $i \in \Gamma$
        **if** $\sum_{i\in\Gamma}\mu_i[x_i] \leq x_{\text{total}}$ **then**
          $x_i = x_i + \mu_i[x_i]$ for all $i \in \Gamma$
          $x_{\text{total}} = x_{\text{total}} - \sum_{i\in\Gamma}\mu_i[x_i]$
        **else**
          $\Gamma = \Gamma - \{j^*\}$
        **end**
      **else**
        $x_i = x_i + x_{\text{total}}, i \in \Gamma$
        $x_{\text{total}} = 0$
      **end**
    **end**
**end**
```

As an example, consider a system that consists of two users and $x_{\text{total}} = 3$. As shown in Fig. 2, we have $j^* = 2$, $\mu_1[x_1] = 1$, and $\mu_1[x_2] = 1$ in iteration 1. Therefore, two resource blocks are allocated, one to each user. In iteration 2, we have $j^* = 2$, $\mu_1[x_1] = 1$, and $\mu_1[x_2] = 1$ again. However, the remaining resource, which is only one block, is insufficient for allocation. Consequently, user 2 is removed from the user set, and user 1 will be allocated the remaining resource block in iteration 3.

It is not hard to see that RBEA always maintains the resource-block-based marginal fairness property and, thus, obtains an optimal solution. According to the simulation results presented in the next section, the execution time of the RBEA algorithm is smaller than that of the SA algorithm as long as the block size B is small, which is normally the case in a real system.

To further speed up resource allocation, one can use the MEA algorithm to find a preliminary solution $\pi_{\text{MEA}}(\Omega) = \{r_i | i \in \Omega\}$, take the floor function $\lfloor(r_i/B)\rfloor$ for all $i \in \Omega$ to get an initial feasible solution, and finally perform SA, starting from the initial feasible solution, to obtain a suboptimal solution. We will refer to such a solution as MEA+SA solution and compare it with the optimal solution in the next section.

B. Generally Backlogged Queues

For the case of generally backlogged queues, the definition of $u_i(r)$ [see equation (1)] implies $\Delta U_i[j] > \Delta U_i[j+1]$ if $1 \leq j \leq \lceil(Q_i/c_i)\rceil$ and $\Delta U_i[j] = 0$ if $j \geq \lceil(Q_i/c_i)\rceil + 1$. Note that if $\delta = \lfloor(Q_i/B \cdot c_i)\rfloor < (Q_i/B \cdot c_i)$, then we have $\Delta U_i[\delta + 1] = U_i(Q_i/c_1) - U_i(B \cdot \delta)$. Obviously, $x_i = \lceil(Q_i/B \cdot c_i)\rceil$ for all $i \in \Omega$ is an optimal allocation if $\sum_{i\in\Omega}\lceil(Q_i/B \cdot c_i)\rceil \leq x_{\text{total}}$. Assume that $\sum_{i\in\Omega}\lceil(Q_i/B \cdot c_i)\rceil > x_{\text{total}}$. The condition stated in Theorem 5 for constantly backlogged queues is also a necessary and sufficient condition for an allocation to be optimal for generally backlogged queues. The proof is straightforward and, thus, omitted.

It is clear that SA can directly be applied to find an optimal allocation for generally backlogged queues because the resource-block-based marginal fairness is maintained after each resource block is allocated. For the RBEA algorithm, slight modification in calculating $\mu_i[x_i]$ is needed. Equation (11) is valid only if $\lfloor g_i^{-1}(\Delta U_{j^*}[x_{j^*}+1])\rfloor - x_i \leq \lfloor (Q_i/B \cdot c_i)\rfloor$. In case $\lfloor g_i^{-1}(\Delta U_{j^*}[x_{j^*}+1])\rfloor - x_i > \lfloor (Q_i/B \cdot c_i)\rfloor$, we have $\mu_i[x_i] = \lceil (Q_i/B \cdot c_i)\rceil$ if $\Delta U_{j^*}[x_{j^*}+1] \leq \Delta U_i[\lceil (Q_i/B \cdot c_i)\rceil]$ or $\mu_i[x_i] = \lfloor (Q_i/B \cdot c_i)\rfloor$ otherwise. The modified version is named GRBEA algorithm. Note that when performing GRBEA, we have to update the corresponding queue status after allocating resource block(s) to some user. The pseudocode for GRBEA is subsequently presented.

Algorithm 5: GRBEA

```
Data:
    1) $\Gamma = \{i | Q_i > 0, i \in \Omega\}$
    2) $x_i = 0$ for all $i \in \Omega$
Result: $x_i$ for all $i \in \Gamma$
begin
    if $\sum_{i\in\Omega} \lceil (Q_i/B \cdot c_i)\rceil \leq x_{\text{total}}$ then
        $x_i = \lceil (Q_i/B \cdot c_i)\rceil$ for all $i \in \Gamma$
    else
        while $x_{\text{total}} > 0$ and $\Gamma \neq \emptyset$ do
            if $|\Gamma| > 1$ then
                $j^* = \arg\min_{j\in\Gamma} \Delta U_j[x_j + 1]$
                for all $i \in \Gamma$ do
                    if $\lfloor g_i^{-1}(\Delta U_{j^*}[x_{j^*}+1])\rfloor - x_i \leq \lfloor (Q_i/B \cdot c_i)\rfloor$
                    then
                        $\mu_i[x_i] = \lfloor g_i^{-1}(\Delta U_{j^*}[x_{j^*}+1])\rfloor - x_i$
                    else
                        if $\Delta U_{j^*}[x_{j^*}+1] \leq \Delta U_i[\lceil (Q_i/B \cdot c_i)\rceil]$
                        then
                            $\mu_i[x_i] = \lceil (Q_i/B \cdot c_i)\rceil$
                        else
                            $\mu_i[x_i] = \lfloor (Q_i/B \cdot c_i)\rfloor$
                        end
                    end
                end
                if $\sum_{i\in\Gamma} \mu_i[x_i] \leq x_{\text{total}}$ then
                    for all $i \in \Gamma$ do
                        $x_i = x_i + \mu_i[x_i]$
                        if $Q_i \leq \mu_i[x_i] \cdot B \cdot c_i$ then
                            $Q_i = 0$
                            $\Gamma = \Gamma - \{i\}$
                        else
                            $Q_i = Q_i - \mu_i[x_i] \cdot B \cdot c_i$
                        end
                    end
                    $x_{\text{total}} = x_{\text{total}} - \sum_{i\in\Gamma} \mu_i[x_i]$
                else
                    $\Gamma = \Gamma - \{j^*\}$
                end
            else
                if $Q_i \leq x_{\text{total}} \cdot B \cdot c_i$ then
                    $Q_i = 0$
                    $\Gamma = \Gamma - \{i\}$
                    $x_{\text{total}} = x_{\text{total}} - \lceil (Q_i/B \cdot c_i)\rceil$
                else
                    $Q_i = Q_i - x_{\text{total}} \cdot B \cdot c_i$
                    $x_{\text{total}} = 0$
                end
            end
        end
    end
end
```

An example of GRBEA is shown in Fig. 3, where a system that consists of two users with $x_{\text{total}} = 3$ is considered. In iteration 1, we have $j^* = 2$, $\mu_1[x_1] = 1$ and $\mu_2[x_2] = 1$. Therefore, two resource blocks are allocated: one to each user. In iteration 2, we have $j^* = 2$, $\mu_1[x_1] = 1$, and $\mu_2[x_2] = 1$. Note that, in this iteration, the utility increment of user 1 is limited by its queue occupancy. The remaining resource is not enough to allocate one resource block to each user. Consequently, user 1 is removed from the user set in this iteration, and user 2 will be allocated the remaining resource block in iteration 3.

It is clear that the resource-block-based marginal fairness is kept after each iteration. Therefore, the solution obtained by GRBEA is an optimal allocation. Again, according to the simulation results presented in the next section, the execution time of the GRBEA algorithm is smaller than that of SA as long as the block size B is small.

Similar to MEA+SA design, we can also derive a preliminary solution $\pi_{GEA}(\Omega) = \{r_i | i \in \Omega\}$ based on the GEA algorithm. An initial feasible solution can then be calculated by taking $\lfloor (r_i/B)\rfloor$ for all $i \in \Omega$. After updating the number of remaining resource blocks and queue statuses, one can perform SA to obtain a suboptimal solution. Such an algorithm will be referred to as GEA+SA.

VI. Simulation Results

In this section, we evaluate the performances of the proposed algorithms. Two types of increasing strictly concave utility functions are considered. The investigated system is first described, followed by two simulation scenarios. Queues are constantly backlogged and generally backlogged in Scenarios 1 and 2, respectively.

A. *Investigated System*

In the investigated system, there is a cell with one BS and several users. Users are uniformly distributed in a circular area of radius 1 km, and the BS is located at the center. We consider the downlink of an OFDMA-based IEEE 802.16 WiMAX wireless network that consists of M subchannels and is operated under the partial usage subchannel mode so that the subchannel qualities of a specific user are identical. Every subchannel comprises S subcarriers and is divided into T slots. As a result, we have $r_{\text{total}} = M \cdot S \cdot T$. In the simulations, we set $M = 30$, $S = 25$, and $T = 10$, which imply $r_{\text{total}} = 7500$. Signals are transmitted with power equal to 1 W, attenuated due to path loss with exponent equal to 3, faded according to Rayleigh fading model, and finally suffered interference and noise with their total average power equal to -98 dBm. The available AMC schemes, which are the same as those adopted

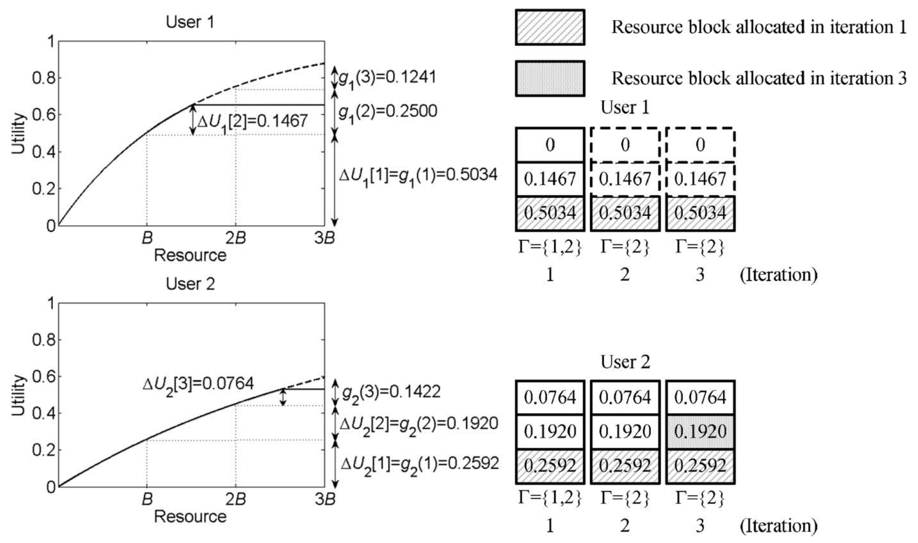
User 1
Utility
Resource
$g_1(3)=0.1241$
$g_1(2)=0.2500$
$\Delta U_1[2]=0.1467$
$\Delta U_1[1]=g_1(1)=0.5034$
Resource block allocated in iteration 1
Resource block allocated in iteration 3
User 2
$\Delta U_2[3]=0.0764$
$g_2(3)=0.1422$
$\Delta U_2[2]=g_2(2)=0.1920$
$\Delta U_2[1]=g_2(1)=0.2592$
Γ={1,2} Γ={2} Γ={2}
1 2 3 (Iteration)

Fig. 3. Example of GRBEA for $U(r) = 1 - e^{(-r/1000)}$, $x_{\text{total}} = 3$, $B = 1000$, $(c_1, c_2) = (0.7, 0.3)$, and $(Q_1, Q_2) = (1050, 750)$.

TABLE II
ADOPTED MODULATION AND CODING SCHEMES [15]

Mode	Modulation	Coding rate	Receiver SNR(dB)
1	QPSK	1/2	5
2	QPSK	3/4	8
3	16QAM	1/2	10.5
4	16QAM	3/4	14
5	64QAM	1/2	16
6	64QAM	2/3	18
7	64QAM	3/4	20

TABLE III
UTILITY SUM OF MEA AND RESOURCE-BLOCK-BASED ALLOCATION ALGORITHMS

		Average Utility Sum		
		$\Omega = 10$	$\Omega = 20$	$\Omega = 30$
MEA		8.2662	13.0352	16.1011
$B = 25$	MEA+SA	8.2659	13.0337	16.0977
	SA	8.2659	13.0337	16.0978
	RBEA	8.2659	13.0337	16.0978
$B = 250$	MEA+SA	8.2239	12.7560	15.7378
	SA	8.2239	12.7569	15.7378
	RBEA	8.2239	12.7569	15.7378

TABLE IV
EXECUTION TIME OF MEA AND RESOURCE-BLOCK-BASED ALLOCATION ALGORITHMS

		Average Execution Time (ms)		
		$\Omega = 10$	$\Omega = 20$	$\Omega = 30$
MEA		0.0718	0.2012	0.3916
$B = 25$	MEA+SA	0.1139	0.2902	0.5366
	SA	3.0623	3.1372	3.2370
	RBEA	0.8315	1.0873	1.3806
$B = 250$	MEA+SA	0.1123	0.3120	0.4727
	SA	0.3120	0.3276	0.3292
	RBEA	0.4306	0.6508	1.2230

in [15], are shown in Table II. Which AMC scheme is used depends on the receiver signal-to-interference-plus-noise ratio. Simulations are performed for 10, 20, and 30 users. For a resource-block-based system, the size of a resource block is 25 or 250, which correspond to one slot or one subchannel (10 slots), respectively. Simulations are performed using Matlab for 10 000 times on a computer equipped with an Intel Core 2 Q8200 CPU operated at 2.33 GHz with 3 GB of RAM.

B. Scenario 1

In Scenario 1, user queues are assumed to be constantly backlogged. The utility function of user i is $U_i(r) = 1 - e^{(-r \cdot c_i \cdot c_{\max}/1000)}$, where c_i denotes the channel quality of user i, and $c_{\max}$ represents the maximum achievable rate of the adopted AMC schemes. The utility sum and average execution time of the proposed MEA, SA, RBEA, and MEA+SA are shown in Tables III and IV, respectively. As one can see, for each proposed algorithm, the utility sum increases as the number of users increases. This is because of user diversity. However, the execution time increases as well. Both utility sum and execution time decrease as the resource block size increases. For $B = 25$, SA spends more time than RBEA to obtain the solution. The reason is that RBEA allocates more than one resource block in each iteration, whereas SA allocates only one. On the contrary, SA is faster than RBEA for $B = 250$. This is because RBEA is more complicated than SA and for $B = 250$, the number of resource blocks is only 30, which limits the advantage of allocating more than one resource block in an iteration. An interesting observation is that if the user number is fixed, the execution time of MEA+SA is almost independent of resource block size. The reason is that, after taking the floor function, the number of remaining resource blocks is a random variable whose value falls in $\{0, 1, 2, \ldots, |\Omega| - 1\}$, and the results were averaged over 10 000 simulations. Based on simulation results, we conclude that, for small resource blocks, for example, $B = 25$, MEA+SA should be the best choice because it requires 83% and 61% less execution time as compared with SA and RBEA, respectively, while sacrificing very little utility sum (which is 1×10^{-5} or 6×10^{-4}%).

TABLE V
UTILITY SUM OF GEA AND RESOURCE-BLOCK-BASED ALLOCATION ALGORITHMS

		Average Utility Sum		
		$\Omega = 10$	$\Omega = 20$	$\Omega = 30$
GEA		23.3896	32.9941	37.3684
PF-MUX		23.1011	30.8788	31.3548
$B = 25$	GEA+SA	23.3766	32.9802	37.3624
	SA	23.3788	32.9802	37.3629
	GRBEA	23.3788	32.9802	37.3629
$B = 250$	GEA+SA	23.3743	32.9726	37.3124
	SA	23.3743	32.9726	37.3151
	GRBEA	23.3743	32.9726	37.3151

TABLE VI
EXECUTION TIME OF GEA AND RESOURCE-BLOCK-BASED ALLOCATION ALGORITHMS

		Average Execution Time		
		$\Omega = 10$	$\Omega = 20$	$\Omega = 30$
GEA		0.7082	1.2776	1.9001
PF-MUX		0.3136	0.3182	0.3307
$B = 25$	GEA+SA	0.7940	1.4134	2.0608
	SA	3.8610	4.0123	4.2198
	GRBEA	1.2667	1.3026	2.0592
$B = 250$	GEA+SA	0.7706	1.3993	2.0779
	SA	0.4212	0.4290	0.4540
	GRBEA	0.4930	0.6833	0.9953

C. Scenario 2

In Scenario 2, user queues are assumed to be generally backlogged. An ON–OFF model is adopted to describe the traffic arrival process for each user. The lengths of ON and OFF periods are governed by exponential distribution with means equal to 0.01 and 0.03 s, respectively. During the ON period, data arrive at a rate uniformly chosen from 1 to 5 Mb/s and are encapsulated into packets of size equal to 1500 B. The utility function of user i is selected as $U_i(r) = \log(R_i[n])$, where $R_i[n]$ denotes the average throughput of user i up to frame n. In other words, we study the capability of the proposed algorithms to achieve proportional fairness. In our experiments, $R_i[n]$ is updated by

$$R_i[n] = \begin{cases} \left(1 - \frac{1}{w}\right) R_i[n-1] + \frac{r \cdot c_i \cdot c_{\max}}{w \cdot M \cdot T}, & \text{if } Q_i > 0 \\ R_i[n-1], & \text{otherwise} \end{cases} \quad (12)$$

where w, which is a weighting factor, is a design parameter. We set $w = 5$ in the experiments.

Simulations are performed for five resource allocation algorithms, i.e., GEA, SA, GEA+SA, GRBEA, and PF-MUX. For fair comparison, the PF-MUX algorithm is modified to consider queue statuses. When subchannel m is to be allocated to users in Ω_a, it is only partially allocated if the total resource requests of users in Ω_a is smaller than T. The remaining part of subchannel m can be allocated to other users.

The utility sum and average execution time are shown in Tables V and VI, respectively. Again, the effect of user diversity yields a larger utility sum as the number of users increases, and there is tradeoff between resource block size and execution time. Moreover, the execution time of GEA+SA is almost independent of resource block size if user number is fixed. According to simulation results, PF-MUX requires the least execution time. However, its utility sum is the smallest, as compared with those of GEA, GEA+SA, and GRBEA. The difference increases as the number of users increases. For $|\Omega| = 30$, the difference is about 19%. The reason is that when the number of users is small, the total resource of a frame is usually large enough to completely serve all data buffered in the queues. Consequently, the solutions obtained by all the studied algorithms are identical for most of the time. On the other hand, when the number of users becomes large, there are usually some data left in the queues at the end of every frame. As a result, the suboptimal solutions are different from the optimal ones. In particular, the solution obtained by PF-MUX can be far from being optimal. For $B = 25$ and $|\Omega| = 10$, SA, GRBEA, and GEA+SA yield about the same utility sum. However, GEA+SA requires an execution time that is 79% and 28% less than those of SA and GRBEA, respectively. Therefore, we conclude that GEA+SA should be the best choice for a system when resource block size and number of users are small. For a system with large resource block sizes, for example, $B = 250$, the execution times of SA and GRBEA can drop dramatically, whereas that of GEA only changes slightly. Therefore, we suggest using either SA or GRBEA as the resource allocation algorithm for such a system. As in Scenario 1, SA is faster than GRBEA for $B = 250$.

VII. CONCLUSION

In this paper, we have presented optimal resource allocation algorithms that maximize utility sum for wireless networks. The resource is either infinitesimally divisible or has a basic unit and queues can be either constantly backlogged or generally backlogged. Numerical results show that one can combine the optimal algorithm designed for fluid-flow-based systems with SA to quickly obtain a suboptimal solution for a resource-block-based system with small block sizes. The performance of such a simple hybrid algorithm is close to that of the optimal algorithm. For large block sizes, which tend to decrease the utility sum and thus may not be a good design, one can adopt either SA or RBEA to do resource allocation. An interesting further research topic is to extend the results to general utility functions.

APPENDIX A
PROOFS OF LEMMAS 1 AND 2 AND THEOREMS 3–6

Proof of Lemma 1: According to the operation of the EA algorithm, we have $r_{\text{total}} = \sum_{i\in\Omega} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega))= \sum_{i\in\Omega-\Lambda} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega))+\sum_{k\in\Lambda} u_k^{-1}(u_\Sigma(r_{\text{total}};\Omega))$. Since $u_k^{-1}(x)$ is decreasing, it holds that $u_k^{-1}(u_\Sigma(r_{\text{total}};\Omega) < 0$ if $k \in \Lambda$, which implies, when $\Lambda \neq \emptyset$, $\sum_{i\in\Omega} r_i^* = \sum_{i\in\Omega-\Lambda} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega)) > \sum_{i\in\Omega-\Lambda} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega)) + \sum_{i\in\Lambda} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega))=\sum_{i\in\Omega} u_i^{-1}(u_\Sigma(r_{\text{total}};\Omega))= r_{\text{total}}$. This completes the proof of Lemma 1. ■

Proof of Lemma 2: Assume that $i \in \Omega - \Lambda$ and $k \in \Lambda$. According to the operation of the EA algorithm, we have $u_i(r_i^*) = u_k(u_k^{-1}(u_\Sigma(r_{\text{total}}; \Omega))) = u_\Sigma(r_{\text{total}}; \Omega)$, where $u_k^{-1} \times (u_\Sigma(r_{\text{total}}; \Omega)) < 0$. Since the marginal utility function is decreasing, it holds that $u_k(0) < u_k(u_k^{-1}(u_\Sigma(r_{\text{total}}; \Omega))) = u_i(r_i^*)$. It is not hard to see that, if we reduce the user set from Ω to $\Omega - \Lambda$ and perform the EA algorithm with total resource $\sum_{i \in \Omega - \Lambda} r_i^*$, then the aggregate marginal utility is also equal to $u_\Sigma(r_{\text{total}}; \Omega)$. Therefore, we have $r_i^* = u_i^{-1}(u_\Sigma(\sum_{i \in \Omega - \Lambda} r_i^*; \Omega - \Lambda))$ or $u_i(r_i^*) = u_\Sigma(\sum_{i \in \Omega - \Lambda} r_i^*; \Omega - \Lambda)$ for all $i \in \Omega - \Lambda$. The fact that $u_i^{-1}(x)$ is decreasing for all $i \in \Omega$ implies $\sum_{i \in \Omega - \Lambda} u_i^{-1}(x)$ is decreasing, which in turn implies $u_\Sigma(r; \Omega - \Lambda)$, the inverse of $\sum_{i \in \Omega - \Lambda} u_i^{-1}(x)$ is decreasing. Therefore, we have $u_i(s_i) = u_\Sigma(r_{\text{total}}; \Omega - \Lambda) > u_\Sigma(\sum_{i \in \Omega - \Lambda} r_i^*; \Omega - \Lambda) = u_i(r_i^*)$, because $\sum_{i \in \Omega - \Lambda} r_i^* > r_{\text{total}}$, according to Lemma 1. ■

Proof of Theorem 3: Let $\pi_{\text{MEA}} = \{r_i | i \in \Omega\}$ be the allocation determined by the MEA algorithm. We need to prove that $\pi_{\text{MEA}}(\Omega)$ is marginally fair and satisfies $\sum_{i \in \Omega} r_i = r_{\text{total}}$. Assume that the MEA algorithm is executed for N iterations to obtain the solution π_{MEA}. Let Γ_N be the user set in the Nth iteration. Since the MEA algorithm ends at the Nth iteration, it holds that $r_i \geq 0$ for all $i \in \Gamma_N$ and $\sum_{i \in \Omega} r_i = r_{\text{total}}$. Moreover, it is true that $r_i = u_i^{-1}(u_\Sigma(r_{\text{total}}; \Gamma_N))$ and $u_i(r_i) = u_\Sigma(r_{\text{total}}; \Gamma_N)$ for all $i \in \Gamma_N$. Note that it is possible to have $r_i = 0$ for some $i \in \Gamma_N$. For any $k \in \Omega - \Gamma_N$, we have $u_\Sigma(r_{\text{total}}; \Gamma_N) > u_k(0)$, according to Lemma 2. Therefore, $\pi_{\text{MEA}} = \{r_i | i \in \Omega\}$ is marginally fair and satisfies $\sum_{i \in \Omega} r_i = r_{\text{total}}$ and, thus, is optimal. This completes the proof of Theorem 3. ■

Proof of Theorem 4: Let $\pi = \{r_i | i \in \Omega\}$ be an allocation that is generalized marginally fair and satisfies $\sum_{i \in \Omega} r_i = r_{\text{total}}$. We will prove π is the only optimal solution by showing that the utility sum obtained by π is greater than the utility sum obtained by any feasible allocation different from π.

Let $\Gamma = \{i | i \in \Omega, Q_i > 0\}$, $\Gamma_Z = \{i | r_i = 0, i \in \Gamma\}$, $\Gamma_P = \{i | 0 < r_i < (Q_i/c_i), i \in \Gamma\}$, and $\Gamma_A = \{i | r_i = (Q_i/c_i), i \in \Gamma\}$. Consider any feasible allocation $\pi' = \{r'_i | i \in \Omega\}$ that is different from π. It is clear that, if $r'_i > 0$ for user i with $Q_i = 0$, then the utility sum can be improved by shifting the resource allocated to user i to user j with $r'_j < (Q_j/c_j)$. Similarly, if $\sum_{i \in \Omega} r'_i < r_{\text{total}}$, then the utility sum can be improved by allocating the remaining resource to user k with $r'_k < (Q_k/c_k)$. Therefore, we assume that $r'_i = 0$ if $Q_i = 0$ and $\sum_{i \in \Omega} r'_i = r_{\text{total}}$. Let E and F be two sets such that user i is in set E or set F iff $r_i > r'_i$ or $r_i < r'_i$, respectively. We have $E \subseteq \Gamma_P \cup \Gamma_A$, $F \subseteq \Gamma_P \cup \Gamma_Z$, and $\sum_{i \in E}(r_i - r'_i) = \sum_{i \in F}(r'_i - r_i)$. According to (3), for any user $i \in E$ and $j \in F$, it holds that $u_i(r_i) \geq u_j(r_j)$. Let $u_{E,\min} = \min_{i \in E} u_i(r_i)$. We have $\sum_{i \in \Omega} U_i(r_i) - \sum_{i \in \Omega} U_i(r'_i) = \sum_{i \in E} U_i(r_i) - \sum_{i \in E} U_i(r'_i) - (\sum_{j \in F} U_j(r'_j) - \sum_{j \in F} U_j(r_j)) = \sum_{i \in E} \int_{r'_i}^{r_i} u_i(r) dr - \sum_{j \in F} \int_{r_j}^{r'_j} u_j(r)\, dr > \sum_{i \in E} u_i(r_i) \cdot (r_i - r'_i) - \sum_{j \in F} u_j(r_j) \cdot (r'_j - r_j) \geq u_{E,\min} (\sum_{i \in E}(r_i - r'_i) - \sum_{j \in F}(r'_j - r_j)) = 0$. Note that the first inequality in the preceding derivation is true because $u_i(x)$ is a decreasing function of x, and both sets E and F are nonempty. As a result, π, if it exists, is the unique optimal resource allocation.

Conversely, let $\pi = \{r_i | i \in \Omega\}$ be an optimal allocation. Note that optimal solution exists because there are obviously feasible solutions. Let $\Gamma = \{i | i \in \Omega, Q_i > 0\}$, $\Gamma_Z = \{i | r_i = 0, i \in \Gamma\}$, $\Gamma_P = \{i | 0 < r_i < (Q_i/c_i), i \in \Gamma\}$, and $\Gamma_A = \{i | r_i = (Q_i/c_i), i \in \Gamma\}$. Since $\sum_{i \in \Omega}(Q_i/c_i) > r_{\text{total}}$, it must hold that $\sum_{i \in \Omega} r_i = r_{\text{total}}$ and $r_i = 0$ if $Q_i = 0$. Assume that $\Gamma_P = \emptyset$. In other words, there exists $\Psi \subset \Omega$ such that $\Psi \neq \Omega$, $r_i = (Q_i/c_i)$ for all $i \in \Psi$, and $\sum_{i \in \Psi} r_i = r_{\text{total}}$. It is true that $u_i(r_i) \leq u_j(r_j)$ for any users $i \in \Omega - \Psi$ and $j \in \Psi$. Otherwise, if $u_i(r_i) > u_j(r_j)$ for some $i \in \Omega - \Psi$ and $j \in \Psi$, then the utility sum of $\pi' = \{r'_k | k \in \Omega\}$ with $r'_i = r_i + \delta$, $r'_j = r_j - \delta$, $0 < \delta < \min((Q_i/c_i), u_i^{-1}(u_\Sigma(r_j; \{i, j\})), r_j)$, and $r'_k = r_k$ for all $k \in \Omega - \{i, j\}$, is greater than that of π, which is a contradiction to the assumption that π is an optimal allocation. (Note that $\delta = u_i^{-1}(u_\Sigma(r_j; \{i, j\}))$ makes $u_i(r'_i) = u_j(r'_j)$.) Therefore, $\pi = \{r_i | i \in \Omega\}$ is generalized marginally fair if $\Gamma_P = \emptyset$.

Consider the case $\Gamma_P \neq \emptyset$. We claim that $u_i(r_i) = u_j(r_j)$ for all $i, j \in \Gamma_P$. Suppose that the claim is false, i.e., there exist $i, j \in \Gamma_P$ such that $u_i(r_i) > u_j(r_j)$. In this case, the utility sum of $\pi' = \{r'_k | k \in \Omega\}$ with $r'_i = r_i + \delta$, $r'_j = r_j - \delta$, $0 < \delta < \min((Q_i/c_i) - r_i, u_i^{-1}(u_\Sigma(r_i + r_j; \{i, j\})) - r_i, r_j)$, and $r'_k = r_k$ for all $k \in \Omega - \{i, j\}$ is greater than that of π, which is a contradiction to the optimality assumption of π. Therefore, the claim is true. Assume that $k \in \Gamma_Z$. It must hold that $u_k(0) \leq u_i(r_i)$ for all $i \in \Gamma_P$. Otherwise, if $u_k(0) > u_i(r_i)$ for some $i \in \Gamma_P$, then the utility sum of $\pi' = \{r'_j | j \in \Omega\}$ with $r'_k = r_k + \delta$, $r'_i = r_i - \delta$, $0 < \delta < \min((Q_k/c_k), u_k^{-1}(u_\Sigma(r_k + r_i; \{i, k\})), r_i)$ and $r_j = r'_j$ for all $j \in \Omega - \{i, k\}$ is greater than that of π, which is a contradiction to the optimality assumption of π again. Finally, assume that $k \in \Gamma_A$. We have $u_k(r_k) \geq u_i(r_i)$ for all $i \in \Gamma_P$. Otherwise, if $u_k(r_k) < u_i(r_i)$ for some $i \in \Gamma_P$, then the utility sum of $\pi' = \{r'_j | j \in \Omega\}$ with $r'_i = r_i + \delta$, $r'_k = r_k - \delta$, $0 < \delta < \min((Q_i/c_i) - r_i, u_i^{-1}(u_\Sigma(r_i + r_k; \{i, k\})) - r_i, r_k)$, and $r'_j = r_j$ for all $j \in \Omega - \{i, k\}$ is greater than that of π, which is a contradiction once again. Therefore, we conclude that $\pi = \{r_i | i \in \Omega\}$ is generalized marginally fair. This completes the proof of Theorem 4. ■

Proof of Theorem 5: Let $\pi(\Omega) = \{x_i | i \in \Omega\}$ be an optimal resource-block-based allocation. Since the utility function is increasing, it must hold that $\sum_{i \in \Omega} x_i = x_{\text{total}}$. Assume that $\pi(\Omega)$ is not (resource-block-based) marginally fair, meaning that there exists i and j such that $x_i > 0$ and $\Delta U_i[x_i] < \Delta U_i[x_j + 1]$. As a result, the allocation $\pi'(\Omega) = \{x'_i | i \in \Omega\}$, where $x'_i = x_i - 1$, $x'_j = x_j + 1$, and $x'_k = x_k$ if $k \neq i$ or j gives larger total utility than $\pi(\Omega)$, which is a contradiction. Therefore, $\pi(\Omega)$ must be marginally fair. Conversely, assume that $\pi(\Omega) = \{x_i | i \in \Omega\}$ is marginally fair with $\sum_{i \in \Omega} x_i = x_{\text{total}}$. Let $\alpha = \min\{\Delta U_i[x_i] | x_i > 0, i \in \Omega\}$ and $\beta = \max\{\Delta U_j[x_j + 1] | j \in \Omega\}$. According to the definition of marginal fairness, we have $\alpha \geq \beta$. Let $\pi'(\Omega) = \{x'_i | i \in \Omega\}$ be any other feasible allocation. Define two sets E and F such that $E = \{i | x_i > x'_i, i \in \Omega\}$ and $F = \{i | x_i < x'_i, i \in \Omega\}$. We have $\sum_{i \in E}(x_i - x'_i) = \sum_{i \in F}(x'_i - x_i)$. Let $d = \sum_{i \in E}(x_i - x'_i)$. It is true that $\sum_{i \in \Omega} U_i(x_i \cdot B) - \sum_{i \in \Omega} U_i(x'_i \cdot B) = \sum_{i \in E} \Delta U_i[x_i] - \sum_{i \in F} \Delta U_i[x'_i] \geq d(\alpha - \beta) \geq 0$. Therefore, $\pi(\Omega)$ is optimal. This completes the proof of Theorem 5. ■

TABLE VII
TIME COMPLEXITIES OF THE PROPOSED RESOURCE ALLOCATION ALGORITHMS

	Fluid-flow based		Resource-block based		
	MEA	GEA	SA	RBEA/GRBEA	MEA+SA/GEA+SA
Time complexity	$O(\|\Omega\|^2)$	$O(\|\Omega\|^2)$	$O(x_{total} \cdot \|\Omega\|)$	$O(x_{total} + \|\Omega\|^2)$	$O(\|\Omega\|^2)$

APPENDIX B
ANALYSIS OF TIME COMPLEXITIES OF THE PROPOSED ALGORITHMS

As analyzed in [3], the time complexity of the EA algorithm is $O(|\Omega|)$. Consequently, the time complexity of our proposed MEA algorithm is $O(|\Omega|^2)$ because, in the worst case, users are removed one by one until one is left to be allocated with the total resource.

For the GEA algorithm, the maximum number of iterations needed is $2 \cdot |\Omega| - 1$, which happens when all users are moved (one by one) from Γ_Z to Γ_P, and $|\Omega| - 1$ users are further moved from Γ_P to Γ_A. The computational complexity to move a user from Γ_Z to Γ_P is $O(|\Omega|)$, while that to move a user from Γ_P to Γ_A is $O(|\Omega| - i)$ if $|\Gamma_P| = |\Omega| - i$. Therefore, the time complexity of GEA is $O(|\Omega|^2)$.

Consider the three algorithms designed for resource-block-based systems with constantly backlogged queues, namely, SA, RBEA, and MEA+SA. Recall that, in SA, one resource block is allocated in each iteration. As a result, the time complexity of SA is $O(x_{\text{total}} \cdot |\Omega|)$ because the time complexity of each iteration is $O(|\Omega|)$. For the RBEA algorithm, define S_i as its state when $|\Gamma| = i$. In the worst case, we have $\Gamma = \Omega$ initially, and the RBEA algorithm visits all states S_i, $i = |\Omega|, |\Omega| - 1, \ldots, 1$. Moreover, when resource blocks are to be allocated to some user, only one is allocated. In other words, if resource blocks are allocated in state S_i, then exactly i resource blocks are allocated. The computational complexity to allocate i resource blocks in state S_i is $O(i)$. Since there are x_{total} resource blocks, the computational complexity to allocate all the resource blocks to users is $O(x_{\text{total}})$. According to the RBEA algorithm, the computational complexity to switch from state S_i to state S_{i-1} is $O(i)$. Therefore, the time complexity of RBEA is $O(x_{\text{total}} + |\Omega|^2)$. For the MEA+SA algorithm, let $\pi_{\text{MEA}}(\Omega) = \{r_i | i \in \Omega\}$ denote the solution obtained by the MEA algorithm and $\{x'_i | i \in \Omega\}$ represent the preliminary solution, where $x'_i = \lfloor (r_i/B) \rfloor$ for all $i \in \Omega$. Recall that the time complexity of the MEA algorithm is $O(|\Omega|^2)$. Since $\sum_{i\in\Omega} r_i = r_{\text{total}}$, we have $x_{\text{total}} - \sum_{i\in\Omega} x'_i \leq |\Omega| - 1$. As a consequence, there are at most $|\Omega| - 1$ resource blocks to be allocated by the SA algorithm, which incurs a time complexity of $O(|\Omega|^2)$. Therefore, the time complexity of the MEA+SA algorithm is $O(|\Omega|^2)$.

Finally, consider GRBEA and MEA+SA, which are the two algorithms designed for resource-block-based systems with generally backlogged queues. For GRBEA, the maximum number of iterations needed is the same as that required by RBEA. As a consequence, the time complexity of GRBEA is the same as that of RBEA. Similarly, the time complexity of GEA+SA is the same as that of MEA+SA. Table VII summarizes the time complexities of the proposed algorithms.

REFERENCES

[1] D. Angelini and M. Zorzi, "On the throughput and fairness performance of heterogeneous downlink packet traffic in a locally centralized CDMA/TDD system," in *Proc. IEEE VTC-Fall*, 2002, pp. 510–514.

[2] W. H. Kuo and W. Liao, "Utility-based radio resource allocation for QoS traffic in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 7, no. 7, pp. 2714–2722, Jul. 2008.

[3] W. H. Kuo and W. Liao, "Utility-based resource allocation in wireless networks," *IEEE Trans. Wireless Commun.*, vol. 6, no. 10, pp. 3600–3606, Oct. 2007.

[4] G. Song and Y. Li, "Cross layer optimization for OFDM wireless networks—Part I: Theoretical frame work," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 614–624, Mar. 2005.

[5] G. Song and Y. Li, "Cross layer optimization for OFDM wireless networks—Part II: Algorithm development," *IEEE Trans. Wireless Commun.*, vol. 4, no. 2, pp. 625–634, Mar. 2005.

[6] Z. Jiang, Y. Ge, and Y. Li, "Max-utility wireless resource management for best-effort traffic," *IEEE Trans. Wireless Commun.*, vol. 4, no. 1, pp. 100–111, Jan. 2005.

[7] L. Kleinrock, *Queueing Systems Vol 2: Computer Applications*. New York: Wiley, 1975.

[8] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks—The single node case," *IEEE/ACM Trans. Netw.*, vol. 1, no. 3, pp. 344–357, Jun. 1993.

[9] A. Demers, S. Kesha, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," *J. Internetw. Res. Exper.*, vol. 1, no. 1, pp. 3–26, Sep. 1990.

[10] IEEE Standard for Local and Metropolitan area Networks—Part 16: Air Interface for Fixed Broadband Wireless Access Systems, IEEE Std. 802.16-2009, May 2009.

[11] E. Dahlman, S. Parkvall, J. Sköld, and P. Beming, *3G HSPA and LTE for Mobile Broadband*. New York: Academic, 2007.

[12] M. Kaneko, P. Popovski, and J. Dahl, "Proportional fairness in multi-carrier system with multi-slot frames: Upper bound and user multiplexing algorithms," *IEEE Trans. Wireless Commun.*, vol. 7, no. 1, pp. 22–26, Jan. 2008.

[13] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, "Rate control in communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res. Soc.*, vol. 49, pp. 237–252, Apr. 1998.

[14] H. Kim and Y. Han, "A proportional fair scheduling for multicarrier transmission systems," *IEEE Commun. Lett.*, vol. 9, no. 3, pp. 210–212, Mar. 2005.

[15] X. Zhu, J. Huo, C. Xu, and W. Ding, "QoS-guaranteed scheduling and resource allocation algorithm for IEEE 802.16 OFDMA system," in *Proc. IEEE ICC*, May 2008, pp. 3463–3468.

Tsern-Huei Lee (S'86–M'87–SM'98) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1981, the M.S. degree in electrical engineering from the University of California, Santa Barbara, in 1983, and the Ph.D. degree in electrical engineering from the University of Southern California, Los Angeles, in 1987.

Since 1987, he has been a member of the faculty of the National Chiao Tung University, Hsinchu, Taiwan, where he is a Professor with the Department of Electrical Engineering. Over the past few years, he served as a Consultant to various companies to develop large-scale quality-of-service-enabled frame-based switches/routers, integrated access devices, and unified threat management Internet appliances. His current research interests are in communication protocols, broadband switching systems, traffic management, wireless communications, and network security.

Dr. Lee received an Outstanding Paper Award from the Institute of Chinese Engineers in 1991.

Yu-Wen Huang (S'07) was born in Gangshan District, Kaohsiung City, Taiwan, in 1982. He received the B.S. and M.S. degrees in communication engineering in 2004 and 2006, respectively, from the National Chiao Tung University, Hsinchu, Taiwan, where he currently working toward the Ph.D. degree.

His current research interests include resource allocation, power management, and communication protocols in wireless networks.

Chien-Nan Chen received the B.S. degree in communication engineering from the National Central University, Taoyuan, Taiwan, in 2009 and the M.S. degree in communication engineering in Chiao Tung University, Hsinchu, Taiwan, in 2011.

He is currently serving the army after completing graduate school. His research interest includes the resource allocation in Long-Term Evolution and WiMAX networks.