Proportional Fair RAT Aggregation in HetNets

Ehsan Aryafar
Portland State University, USA

Alireza Keshavarz-Haddad
Shiraz University, Iran

Carlee Joe-Wong
Carnegie Mellon University, USA

***Abstract*—Heterogeneity in wireless network architectures (*i.e.*, the coexistence of 3G, LTE, 5G, WiFi, *etc.*) has become a key component of current and future generation cellular networks. Simultaneous aggregation of each client's traffic across multiple such radio access technologies (RATs) / base stations (BSs) can significantly increase the system throughput, and has become an important feature of cellular standards on multi-RAT integration. Distributed algorithms that can realize the full potential of this aggregation are thus of great importance to operators. In this paper, we study the problem of resource allocation for multi-RAT traffic aggregation in HetNets (heterogeneous networks). Our goal is to ensure that the resources at each BS are allocated so that the aggregate throughput achieved by each client across its RATs satisfies a proportional fairness (PF) criterion. In particular, we provide a simple distributed algorithm for resource allocation at each BS that extends the PF allocation algorithm for a single BS. Despite its simplicity and lack of coordination across the BSs, we show that our algorithm converges to the desired PF solution and provide (tight) bounds on its convergence speed. We also study the characteristics of the optimal solution and use its properties to prove the optimality of our algorithm's outcomes.**

I. Introduction

The increasing demand for wireless data has led to denser and more heterogeneous wireless network deployments. This heterogeneity manifests itself in terms of network deployments across multiple radio access technologies (*e.g.*, 3G, LTE, WiFi, 5G), cell sizes (*e.g.*, macro, pico, femto), and frequency bands (*e.g.*, TV bands, 1.8-2.4 GHz, mmWave), *etc.* To realize the gains associated with such heterogeneous networks (HetNets), consumer (client) devices are also being equipped with an increasing number of radio access technologies (RATs), and some are already able to simultaneously aggregate the traffic across multiple RATs to increase throughput [1].

To support such traffic aggregation on the network side, the 3GPP (3rd generation partnership project) has been actively developing multi-RAT integration solutions. The introduction of LWA (LTE-WiFi Aggregation) as part of the 3GPP Release 13 [2] was a step in this direction. LWA allows using both LTE and WiFi links for a single traffic flow and is generally more efficient than transport layer aggregation protocols (*e.g.*, MultiPath TCP), due to coordination at lower protocol stack layers. LWA's design primarily follows the LTE Dual Connectivity (DC) architecture (defined in 3GPP Release 12 [3]), which allows a wireless device to connect to two LTE eNBs that are on different carrier frequencies, and utilize the radio resources that belong to both of them. Currently, the 3GPP is working on a solution to support below IP (layer 2) multi-RAT integration across *any* combination of RATs, including LTE, WiFi, 802.11ad/ay, and 5G New Radio (NR) [4]. The proposed architecture would allow for dynamic traffic splitting across RATs for each client, which can lead to a significant increase in the system performance (*e.g.*, total throughput).

However, it is difficult to design resource allocation algorithms for each BS[1] that realize the performance benefits of such integrated HetNets. Specifically, **(i)** backhaul links from different BSs in HetNets show diverse capacity and latency characteristics and depend on the underlying backhauling technology. For example, cable and DSL have on average 28 and 62 ms roundtrip latencies, respectively [5]. The latency can be even higher when a network operator uses a third party ISP to communicate with its BSs (*e.g.*, a mobile operator that uses a wired ISP to control its WiFi BSs). Such latencies make it infeasible for BSs to communicate with each other or a central controller for real-time resource allocation at each BS. *As a result, any practical resource allocation algorithm for multi-RAT HetNets should be fully distributed* (*i.e.*, autonomously executed by each BS). **(ii)** Resource allocation has many practical constraints. Conventional BS hardware allows only minor modifications to existing resource allocation algorithms through software updates, limiting the algorithm design space. New algorithms should also incur minimal signaling overhead and computational complexity. Distributed algorithms based on the traditional network utility maximization (NUM) framework [6], [7] do not meet these requirements, because as we will show later through simulations the resulting algorithms are radically different from how conventional BSs operate, have significant over-the-air signaling overhead, and increase the computational complexity on the client side.

In this paper, we study the problem of resource allocation for traffic aggregation in multi-RAT HetNets. We focus on the *proportional-fair (PF)* fairness objective as it is widely used and implemented in BSs and provides a balance between fairness and throughput [8], [9]. We first consider PF resource allocation in a single BS, and then use our insights from this case to design a distributed algorithm that meets our two key research challenges. We next show that our algorithm converges to an optimal PF resource allocation. The key contributions are as follows:

- **Algorithm Design:** We study the basics of PF resource allocation in a single BS to gain intuition for the distributed algorithm design. We show that PF resource allocation in a single BS can be viewed as a special type of water-filling. We generalize this observation to a new fully distributed water-filling algorithm (named AFRA) that makes a minor

[1] **We use "BS" generically to mean an LTE eNB, WiFi AP, etc.**

modification to the conventional single BS algorithm and achieves PF in HetNets.

- **Convergence and Speed:** We show that AFRA is guaranteed to converge to an equilibrium as BSs autonomously execute it [Theorem 1] and derive tight bounds on its convergence time (speed) [Theorem 2].
- **Optimality:** We first show that at optimality, the sum of the inverse water-fill levels across all BSs is equal to the sum of the weights (numbers that show clients' priorities) across all clients [Theorem 3]. Next, we use this property to prove that any equilibrium outcome of AFRA is *globally optimal* [Theorem 4]. Finally, we show that at equilibrium the vector of throughput rates across all clients is unique; however, there could be infinitely many resource allocations that realize this outcome [Theorem 5].
- **Practicality:** We construct a testbed with programmable BS hardware, and show that we can successfully aggregate the throughput across multiple BSs at the MAC layer. We also show that replacing the conventional resource allocation algorithm on each BS with AFRA can substantially increase the system throughput and fairness.
- **Performance:** We conduct extensive simulations to characterize AFRA's convergence time properties as we scale the number of BSs and clients. We also introduce policies that reduce the convergence time by more than 30%. Finally, we compare the performance of AFRA against DDNUM, a dual decomposition algorithm that we derived from the NUM framework. We show that compared to DDNUM, AFRA is 2-3 times faster with 4-5 times less over-the-air overhead.

This paper is organized as follows. We discuss the related work in Section II. We present the system model and details of AFRA in Section III. In Sections IV and V we prove the convergence and optimality of AFRA. We present the results of our experiments, simulations, and comparisons against DDNUM in Section VI. We conclude the paper in Section VII.

II. Related Work

We discuss the related work in the areas of multi-RAT communication and distributed optimization, and highlight their differences from this paper.

Multi-RAT Communication. Resource allocation algorithms that realize the capacity gains in HetNets are still in their early stages. The problem of PF resource allocation for LWA was studied in [10]. In the proposed setup, each client has one LTE and one WiFi RAT. Further, there is only a single LTE BS in the entire network, and each client's throughput across its WiFi RAT is *fixed*. Next, the authors propose a water-filling based resource allocation algorithm at the LTE BS that achieves PF. Similarly, we show that the optimal PF resource allocation in a single BS can be interpreted as a form of water-filling. However, we use the observation to design an optimal algorithm for the generic problem with any number of BSs and client RATs, and explicitly model the impact of system dynamics on the throughput that each client gets from every BS. In our prior work [11], we addressed the problem of max-min fair resource allocation in HetNets. However, even with *opportunistic centralized* network supervision over autonomous resource allocation at each BS we could not optimally solve the problem. Here, we focus on the PF objective, which is commonly implemented in BSs, and show that we can optimally solve the problem in a purely distributed manner. Other works have built testbeds to evaluate the over-the-air performance of MAC-level cross-RAT throughput aggregation [12]–[14]. All these works have relied on conventional scheduling algorithms on each BS and focused on higher layer transport and application performance. We experimentally show that replacing the conventional resource allocation algorithms with AFRA can substantially increase the system throughput and fairness.

Distributed Network Utility Maximization. There is a large body of general results on the mathematics of distributed computation, some of which are summarized in standard textbooks (*e.g.* [15]). More recently, the framework of NUM [6], [7] has emerged as a mathematical tool to optimize layered network architectures. The framework allows for decomposition of a global optimization problem into subsets of local problems that are carried out distributedly and implicitly solve the global NUM problem. We have derived an alternative distributed algorithm (named DDNUM) by leveraging dual decomposition and the NUM framework. We will show through simulations that DDNUM is 2-3 times slower than AFRA (in terms of convergence time) and increases the wireless signaling overhead by 4-5 times. These disadvantages, coupled with the increased client side computational complexity and lack of compatibility with conventional BSs, make NUM-based algorithms impractical for multi-RAT traffic aggregation.

III. System Model

We discuss the system model and the resource allocation algorithm that is autonomously executed by each BS.

A. Network Model

We consider a HetNet composed of a set of BSs $\mathbf{M} = \{1, ..., M\}$ and a set of clients $\mathbf{N} = \{1, ..., N\}$. Each BS has a limited transmission range and can only serve clients within its range. Each client has a client-specific number of RATs, and therefore has access to a subset of BSs. We model clients that can aggregate traffic across BSs of the same technology (*e.g.*, LTE DC) with multiple such RATs. Fig. 1 shows an example HetNet topology with 4 BSs and 2 clients. We assume that clients split their traffic over the BSs and focus on the resource allocation problem at each BS. It is itself a challenging problem to determine which BS to associate with among same technology BSs (*e.g.*, choosing the optimal LTE BS if a client has an LTE RAT). We assume there exists a rule to pre-determine client RAT to BS association. The pre-determination rule could be for instance any load balancing algorithm [16], [17], or based on the received signal strength. Similar to [11], [16], [18], [19], we assume that the transmission in one BS does not interfere with an adjacent BS. This can be achieved through spectrum separation between

BSs that belong to different access networks and frequency reuse among same technology BSs.

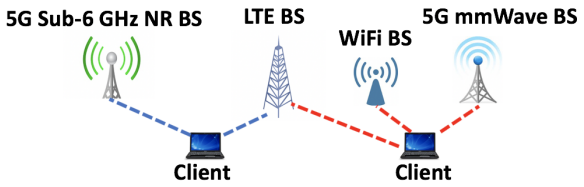


Fig. 1. **A heterogeneous network with 4 access technologies. Each client is in the coverage area of a group of BSs (dotted lines) and can split or aggregate its traffic across the corresponding BSs (RATs).**

B. Throughput Model

We consider a multi-rate system and use $R_{i,j}$ to denote the PHY (physical layer) rate of client i to BS j. Since each BS generally serves more than one client, clients of the same BS need to share resources such as time and frequency slots (*e.g.*, in 3/4/5G) or transmission opportunities (*e.g.*, in WiFi). The throughput achieved by client i from BS j thus depends on the load of the BS and will be a fraction of $R_{i,j}$. We assume that each BS employs a TDMA throughput sharing model[2] and let $\lambda_{i,j}$ denote the fraction of time allocated to client i by BS j. Hence, the throughput achieved by client i from BS j is equal to $\lambda_{i,j}R_{i,j}$ and its total throughput across all its RATs would be

$$\text{Total Throughput of Client } i = r_i = \sum_{j=1}^{M} \lambda_{i,j} R_{i,j} \qquad (1)$$

The total amount of time fractions available to each BS cannot exceed 1. Thus, for the $\lambda_{i,j}s$ to be feasible we have

$$\sum_{i=1}^{N} \lambda_{i,j} \leq 1 \quad \forall j \in \mathbf{M} \qquad (2)$$

$$\lambda_{i,j} \geq 0 \quad \forall i \in \mathbf{N},\ j \in \mathbf{M} \qquad (3)$$

TABLE I
MAIN NOTATION

$\mathbf{N}$ and N: Set and number of all clients in the network
$\mathbf{M}$ and M: Set and number of all BSs in the network
$R_{i,j}$: PHY (physical layer) rate of client i to BS j
R_{max}: maximum PHY rate across all clients and BSs
R_{min}: non-zero minimum PHY rate across all clients and BSs
$\lambda_{i,j}$: Fraction of time allocated to client i by BS j
$\boldsymbol{\lambda}$: Vector of $\lambda_{i,j}$s across all clients and BSs
r_i: Total throughput of client i across all its RATs
ω_i: A positive number that represents client i's weight or priority
θ_j: Water-fill level at BS j

C. Background: Conventional PF Allocation in a Single BS

We first describe the basics of the PF resource allocation that is conventionally implemented in today's BSs. **Consider a network topology consisting of only a single BS j and n' clients.** Let r_i denote the throughput of client i and ω_i a positive number that denotes its weight (or priority). A widely used objective function for PF is to maximize $\sum_{i=1}^{n'} \omega_i \log(r_i)$ [8], [9]. It represents a tradeoff between throughput and fairness among the clients. Let λ_i denote the time fraction allocated to client i by BS j. To maximize the PF objective function, the BS needs to solve the following problem

$$\begin{aligned} \mathcal{P}_1: \quad & \max \sum_{i=1}^{n'} \omega_i \log(R_{i,j}\lambda_i) \\ s.t. \quad & \sum_{i=1}^{n'} \lambda_i \leq 1 \\ \text{variables:} \quad & \lambda_i \geq 0 \end{aligned}$$

Problem $\mathcal{P}_1$ can be easily solved through a simple algorithm. The Lagrangian of $\mathcal{P}_1$ can be expressed as

$$L(\boldsymbol{\lambda}, \mu) = \sum_{i=1}^{n'} \omega_i \log(R_{i,j}\lambda_i) + \mu(1 - \sum_{i=1}^{n'} \lambda_i) \qquad (4)$$

where μ is a constant number (Lagrange multiplier) chosen to meet the time resource constraint. Differentiating with respect to time fraction resource λ_i and setting to zero gives

$$\frac{R_{i,j}\omega_i}{R_{i,j}\lambda_i} - \mu = 0 \implies \frac{\omega_i}{\lambda_i} = \mu \quad \forall i \in \{1, ..., n'\} \qquad (5)$$

Since the sum of time fractions at optimality is equal to 1, we can conclude from Eq. (5) that $\mu = \sum \omega_i$. With known μ and ω_i, we can derive λ_is from Eq. (5).

Now, let θ_j be defined as $\frac{1}{\mu}$. Leveraging Eq. (5), we have

$$\frac{\lambda_i}{\omega_i} = \theta_j \ \forall i \in \{1, ..., n'\} \implies \boxed{\frac{r_i}{\omega_i R_{i,j}} = \theta_j \ \forall i \in \{1, ..., n'\}} \qquad (6)$$

Eq. (6) has an interesting water-filling based interpretation: *the time allocated to each client is such that the throughput of the client divided by its PHY rate times its weight is the same across all clients*. We refer to this ratio (*i.e.*, θ_j) as the water-fill level of BS j. In the next section, we will turn this observation in a single BS into a distributed resource allocation algorithm in HetNets.

D. Distributed Resource Allocation in HetNets

There are two approaches to designing a distributed resource allocation algorithm for generic HetNets. One approach, as we show in [20], is to extend the formulation in $\mathcal{P}_1$ to include multiple BSs and client RATs, and use dual decomposition to derive a distributed algorithm. This approach converges to the optimal solution; however, the Lagrange multipliers across BSs would no longer correspond to BSs' water-fill levels [20]. The second approach is to directly generalize the water-filling interpretation to derive an alternative algorithm, which still converges to the optimal solution (Section V) with far less overhead, convergence time, and complexity than the dual decomposition based algorithm (Section VI-C).

[2]In Section VI-A, we discuss how we can extend our model and algorithm to capture practical implementation issues such as WiFi contention.

From Eq. (6), we observe that in a network with only a single BS, the BS allocates its time resources so that the clients who get the time resources reach the same water-fill level (*i.e.*, throughput divided by $\omega_i R_{i,j}$). Thus, in generic HetNets, if each BS considers the *total throughput of each client across all its RATs* (*i.e.*, r_i) divided by $\omega_i R_{i,j}$ in its water-fill definition, this should lead to a fair distributed algorithm. In other words, each BS j should share its time resources across its clients such that: *(1) all clients who get the time resources reach the same water-fill level at BS j (i.e., θ_j), and (2) if a client (e.g., i') does not get any time resources from BS j, its $\frac{r_{i'}}{\omega_{i'} R_{i',j}}$ is greater than or equal to θ_j*. Fig. 2 illustrates this water-filling operation, which is executed by BS j.

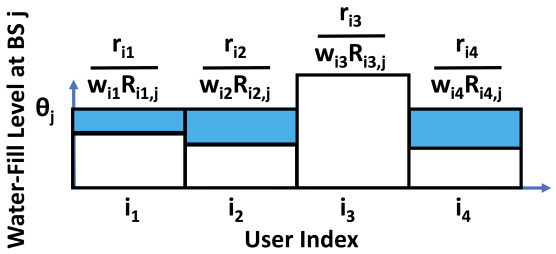


Fig. 2. **There are 4 clients with non-zero PHY rates to BS j. Blue boxes denote contributions to $\frac{r_i}{\omega_i R_{i,j}}$ by BS j (when it allocates time resources) and white boxes show contributions to it by other BSs. BS j allocates its time resources so that all clients that get resources achieve the same water-fill level (θ_j). Clients that do not get any resources from BS j have a higher $\frac{r_i}{\omega_i R_{i,j}}$ than θ_j. Client i_3 is one such client in this example.**

We next turn this idea into a distributed resource allocation algorithm. Consider slotted time for now. Algorithm AFRA (Fig. 3) summarizes the steps that are autonomously executed by each BS j. There are three main steps in the algorithm: (i) clients are sorted based on the total throughout they receive from other BSs (r'_i) divided by $\omega_i R_{i,j}$ (Line 3), (ii) BS j finds the water-fill level (θ_j) and allocates the time resources accordingly (Line 4), and (iii) finally we introduce a randomization parameter to limit concurrent resource adaptation of a single client by multiple BSs (Line 5).

We next elaborate on how each BS j finds its water-fill level and its clients' time resource fractions (*i.e.*, Line 4). Let n' denote the number of clients such that $R_{i,j} > 0$. Let r'_i denote the total throughput of client i from all BSs other than j. Consider an ordering in clients' $\frac{r'_i}{\omega_i R_{i,j}}$ according to Line 3 of AFRA. In order to solve the water-fill problem (*i.e.*, Line 4 of AFRA), we need to find the water-fill level θ_j, client index k, and time fractions $\lambda_{i,j}$s such that

$$\frac{r'_1 + \lambda_{1,j} R_{1,j}}{\omega_1 R_{1,j}} = \frac{r'_2 + \lambda_{2,j} R_{2,j}}{\omega_2 R_{2,j}} = \ldots = \frac{r'_k + \lambda_{k,j} R_{k,j}}{\omega_k R_{k,j}} = \theta_j \tag{7}$$

$$\frac{r'_k}{\omega_k R_{k,j}} < \theta_j \leq \frac{r'_{k+1}}{\omega_{k+1} R_{k+1,j}} \tag{8}$$

$$\sum_{i=1}^{k} \lambda_{i,j} = 1, \quad \lambda_{i,j} > 0 \tag{9}$$

Algorithm AFRA: Autonomous Fair Resource Allocation
Input: r_i , $R_{i,j}$, and $\lambda_{i,j}$ $\forall$ client i for which $R_{i,j} > 0$
randomization parameter $p_j \in (0,1)$
Output: Updated $\lambda_{i,j}$
1. Let n' denote the number of clients *s.t.* $R_{i,j} > 0$
2. Find the throughput of each client i provided to it by all other BSs (*i.e.,* r'$_i$)
3. Sort clients based on their throughput from **other** BSs divided by their weight times PHY rate, *i.e.*,
$$\frac{r'_1}{w_1 R_{1,j}} \leq \ldots \leq \frac{r'_k}{w_k R_{k,j}} \leq \frac{r'_{k+1}}{w_{k+1} R_{k+1,j}} \leq \ldots \leq \frac{r'_{n'}}{w_{n'} R_{n',j}}$$
4. Run water-fill, *i.e.*, find k, θ_j , and $\lambda_{i,j}$ *s.t.*
$$\frac{r_1}{w_1 R_{1,j}} = \ldots = \frac{r_k}{w_k R_{k,j}} = \theta_j \quad | \ \lambda_{i,j} > 0 \ , \sum_{i=1}^{k} \lambda_{i,j} = 1$$
$$\frac{r_{k+1}}{w_{k+1} R_{k+1,j}}, \ldots, \frac{r_{n'}}{w_{n'} R_{n',j}} \geq \theta_j \quad \textit{and} \quad \lambda_{i,j} = 0$$
5. **if** *(rand* < p_j*)* **then** *Update* $\lambda_{i,j}$

Fig. 3. **Resource allocation algorithm autonomously run by each BS j.**

We can find these variables with a simple set of linear operations. First, we can find k by checking a set of inequalities

$$\begin{cases} \frac{\frac{r'_2 \omega_1 R_{1,j}}{\omega_2 R_{2,j}} - r'_1}{R_{1,j}} \geq 1 \Rightarrow k = 1 \quad \textbf{else} \\ \frac{\frac{r'_3 \omega_1 R_{1,j}}{\omega_3 R_{3,j}} - r'_1}{R_{1,j}} + \frac{\frac{r'_3 \omega_2 R_{2,j}}{\omega_3 R_{3,j}} - r'_2}{R_{2,j}} \geq 1 \Rightarrow k = 2 \quad \textbf{else} \\ \ldots \\ \frac{\frac{r'_{n'} \omega_1 R_{1,j}}{\omega_{n'} R_{n',j}} - r'_1}{R_{1,j}} + \ldots + \frac{\frac{r'_{n'} \omega_{n'-1} R_{n'-1,j}}{\omega_{n'} R_{n',j}} - r'_{n'-1}}{R_{n'-1,j}} \geq 1 \\ \qquad\qquad \Rightarrow k = n' - 1 \quad \textbf{else} \\ k = n' \end{cases}$$

In the first inequality, we first check if $\frac{r'_1 + R_{1,j}}{\omega_1 R_{1,j}} \leq \frac{r'_2}{\omega_2 R_{2,j}}$. If this is true, from Eq. (7) we conclude that client 2 would have a higher $\frac{r'_2}{\omega_2 R_{2,j}}$ than $\frac{r_1}{\omega_1 R_{1,j}}$ even if BS j allocated **all** its time resources to client 1 (*i.e.*, to the client with minimum $\frac{r'_i}{\omega_i R_{i,j}}$ across all n' clients). As a result k should be equal to 1. This procedure (and logic) is continued until k is found.

With known k, we can find θ_j by combining Eqs. (7) and (9) and solving the following linear equation

$$\sum_{i=1}^{k} \frac{\theta_j \omega_i R_{i,j} - r'_i}{R_{i,j}} = 1 \tag{10}$$

With known k and θ_j, the $\lambda_{i,j}$s can be found from Eq. (7).

AFRA's Computational Complexity and Message Passing Overhead. We calculate AFRA's computational complexity in finding the new time resource fractions ($\lambda_{i,j}$s) for a BS j. Let n' denote the number of clients with non-zero PHY rates to j. The complexity of sorting clients (Line 3) is $O(n' \log(n'))$. The complexity of finding the water-fill level and the new time resource fractions (Line 4) is $O(n' \log(n'))$ (with a binary search to find k). Thus, the overall computational complexity is $O(n' \log(n'))$. If we assume that each client has

on average K RATs, then on average n' would be equal to $\frac{KN}{M}$. Thus, the computational complexity would also be equal to $O(\frac{KN}{M}\log(\frac{KN}{M}))$.

Each BS uses the *total throughput of each client across all its RATs* in its calculations to find the water-fill level and the new $\lambda_{i,j}$s. Each time a client's time resource (and hence total throughput) is changed, the client needs to inform all BSs to which it is connected about its new total throughput. Thus, the total message passing overhead generated by clients of a single BS is at most equal to $O(n'K)$, or alternatively $O(\frac{K^2N}{M})$.

IV. Convergence and Speed of AFRA

In this section, we investigate the convergence properties of AFRA. We first show that as BSs autonomously execute AFRA, the system converges to an equilibrium. Next, we investigate the convergence time properties of AFRA and provide tight bounds to quantify it.

Before we discuss convergence, we present a formal definition of an equilibrium.

Definition 1 *Equilibrium: The vector of time fractions across all the BSs and clients is an equilibrium outcome if none of the BSs can increase its water-fill level through unilateral change of its time resource allocations.*

Our next theorem guarantees the convergence of AFRA.

Theorem 1 *Let each BS autonomously execute AFRA. Then, the system converges to an equilibrium, i.e., $\forall i \in \mathbf{N}$ and $j \in \mathbf{M}$ $\lambda_{i,j} \to \lambda_{i,j}^{eq}$, $\theta_j \to \theta_j^{eq}$, and $r_i \to r_i^{eq}$.*

Sketch of Proof: Let $\boldsymbol{\lambda}$ denote the vector of time fractions ($\lambda_{i,j}$s) across all clients and BSs, and $f(\boldsymbol{\lambda}) = \sum_{i=1}^{N} \omega_i \log(r_i)$ be the potential function. A potential function [21] is a useful tool to analyze equilibrium properties, as it maps the payoff (*e.g.*, throughput) of all clients into a single function.

Since the number of clients and BSs is finite, f is bounded. The key step to prove convergence, is to show that each time a BS j adjusts its time fractions (*i.e.*, $\lambda_{i,j}$s), the potential function (f) increases. This property coupled with f's boundedness guarantees its convergence. Next, we show in [20] that the change in potential function is proportional to the product of the change in water-fill levels and the change in $\lambda_{i,j}$s. Since f converges (*i.e.*, its variations converge to 0), one or both of these terms should converge to 0. Either of these conditions guarantee the convergence of the $\lambda_{i,j}$s (and hence, θ_js and r_is). Details of the proof are provided in [20]. ■

Next, we discuss the convergence time properties of AFRA. Before we can derive a bound on convergence time, we need to define a discretization factor on the time fractions (*i.e.*, $\lambda_{i,j}$s). This technicality is due to the fact that $\lambda_{i,j}$s in our model are continuous variables, which can cause some BSs to continuously make infinitesimal adjustments to them. These adjustments converge to 0 as time goes to infinity.

In practice, operations always happen in discretized levels. For example, consider the following discretization policy:

Definition 2 *Discretization Policy: During water-fill calculation by a BS j in AFRA, the time fraction allocated to the client with minimum $\frac{r_i}{\omega_i R_{i,j}}$ should increase by at least ϵ. Otherwise, the BS would not update its time fractions.*

Based on the above discretization policy, we can derive the following bound on the convergence time.

Theorem 2 *Consider a HetNet with N clients and M BSs. Then, the number of steps that it takes for AFRA to converge is upper bounded by $O(\frac{NM^2\log(MN)}{\epsilon^2})$.*

Sketch of Proof: Let $f(\boldsymbol{\lambda}) = \sum_{i=1}^{N} \omega_i \log(r_i)$ be the potential function from the proof of Theorem 1. To compute a bound on the convergence time, we study the increments of f. The key step is to find a lower bound on f's increments. Since f increases whenever a BS makes adjustments to its $\lambda_{i,j}$s, the convergence time is then upper bounded by the difference between the maximum and minimum possible values of f divided by the lower bound on f's increments [20]. ■

V. Optimality of AFRA

Beyond convergence, we study the optimality properties of equilibria. We first derive some useful properties of the equilibria that we leverage for optimality analysis. Next, we prove that the equilibria also maximize the global proportional fair resource allocation problem across all the BSs, and hence are globally optimal. Finally we discuss the uniqueness of the equilibria and prove that while the equilibrium throughput vector across all the clients is unique, there could be infinitely many resource allocations that realize this outcome. For simplicity, we do not consider discretization in this section.

Theorem 3 *Consider an equilibrium outcome of AFRA. Let r_i^{eq} denote the throughput of client i, θ_j^{eq} the water-fill level of BS j, and $\lambda_{i,j}^{eq}$ the fraction of time allocated to client i by BS j. Then*
(I) $\frac{\omega_i R_{i,j}}{r_i^{eq}} \leq \frac{1}{\theta_j^{eq}} \quad \forall i \in \mathbf{N}, j \in \mathbf{M}$
(II) $\sum_{i=1}^{N} \lambda_{i,j}^{eq} = 1 \quad \forall j \in \mathbf{M}$
(III) $\sum_{i=1}^{N} \omega_i = \sum_{j=1}^{M} \frac{1}{\theta_j^{eq}}$

Proof: Part 1. From the water-fill definition we have

$$R_{i,j}, \lambda_{i,j}^{eq} > 0 \implies \frac{r_i^{eq}}{\omega_i R_{i,j}} = \theta_j^{eq} \tag{11}$$

$$R_{i,j} > 0, \lambda_{i,j}^{eq} = 0 \implies \frac{r_i^{eq}}{\omega_i R_{i,j}} \geq \theta_j^{eq} \tag{12}$$

Property (I) follows from Eqs. (11) and (12).

Part 2. Every BS can always increase its water-fill level by distributing its unused time resources across its clients. The property follows, since at equilibrium the water-fill levels cannot be further increased.

Part 3. We leverage (I) and (II) to derive property (III) as follows

$$\sum_{i=1}^{N}\omega_i = \sum_{i=1}^{N}\frac{\omega_i r_i^{eq}}{r_i^{eq}} = \sum_{i=1}^{N}\sum_{j=1}^{M}\frac{\omega_i \lambda_{i,j}^{eq} R_{i,j}}{r_i^{eq}} = \sum_{\lambda_{i,j}^{eq}>0}\frac{\omega_i \lambda_{i,j}^{eq} R_{i,j}}{r_i^{eq}}$$

$$\overset{\text{Eq. (11)}}{=} \sum_{\lambda_{i,j}^{eq}>0}\frac{\lambda_{i,j}^{eq}}{\theta_j^{eq}} \overset{\text{(II)}}{=} \sum_{j=1}^{M}\frac{1}{\theta_j^{eq}} \tag{13}$$

■

We next show that any equilibrium outcome of AFRA is globally optimal, *i.e.*, it maximizes the global PF resource allocation problem across all the clients and BSs.

Theorem 4 *Consider an equilibrium outcome of AFRA. Then, the equilibrium outcome also maximizes the global PF resource allocation problem, i.e., it maximizes* $\sum_{i=1}^{N}\omega_i log(r_i)$ *subject to the feasibility constraints in Eqs. (1)-(3).*

Proof: Let r_i^{eq} and θ_j^{eq} denote the throughput of client i and water-fill level of BS j at an equilibrium, respectively.

We prove that for any feasible selection of $\lambda_{i,j}$s (*i.e.*, $\lambda_{i,j}$s that satisfy the feasibility conditions in Eqs. (2) and (3)) and the corresponding clients' throughput values (*i.e.*, r_is as defined in Eq. (1)) we have

$$\sum_{i=1}^{N}\omega_i \log(r_i) \leq \sum_{i=1}^{N}\omega_i \log(r_i^{eq}) \tag{14}$$

Define $\mathrm{W} = \sum_{i=1}^{N}\omega_i$. Eq. (14) can then be proved through the following inequalities by leveraging properties (I) and (III) from Theorem 3:

$$\sum_{i=1}^{N}\omega_i \log(r_i) - \sum_{i=1}^{N}\omega_i \log(r_i^{eq}) = \sum_{i=1}^{N}\omega_i \log(\frac{r_i}{r_i^{eq}}) =$$

$$\mathrm{W}\sum_{i=1}^{N}\frac{\omega_i}{\mathrm{W}}\log(\frac{r_i}{r_i^{eq}}) \overset{\text{Jensen Inequality [22]}}{\leq} \mathrm{W}\log(\sum_{i=1}^{N}(\frac{\omega_i}{\mathrm{W}}\times\frac{r_i}{r_i^{eq}})) =$$

$$\mathrm{W}\log(\frac{1}{\mathrm{W}}\times\sum_{i=1}^{N}(\frac{\omega_i r_i}{r_i^{eq}})) = \mathrm{W}\log(\frac{1}{\mathrm{W}}\times\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{\omega_i\lambda_{i,j}R_{i,j}}{r_i^{eq}})$$

$$\overset{\text{(I)}}{\leq} \mathrm{W}\log(\frac{1}{\mathrm{W}}\times\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{\lambda_{i,j}}{\theta_j^{eq}}) \overset{\text{Eq. (2)}}{\leq} \mathrm{W}\log(\frac{1}{\mathrm{W}}\times\sum_{j=1}^{M}\frac{1}{\theta_j^{eq}}) \overset{\text{(III)}}{=}$$

$$\mathrm{W}\log(\frac{1}{\mathrm{W}}\times\sum_{i=1}^{N}\omega_i) = 0 \tag{15}$$

■

In our last theorem we prove that while the equilibrium throughput vector across all clients is unique, there could be infinitely many resource allocations that realize this outcome.

Theorem 5 *Let* $\mathbf{r}^{eq} = (r_1^{eq}, ..., r_N^{eq})$ *denote the vector of throughput rates across all clients at an equilibrium. Then,* $\mathbf{r}^{eq}$ *is unique. However, there could be infinitely many resource allocations across the BSs that realize* $\mathbf{r}^{eq}$.

Proof: Part 1. We first prove that $\mathbf{r}^{eq}$ is unique. Let $\mathbf{r}^{eq}$ maximize the global proportional-fair resource allocation across all clients and assume $\mathbf{r}'^{eq}$ is a different equilibria. From Theorem 4, we know that every other equilibrium should also maximize the global PF resource allocation. This means that all inequalities in Eq. (15) should be equalities for any equilibrium, including $\mathbf{r}'^{eq}$. Now, for the first inequality to be an equality (*i.e.*, Jensen inequality of Eq. (15)), the following condition needs to be satisfied [22]

$$\frac{r_1^{eq}}{r_1'^{eq}} = \frac{r_2^{eq}}{r_2'^{eq}} = ... = \frac{r_n^{eq}}{r_n'^{eq}} \implies r_i'^{eq} = \alpha\, r_i^{eq} \quad \forall i \in \mathbf{N} \tag{16}$$

Further, since $\sum_{i=1}^{N}\omega_i \log(r_i^{eq}) = \sum_{i=1}^{N}\omega_i \log(r_i'^{eq})$, we conclude that

$$r_i^{eq} = r_i'^{eq} \quad \forall i \in \mathbf{N} \tag{17}$$

Part 2. To prove that there could be infinitely many resource allocations that realize $\mathbf{r}^{eq}$, we provide an example. Consider a topology with two BSs (j_1, j_2) and two clients (i_1, i_2). Let $R_{i_1,j} = 1 \ \ \forall j \in \mathbf{M}$, $R_{i_2,j} = 2 \ \ \forall j \in \mathbf{M}$, and $\omega_{i_1} = \omega_{i_2} = 2$. Then, $\sum\omega_i \log(r_i)$ is maximized by the following time fractions for any $\alpha \in [0 \ \ 1]$.

$$\begin{aligned} \lambda_{i,j} &= \alpha \quad \text{for} \ \ i = i_1, j = j_1 \ \ \text{and} \ \ i = i_2, j = j_2 \\ \lambda_{i,j} &= 1-\alpha \quad \text{for} \ \ i = i_1, j = j_2 \ \ \text{and} \ \ i = i_2, j = j_1 \end{aligned} \tag{18}$$

Here, irrespective of α, $r_{i_1} = 1$ and $r_{i_2} = 2$. ■

VI. Performance Evaluation

In this section, we evaluate AFRA's performance through experiments and simulations. First, we investigate the benefits of MAC level traffic aggregation in a small testbed composed of four SDR (software-defined radio)-based BSs and clients. Next, we conduct simulations to evaluate AFRA's equilibria properties as we scale the number of clients and BSs. Finally, we compare AFRA's speed and over-the-air signaling overhead against DDNUM, a dual decomposition based algorithm that we derived from the NUM framework [6].

A. SDR-Based Implementation and Real-World Performance

Implementation. We construct a HetNet topology composed of a WiFi BS, a cellular BS, and two clients. The two BSs are physically separated from each other and are placed in an indoor lab environment (Fig. 4(a)). We use a WARP software-defined radio [23] with 802.11a reference design as our WiFi BS. We use another WARP board with OFDM PHY (WARP OFDM reference design) and a custom TDMA (Time Division Multiple Access) MAC to mimic a cellular BS. We use two other WARP boards to construct our two clients. Each client has access to both WiFi and cellular radios, and remains static and connected to both BSs throughout the experiments.

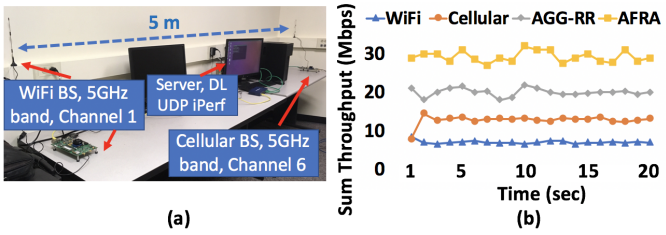


Fig. 4. **We use two WARP boards to construct two BSs in our testbed. The BSs are connected to a server through Ethernet. The server runs a single fully-backlogged downlink (DL) UDP iPerf session to each client. A sublayer implementation below the IP layer at the server, selects the BS for each packet of every traffic flow (a); Total throughput across the two clients for four schemes: WiFi only (WiFi), Cellular only (Cellular), AGG-RR, and AFRA. AFRA achieves a higher average total throughput (29 Mbps vs 20 Mbps) and PF index (2.3 vs 1.97) compared to AGG-RR.**

A server running iPerf sessions is connected to both BSs through Ethernet. For each client, the server generates a **single fully-backlogged UDP traffic flow** with 500 byte packets. We implement a below-IP sublayer to split this traffic flow between the two BSs. This sublayer is responsible for selection of the BS to be used for each packet, and acts similar to the LWA Adaptation Protocol (LWAAP) in the LWA standard [2]. In our implementation, we sequentially iterate between the WiFi and cellular BSs to route the packets of each traffic flow.

AFRA, as presented in Section III-D, does not account for various types of overhead (*e.g.*, PHY/MAC header, ACKs, idle slots, collisions) that exist in PHY/MAC protocols. To address the issue, we introduce the notion of effective rate (R^{eff}) and replace all $R_{i,j}$s in AFRA with $R^{\text{eff}}_{i,j}$s. For a single packet, R^{eff} can be calculated as the number of bits in the packet divided by the *total time* it takes by a BS to successfully transmit that packet (including all overhead). In our implementation, each BS keeps track of the total time spent in successfully transmitting the past 5 packets of each traffic flow (*i.e.*, the past 5 packets of each client) to calculate its $R^{\text{eff}}_{i,j}$. The averaging over 5 packets is to account for channel fluctuations in our experiments, and can be adjusted based on the client mobility.

We implement the following mechanisms: (i) WiFi only: the cellular BS is off but the WiFi BS is active; (ii) Cellular only: WiFi BS is off; (iii) AGG-RR: this scheme uses aggregation but with a round robin (RR) scheduler at the WiFi BS and conventional PF MAC at the cellular BS. With the RR scheduler, the WiFi BS maintains a different queue for each client and sequentially serves a single packet from each queue at every round. With the PF MAC at the cellular BS, the BS dedicates its time resources to each client according to Section III-C (single BS PF); (iv) AFRA: each BS uses its calculated $\lambda_{i,j}$s to determine the number of packets that should be served from each queue in WiFi and the number of time slots that should be dedicated to each queue (client) in cellular, at every round. In our implementation, both clients' ω_i are equal to 1 and the BSs updates their $\lambda_{i,j}$s every 5 ms[3].

Performance Results. Fig. 4(b) shows the performance of the four schemes. In both the *WiFi only* and *Cellular only* options, only a single BS is active throughout the experiments. We observe that the *Cellular only* scheme provides a higher sum throughput than the *WiFi only* scheme. With careful evaluation of packet transmission traces, we discovered that this higher throughput is primarily due to the corresponding MAC protocols. In particular, WiFi MAC provides the same *transmission opportunity* to each traffic flow (client). As a result, the client with lower PHY rate occupies the channel for a longer duration that the other client. This decreases the throughput for both clients. In contrast, the cellular TDMA MAC provides the same *transmission time* for both clients (with 2 clients, single BS PF equally divides the time between the clients (Eq. 5)). As a result, the throughput of the client with higher PHY rate does not drop because of the client with a lower PHY rate. This, along with other MAC issues such as WiFi contention reduce the WiFi only throughput.

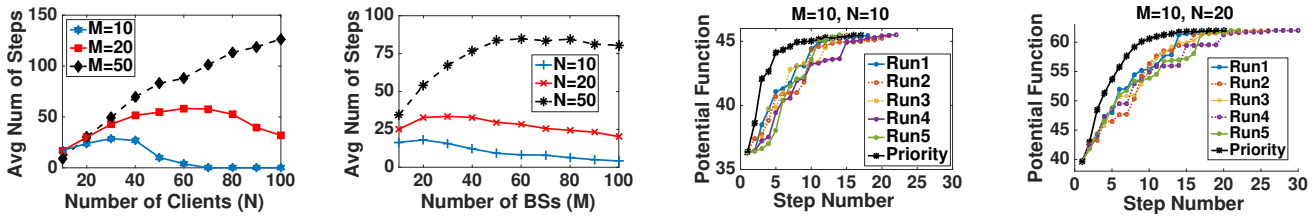
Fig. 4(b) also shows that the two RAT aggregation schemes (AGG-RR and AFRA) can successfully aggregate WiFi and cellular capacities and provide a higher sum throughput than the *WiFi only* and *Cellular only* options. Further, AFRA increases the average total throughput by 45% (from 20 to 29 Mbps) with 18 and 11 Mbps per-client total throughput values (per-client throughput plots are provided in [20]). Let us define the proportional fairness index as PF = $\sum_{i=1}^{2} \log(r_i)$ (r_i is the total throughput of each client across its RATs in Mbps). Then, the PF index in AFRA would be 2.3. With AGG-RR, the per-client throughput rates drop to 12.5 and 7.5 Mbps. Thus, the PF index reduces to 1.97. AGG-RR uses the conventional scheduling algorithms on each BS (*i.e.*, it uses RR in WiFi and single BS PF in cellular), which reduce both the sum throughput and the PF fairness index.

B. AFRA's Equilibria Properties

Setup. We simulated network deployments with N clients and M BSs to evaluate AFRA's equilibria properties as we scale the number of clients and BSs. All clients' ω_is are equal to 1. Half of the BSs are WiFi and the other half are cellular. Each client has access to 4 RATs, two WiFi and two cellular. The PHY rates for the WiFi and cellular RATs are randomly selected from the sets $\{1, 2, 5.5, 11\}$ Mbps and $\{5.2, 10.3, 25.5, 51\}$ Mbps, respectively. In each simulation realization, we randomly associate clients' RATs with BSs. Next, we run AFRA until an equilibrium is reached. We set the discretization factor ϵ equal to 0.05, *i.e.*, a BS adjusts its time fractions only if the increase in time fraction (*i.e.*, $\lambda_{i,j}$) at its client with minimum $\frac{r_i}{\omega_i R_{i,j}}$ is greater than or equal to 0.05. For the initial allocation, each BS equally divides its time across its clients. Unless otherwise specified, each of our simulation points is an average of 100 simulation realizations.

AFRA's Convergence Time. Figs. 5(a) and 5(b) depict the impact of the number of clients and BSs on AFRA's convergence time. In each of these figures, we count the number of steps until convergence is reached. At each step, a single BS that needs to adjust its time fractions is randomly selected. In Fig. 5(a), we vary the number of clients from 10 to 100 and plot the corresponding convergence times for three

[3]We intend to extend our work to study other issues such as TCP performance, network/flow dynamics, and PF scheduling in WiFi.



(a) (b) (c) (d)

Fig. 5. **AFRA's performance evaluation results. Average number of steps to convergence as a function of number of clients (a) and number of BSs (b). Evolution of potential function for two simulation scenarios one with M=10, N=10 (c) and the other with M=10, N=20 (d). Each Run in these figures corresponds to a different simulation realization. In the priority curves (solid black curve with * markers), the BS with the highest local increase in potential function gets priority in executing AFRA. Leveraging this policy reduces the average convergence time by more than 30%.**

different M values: 10, 20, and 50. We repeat this simulation by changing the N and M variables and plot the corresponding results in Fig. 5(b). From these two figures, we observe that time to convergence is highest when the number of clients is 1 to 2 times the number of BSs. As the ratio between the number of clients and BSs ($\frac{N}{M}$) leaves this range, the convergence time rapidly drops and then stabilizes. The results show that AFRA requires a small number of steps to reach an equilibrium.

Policies to Further Reduce AFRA's Convergence Time. Our next goal is to design policies that can further reduce AFRA's convergence time. To gain intuition on how to design such policies, we simulated a topology with 10 clients and 10 BSs and plotted the evolution of the potential function (*i.e.*, $\sum_i \log(r_i)$) as BSs adjusted their time fractions. The results are shown in Fig. 5(c). Here, each Run corresponds to a different simulation realization. From these realizations we make two observations. First, there is a wide gap in the convergence times. Second, a high jump in the potential function pushes the system closer to equilibrium. Based on these observations, we designed a *prioritization policy* among the BSs to reduce the convergence time.

We let each BS calculate the increase in the potential function assuming that it is the only BS executing AFRA. Since in AFRA each BS knows the current total throughput of its clients, it has all the needed information to calculate the increase in the potential function due to its action. Next, each BS broadcasts its calculated value. Finally, the BS with the highest value gets priority in executing AFRA. This distributed policy can be easily implemented in networks where all the BSs are connected to the same backbone (*e.g.*, Ethernet). The solid black curve in Fig. 5(c) shows the potential function's evolution with this policy. We observe that on average, the convergence time drops from 15 steps to 10, *i.e.*, the prioritization policy reduces the convergence time by 33%. We repeated this simulation for another setup with 20 clients to increase the topological redundancy. The results are plotted in Fig. 5(d). Similarly, the average convergence time reduces from 19 steps to 13, *i.e.*, a 32% reduction in convergence time.

C. Comparison Against DDNUM

We have compared AFRA's performance against DDNUM, a distributed algorithm that we developed by leveraging dual decomposition and the NUM framework. Dual decomposition is appropriate to solve the multi-RAT PF allocation problem, because the coupling constraint (Eq. (2)) can be relaxed through the dual problem and then the problem decouples into subproblems that can be iteratively solved by clients and BSs.

DDNUM is in essence similar to the standard dual algorithm presented in [6] to solve the basic NUM problem. We modified the algorithm in [6] to capture the constraints of our problem. At a high level, DDNUM has three main steps (for detailed algorithm derivation and discussions, refer to [20]):

- **Step 1:** Initialization: set $t = 0$ and $\boldsymbol{\mu}(0)$ to some nonnegative value for each BS. Here, $\boldsymbol{\mu}(t)$ is the vector of Lagrange multipliers that shows the cost or congestion across all BSs. Each BS broadcasts its $\mu_j(0)$ to clients with $R_{i,j} > 0$.
- **Step 2:** Each client i locally solves its Lagrangian problem, *i.e.*, finds its time fractions $(\lambda^*_{i,j}(\mu_j(t)))$ for each BS with $R_{i,j} > 0$ and informs those BSs.
- **Step 3:** Each BS updates its price with a step size γ and broadcasts the new price $\mu_j(t+1)$ to all its clients.

This procedure is repeated until a satisfying termination point is reached (*e.g.*, the solution is within a desired proximity of the optimal solution). Similar to AFRA, DDNUM is guaranteed to converge and maximize the global optimization problem [20]. However, there are several practicality and performance issues. We highlight a few of these issues next.

Setup. To compare AFRA to DDNUM, we used the simulation setup in Section VI-B (without the BS prioritization policy). We first run AFRA and let the system converge to an equilibrium. Next, we consider the 95% value of AFRA's potential function at equilibrium as the desired algorithm termination point. We count the number of steps to reach the termination point and the resulting over-the-air signaling overhead in each of these two schemes. In DDNUM, the step size γ (step 3) provides a balance between the final throughput values and speed. We choose the γ that results in the fastest convergence time, subject to the potential function reaching the termination point. Finally, both AFRA and DDNUM can operate in either parallel or sequential mode with similar relative performance. We present the sequential mode results, *i.e.*, at each time only a single BS adjusts its water-fill level (in AFRA) or announces a new price (in DDNUM). We assume that clients immediately update their BSs about their new throughput values (in AFRA) and desired $\lambda_{i,j}$s (in DDNUM) with no impact on convergence time (similar to an FDD system in which uplink data is immediately available at each BS).

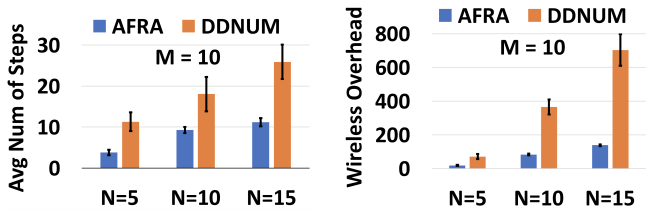


Fig. 6. **Compared to AFRA, DDNUM increases the average convergence time by 2.4x and the average over-the-air signaling overhead by 4.5x.**

Speed. Fig. 6(a) show the convergence time results for a scenario with 10 BSs and varying number of clients. We observe that irrespective of the number of clients, DDNUM increase the convergence time by a factor of 2-3x with an average of 2.4x. In AFRA, each BS simultaneously calculates the water-fill level and finds the corresponding time fraction for each client. In DDNUM, the pricing mechanism requires a high number of iterations so that clients can find their optimal time fractions. This increases the convergence time.

Over-the-Air Overhead. Fig. 6(b) shows the wireless signaling overhead results of the two schemes. We observe that DDNUM increases the signaling overhead by a factor of 4-5x with an average of 4.5x. There are several factors that contribute to DDNUM's high signaling overhead. First, the increases in convergence time results in a similar multiplicative increase in overhead. Second, in DDNUM both BSs and clients contribute to overhead. BSs continuously broadcast new prices and clients continuously inform each of their BSs about their desired time fractions. In contrast, in AFRA only clients update the BSs regarding their new throughput values. Third, with careful examination of simulation traces, we observed that in AFRA the water-fill operation only impacts a few of a BS's clients each time. In DDNUM, each time a BS updates its price, most of its clients would request new time fractions.

Practicality. In DDNUM, each BS broadcasts its price while each client finds its desired $\lambda^*_{i,j}$s from its BSs. However, in real wireless systems BSs are responsible for resource allocation. Note that in DDNUM, it is not practical to shift the calculation of $\lambda^*_{i,j}$s (*i.e.*, step 2) to BSs. This is because in order for a BS j to find the $\lambda^*_{i,j}$s for each of its clients (*e.g.*, i), it would require knowledge about the client's $R_{i,j}$ and μ_j to every other BS for which the client's rate (*i.e.*, $R_{i,j}$) is greater than zero. This information is only available at the client and pushing it to the BS would significantly increase the overhead, which is already very high in DDNUM.

Complexity. In DDNUM, each client has to solve a complex Lagrangian subproblem to find its desired time fraction for each BS (step 2). This increases the computational complexity on the client devices. In contrast, AFRA identifies the time resources at the BSs, which have higher power and computing resources. Moreover, as we discussed in Section III-D, AFRA has a very low total computational complexity.

VII. Conclusion

We addressed the problem of proportional fair multi-RAT traffic aggregation in HetNets. We studied the conventional PF resource allocation in a single BS and showed that we can look at the problem as a special type of water-filling. Based on this observation, we designed a new fully distributed water-filling algorithm for HetNets. We also studied the convergence, speed, and optimality of our algorithm. We proved that our algorithm quickly converges to equilibria and derived tight bounds to quantify its speed. We also studied the characteristics of the optimal outcome, and used the properties to prove the outcomes of our algorithm are globally optimal.

VIII. Acknowledgement

The authors would like to thank the anonymous reviewers for their useful comments.

References

[1] "Samsung download booster: use WiFi and LTE simultaneously," https://www.pcmag.com/article2/0,2817,2455011,00.asp
[2] 3GPP, "Introduction of LTE-WLAN radio level integration and interworking enhancement," in *3GPP Technical Report, R2-156737*, 2015.
[3] A. Zakrzewska, D. Lopez-Perez, S. Kucera, and H. Claussen, "Dual connectivity in LTE hetnets with split control and user plane," in *Proceedings of IEEE GLOBECOM Workshops*, 2013.
[4] 3GPP, "Study on new radio (NR) access technology (release 14)," in *3GPP Technical Report, TR 38.912*, 2017.
[5] FCC, "2016 broadband progress report," January 2016.
[6] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," in *IEEE Journal on Selected Areas in Communications*, 2006.
[7] X. Lin, N. B. Shroff, and R. Srikant, "A tutorial on cross-layer optimization in wireless networks," in *IEEE Journal on Selected Areas in Communications*, 2006.
[8] A. Stolyar, "On the asymptotic optimality of the gradient scheduling algorithm for multi-user throughput allocation," in *Operations Research Journal*, 2005.
[9] S.-B. Lee, S. Choudhury, A. Khoshnevis, S. Xu, and S. Lu, "Downlink MIMO with frequency-domain packet scheduling for 3GPP LTE," in *Proceedings of IEEE INFOCOM*, 2009.
[10] S. Singh, M. Geraseminko, S.-P. Yeh, N. Himayat, and S. Talwar, "Proportional fair traffic splitting and aggregation in heterogeneous wireless networks," in *IEEE Communications Letters*, 2016.
[11] E. Aryafar, A. K. Haddad, C. Joe-Wong, and M. Chiang, "Max-min fair resource allocation in hetnets: Distributed algorithms and hybrid architecture," in *Proceedings of IEEE ICDCS*, 2017.
[12] D. Ibarra, N. Desai, and I. Demirkol, "Software-based implementation of LTE/Wi-Fi aggregation and its impact on higher layer protocols," in *Proceedings of IEEE ICC*, 2018.
[13] Y. Khadraoui, X. Lagrange, and A. Gravey, "Implementation of LTE/WiFi link aggregation with very tight coupling," in *Proceedings of IEEE PIMRC*, 2017.
[14] T. V. Pasca, N. Sen, V. Reddy, B. R. Tamma, and A. Franklin, "A framework for integrating MPTCP over LWA - a testbed evaluation," in *Proceedings of ACM WiNTECH*, 2018.
[15] D. P. Bertsekas and J. N. Tsitsiklis, "Parallel and distributed computation: numerical methods," in *Englewood Cliffs, NJ: Prentice-Hall*, 1989.
[16] W. Wang, X. Liu, J. Vicente, and P. Mohapatra, "Integration gain of heterogeneous WiFi/WiMAX networks," in *IEEE Transactions on Mobile Computing*, 2011.
[17] Q. Ye, B. Rong, Y. Chen, M. Al-Shalash, C. Caramanis, and J. G. Andrews, "User association for load balancing in heterogeneous cellular networks," in *IEEE Transactions on Wireless Communications*, 2013.
[18] S. Shakkottai, E. Altman, and A. Kumar, "Multihoming of users to access points in WLANs: a population game perspective," in *IEEE Journal on Selected Areas in Communication*, 2009.
[19] L. Li, M. Pal, and Y. Yang, "Proportional fairness in multi-rate wireless lans," in *Proceedings of IEEE INFOCOM*, 2008.
[20] Technical Report, available at arXiv:1906.00284
[21] A. Monderer and L. S. Shapley, "Potential games," in *Games and Economic Behavior*, 1996.
[22] Jensen Inequality, https://en.wikipedia.org/wiki/Jensen%27s_inequality
[23] "WARP Project," https://warpproject.org/trac