An Approximation Algorithm for AP Association under User Migration Cost Constraint

Wangkit Wong Avishek Thakur S.-H. Gary Chan
Department of Computer Science and Engineering
The Hong Kong University of Science and Technology, Hong Kong, China
Email: {wwongaa, akthakur, gchan}@cse.ust.hk

Abstract—**In wireless local area networks (WLANs), association of the access points (APs) is often not uniform due to joining and leaving of wireless users. Such imbalance leads to unsatisfactory user throughput due to congestion at some APs and channel under-utilization at others. As users may be covered by multiple APs in WLAN deployment, their throughput can be improved by AP re-association, i.e., migrating some users from the congested APs to the less loaded neighboring APs. Such re-association is expected to come with some cost (due to overhead in handshaking, authentication and data flow management).**

In this paper, we study the novel problem of optimizing AP re-association by maximizing the minimum user throughput (i.e., max-min fairness), subject to a certain total user migration cost constraint. We show that the problem is NP-hard. We then propose an efficient approximation algorithm called CACA (Cost-constrained Association Control Algorithm). CACA has provable performance, achieving an approximation factor of $(4+\epsilon)$, for any $\epsilon > 0$. It is simple and implementable. Our extensive simulation results based on NS3 show that it substantially outperforms other comparison schemes with close-to-optimal performance (the case without cost constraint). Our testbed experiments further confirm the effectiveness of CACA.

Keywords*—*WLANs, Association control, Optimization, Approximation algorithm, Migration cost, Load balancing.

I. Introduction

IEEE 802.11-based wireless local area network (WLAN) offers mobile users (MUs) access to the Internet via access points (APs). These AP coverages often overlap, due to inflexible locations to install APs, the need to reduce blind spots, the requirement on signal strengths, etc. Consequently, an MU is often simultaneously covered by multiple APs. Such overlap offers an opportunity to optimize the AP association of clients so as to improve network performance.

In a network consisting of multiple APs, it is desirable to optimize AP association over time. This is because user traffic is dynamic, unpredictable and spatially non-uniform. As the system evolves with MU joining and leaving, user throughput would no longer be balanced for all the APs, with clusters of MUs associated with some "hot" APs while some "colder" ones with few MUs. This leads to unsatisfactory throughput performance at congested APs and channel under-utilization at others. To address it, we can use an AP association controller to migrate (i.e., re-associate) some connected users from the congested APs to those lightly loaded neighboring APs.

This work was supported, in part, by the HKUST Special Research Fund Initiative (SRFI11EG15), and Hong Kong Research Grant Council (RGC) General Research Fund (610713)

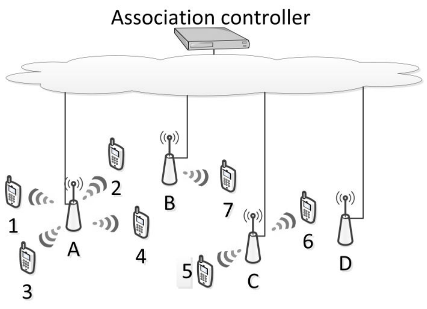

Fig. 1. A wireless netwrok with non-uniform user association. Using strongest signal association, users 1, 2, 3 and 4 associate with AP A, creating congestion at the AP.

We show in Figure 1 a wireless network formed by multiple APs with associated clients as indicated. The MUs may not be uniformly distributed. Without loss of generality, we consider that each AP is equipped with one interface (for an AP with multiple interfaces, we may consider it as multiple co-located APs with single interface). Typically, the power range of the APs is fixed, and their channels are properly set such that co-channel interference in their overlapping regions is minimal.

In a WLAN, a joining MU usually scans all the APs which it can associate with. Then, it associates with the one according to some association rule such as strongest beacon power or maximum expected throughput [1]. Such AP association, usually performed in a distributed and local manner, is not globally optimal in terms of throughput. Furthermore, users may leave the network any time. Previous work often considers that AP association is *irrevocable*, i.e., once associated with an AP at join time, an MU cannot change the association during its lifetime. As a result, the performance tends to degrade over time due to sub-optimal AP association. We hence need to re-optimize association over time by migrating some MUs, either periodically or upon detecting a drop in network performance. A desirable objective for AP re-association is network-wide fair bandwidth allocation. In this paper, we consider max-min fairness as it maximizes the fair share of each user.

To effect AP re-association, the MU needs to handshake with the target AP for authentication and association. Furthermore, the target AP and the underlying network need to re-establish new route for the packet flow to the user. Therefore, there is a migration cost in AP re-association, because of the traffic and management overhead which may even lead to (momentary) connection interruption. Such cost may be heterogeneous for different users, due to user priority or the elasticity of its current connection.

Association re-optimization may be conducted by a central association controller. The controller computes which MU should be associated with which AP according to some rules or objectives (such as load balancing). Prior art has not considered the impact of extensive user migration [2], [3]. As a result, re-optimizing AP association may lead to global large-scale MU migration or oscillation. This is undesirable, disruptive and costly.

We consider the general case of heterogeneous user migration cost. The critical problem is then how to maximize the minimum user throughput (max-min fairness) subject to a certain total cost constraint. Note that for the special case of homogeneous (uniform) migration cost, our problem is reduced to achieving max-min fairness subject to a given maximum number of total migrations.

Our problem is novel and practical because previous work on AP association has not considered migration cost constraint. The introduction of such constraint fundamentally changes the nature of the problem and merits a new study. We approach the problem via the following:

- *Problem formulation and complexity analysis:* We formulate the optimization problem for AP re-association subject to a cost constraint. The problem is an integer linear programming, and we show that it is NP-hard.
- *An approximation algorithm for AP re-association under migration cost constraint:* We propose CACA (**C**ost-constrained **A**ssociation **C**ontrol **A**lgorithm), to address the optimization problem. CACA is simple and implementable. It has provable performance, achieving an approximation factor of $4 + \epsilon$.
- *Extensive NS3 simulations and experimental studies:* We demonstrate the effectiveness of CACA by validating its performance through extensive packet-level NS3 simulations and real proof-of-concept experiments. We demonstrate that CACA achieves highly optimal user throughput, with performance close to the optimum (as given by the scheme without cost consideration). Our experiments further confirm its implementability and high performance.

The rest of the paper is organized as follows. We first discuss related work in Section II, followed by discussing the system models and problem formulation in Section III. In Section IV we present CACA, an efficient approximation algorithm for migrating MUs to achieve near-optimal throughput. We present in Section V illustrative simulation results with NS3 and proof-of-concept experimental studies. We conclude in Section VI. Important proofs are provided in the Appendix.

II. Related works

AP association has been an active research topic. There has been much research to alleviate the disruptive effect of re-association. Ramya et al. [4] analyze how a large number of re-association affect performance. Approaches in [5]–[7] attempt to reduce the re-authentication management frames by caching and propagating authentication information among APs. The work in [8] minimizes the number of re-associations in a mobile network. All these work shows that re-association is costly and should be contained.

Much work has studied on-line AP association schemes [9]–[12]. In these schemes, the APs broadcast their information, such as load conditions or queue length. Each arriving user makes decision on which AP to associate to. While these works are impressive, the schemes are not adaptive to achieve high performance by accommodating dynamic traffic when users may join and leave any time.

There has been work which makes periodic association decision over time by a central controller [2], [3], [13]–[16]. It recalculates the optimal user association by an offline algorithm, and migrates users accordingly. However, migration cost has not been considered and hence the migration can be substantially costly or disruptive to the network. We present here a formulation with migration cost constraint, and a novel algorithm which achieves low migration cost with guaranteed performance (in terms of user throughput).

Another body of work studies distributed association control algorithm [17]–[20]. MUs in these schemes distributively switch to the suitable AP whenever their throughput can be improved. These works have not considered migration cost, and it is unclear how they perform (i.e., performance bound) with respect to the optimum. CACA, on the other hand, is an approximation algorithm with provable performance.

III. System Model and Problem Formulation

A. Network Model

We consider a general 802.11 WLAN consisting of a number of APs, with properly set channels such that co-channel interference is negligible. Each AP is wired to the Internet. Let A be the set of APs and m be its cardinality, i.e. $m = |A|$. The transmission range of an AP ($i \in A$) is given by $R(i)$. For AP i, it can only be associated with users in its coverage.

We denote the set of MUs as U, where $|U| = n$. Users in the network are dynamic (in the sense that they may join or leave at any time). Let r_{ij} be the time-averaged data rate between user j and AP i. An MU accesses to the Internet by associating to any AP. We denote the set of APs that MU i is potential to associate with as $A(i)$. Analogously, $U(i)$ is the set of MUs associated with AP i.

Data rate r_{ij} can be calculated from channel scanning at the user. User j scans the channel by sending probe request frames. The APs in the range reply with probe response frames. According to the RSSI (Received Signal Strength Indicator) of the response frames from AP i, the client can calculate which modulation scheme is going to be used if it associates with i. Each modulation scheme corresponds to a certain data rate, thus we can get the rate r_{ij}.

There is a central controller in the network. The central controller carries out re-optimization periodically or when AP congestion is detected. Our association control consists of two phases, *joining phase* and *re-optimization phase*, discussed below:

Joining phase: On the joining of an MU, it associates with an AP according to any association rule such as strongest beacon power based association or maximum expected throughput association [1].

Re-optimization phase: In this phase, APs report the current association to the central controller. We first decide which subset of MUs should be migrated according to current AP load information. Our decision should not violate the migration cost constraint. Then, the central controller requests the selected MUs to scan for feasible APs for migration. These MUs report the scan results (i.e. the expected data rate r_{ij} from user j to potential AP i) to the central controller. With the scan information, central controller chooses an optimal AP from the candidate APs for each MU being migrated and sends the migration decision to the corresponding MU. Finally, A migrating MU dissociates from the current AP and re-associates to the AP chosen by central controller via control frames exchanging with APs.

Since only the migrating MUs perform channel scanning, the introduction of migration cost constraint also limits scanning cost. To prevent an MU from being migrated too frequently (i.e., flip-flopping between APs), we can increase the migration cost of an MU according to its migration/handover history. This is important to maintain association stability, thereby enhancing user experience.

B. Multiple Access Model

Because our objective is to maximize the minimum throughput for active users, we consider heavy traffic case, i.e., each active user always has packets to send or receive. Due to Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) mechanism in 802.11 protocols, all MUs associated with the same AP will seize the AP's channel for packet transmission with the same probability. We may view that APs serve their associated clients in the round-robin manner. Therefore, for any given packet size δ (bits), the service time t_{ij} (seconds) allocated to user j by its associated AP i is proportional to the time needed for one packet transmission between i and j, given by $t_{ij} = (\delta/r_{ij})/(\sum_{k\in U(i)} \delta/r_{ik})$. This model is also confirmed by [21].

Let T_j be the throughput of j and $d_{ij} = 1/r_{ij}$. MUs associated with the same AP i get the same throughput in spite of the difference in their data rates. The throughput they get is given by $T_j = r_{ij}t_{ij} = 1/(\sum_{k\in U(i)} d_{ik})$.

C. Problem Formulation

We begin with some definitions. The notations we use have been summarized in Table I. Let x be the state of the current association, and $a(j)$ the AP that j is currently associated with. If j is associated with i, then $x_{ij} = 1$; otherwise $x_{ij} = 0$.

Definition 1. *AP Load:* The load of AP i is defined as $L_i = \sum_j x_{ij}d_{ij}$.

Definition 2. *Max-min User Throughput:* An AP association x^* achieves max-min user throughput if the network-wide minimum user throughput is maximized. i.e. $x^* = \arg\max_x \min_j T_j$, subject to $x_{ij} \in \{0,1\}$ and $\sum_{i\in A(j)} x_{ij} = 1, \forall j$.

Due to the DCF MAC in 802.11, all MUs associated with the same AP have similar throughput. It is hence natural to consider maximizing the minimum user throughput as the objective.

TABLE I. MAJOR SYMBOLS USED IN THIS PAPER.

Notation	Definition
r_{ij}	the data rate between user j and AP i (Mbits/s)
d_{ij}	$\frac{1}{r_{ij}}$ (seconds)
T_j	throughput of user j (Mbits/s)
L_i	current load of AP i
L^*	optimal value of Min-max AP Load Problem (MinmaxLP)
x_{ij}	binary variable indicates whether user j associates with i in the current association
$a(j)$	the AP user j is currently associated with
$A(j)$	the set of APs that MU j is possible to associate with
$U(i)$	the set of Users currently associated with i
c_j	the migration cost of user j
y_{ij}	binary variable indicates whether user j should dissociate with the current AP $a(j)$ and be reassigned to AP i
G_i	the load of AP i after MU removing
G	$max_i G_i$, maximum AP load after MU removing
D	the subset of MUs removed by MRemove
H_i	the load of AP i after MU migration
H	$max_i H_i$, maximum AP load after MU migration
K	maximum migration cost constraint

Definition 3. *Min-max AP Load Problem (MinmaxLP):* $\arg\min_x \max_i L_i$, s.t. $x_{ij} \in \{0,1\}$ and $\sum_{i\in A(j)} x_{ij} = 1, \forall j$, where $i \in A, j \in U$.

The achievable throughput of MUs is the inverse of the load of the AP that they are associated with. For each MU j, $T_j = 1/L_i$, where $i = a(j)$. It is not hard to see that maximizing $\min_j T_j$ is equivalent to minimizing $\max_i L_i$. Therefore, x^* is the optimal solution to MinmaxLP. Let H_i be the load of AP i after MU migration. For the sake of conciseness in notation, we will consider $\min\max_i H_i$ as the objective of our association control in the rest of this paper. Formally, our optimization problem can be stated by Definition 4.

Definition 4. *CACP (Cost-constrained Association Control Problem):* Given A, U, r_{ij} and current user-AP association x, re-associate users by migration to minimize $\max_i H_i$, under the constraint that no more than a given K migration cost is incurred.

We next formulate the CACP. Let y_{ij} be the binary variable indicating whether user j is dissociated with the current AP and re-associated to AP i. We denote the load of AP i given by the current association as L_i. We can view association optimization as removing some MUs from the current association and re-associate them to some APs. Clearly, $\sum_{j:a(j)=i}\sum_{i':i'\neq i, i'\in A(j)} d_{ij}y_{i'j}$ is the amount of load re-associated to other APs from i. Let G_i be the load of AP i after MU removing. We have

$$G_i = L_i - \sum_{j:a(j)=i}\ \sum_{i':i'\neq i, i'\in A(j)} d_{ij}y_{i'j}. \tag{1}$$

Clearly, $\sum_{j:i\in A(j), a(j)\neq i} d_{ij}y_{ij}$ is the total amount of load migrated to AP i from other APs. Therefore,

$$H_i = G_i + \sum_{j:i\in A(j), a(j)\neq i} d_{ij}y_{ij}. \tag{2}$$

Let $H = max_i H_i$. The objective of CACP can be expressed as:

$$\min H. \tag{3}$$

We denote the migration cost of MU j as c_j. As the migration cost can be at most K, the following constraint must be in place:

$$\sum_i \sum_j y_{ij} c_j \leq K. \tag{4}$$

We also require $y_{ij} \in \{0, 1\}$. Clearly, constraints $\sum_i y_{ij} \leq 1, \forall j$ are implied by the objective. If we drop the migration cost constraint given by Inequality (4), CACP reduces to MinmaxLP. It is not hard to see the following is true as the MinmaxLP is more relaxed than the CACP.

Lemma 1. *Let L^* be the optimal value of the Min-max AP Load Problem (MinmaxLP) and OPT be the optimal value of CACP. $L^* \leq OPT$.*

We now analyze the hardness of CACP. We define the restricted version of Cost-constrained Association Control Problem by restricting $K = \sum_j c_j$ as RCACP. The CACP is NP-hard as the PUMSP (Parallel Unrelated Machine Scheduling Problem) can be reduced to RCACP. The formal definition of PUMSP is stated in [22].

IV. CACA: An Approximation Algorithm for Cost-Constrained AP Re-association

In this section, we first provide some background knowledge for our study (Section IV-A). Then we overview the important elements of CACA (Section IV-B), followed by its details on MU removal (Section IV-C) and re-association (Section IV-D).

A. Preliminary

Since Shmoys and Tardos' rounding algorithm is used in our algorithm design, we here briefly discuss their major result in Theorem 1 [22]. Given $\lambda \in \mathbb{R}^m_+$, $D \in \mathbb{R}^{m \times n}_+$, $\lambda = (\lambda_i)$, $D = (d_{ij})$ and $\max_j d_{ij} \leq \lambda_i, \forall i$, we have the feasibility problem given by the following Integer Linear programming ILP(λ):

$$\sum_i x_{ij} = 1, \forall j$$

$$\sum_j d_{ij} x_{ij} \leq \lambda_i, \forall i \tag{5}$$

$$x_{ij} \in \{0, 1\}. \tag{6}$$

We refer problems without objective as feasibility problems. By relaxing the integer constraint (6) to $0 \leq x_{ij} \leq 1$, we get the corresponding Linear Programming denoted by LP(λ).

Theorem 1. *(Rounding Theorem). If the feasibility problem given by LP(λ) has a feasible fractional solution x, then the fractional solution x can be rounded (via Shmoys and Tardos' rounding algorithm) to a feasible integer solution $\bar{x}$ such that $\sum_j d_{ij} \bar{x}_{ij} \leq 2\lambda_i, \forall i$.*

Theorem 1 states that if $max_j d_{ij} \leq \lambda_i, \forall i$, we can use Shmoys and Tardos' rounding algorithm to round a feasible fractional solution x of LP(λ) to an integer solution $\bar{x}$ such that $\bar{x}$ violates the constraint (5) by at most λ_i. This implies Shmoys and Tardos' rounding algorithm is a factor 2 approximation algorithm.

We now introduce the Minimum knapsack problem. Given a set of items, a_j and b_j are the value and cost of item j respectively. The problem is to select a subset of items to minimize the sum cost under the constraint that the sum value is at least V. An Integer Minimum knapsack Problem instance IMKP(a, b, V) can be formulated as

$$\min \sum_j b_j z_j,$$

$$\sum_j a_j z_j \geq V, z_j \in \{0, 1\},$$

where z is the binary decision variable. An IMKP can be solved optimally in pseudo-polynomial time by dynamic programming. We denote the algorithm solving IMKP as IMKA (Integer Min-knapsack Algorithm). (I^*, B^*) = IMKA (a, b, V), where I^* is the optimal subset and B^* is the optimal cost.

B. Algorithmic Overview

We propose a 2-step approximation algorithm termed CACA (Cost-constrained Association Control Algorithm) to tackle CACP (Definition 4). Our task is to migrate MUs from heavily loaded APs to neighboring lightly loaded APs. We finish this task in two steps. In Step 1, we will remove a subset of MUs from the current association. In Step 2, we re-associate the removed MUs. Therefore, we decompose the CACP into two sub-problems: MU Removal Problem (MRP) and MU Re-association Problem (MRAP).

C. MU Removal

We need to determine which MUs to be removed from the current association. Let z_j be the binary variable indicating whether j is removed from $a(j)$. Recall that G_i is the load of AP i after MU removal. Clearly,

$$G_i = L_i - \sum_{j:a(j)=i} d_{ij} z_j.$$

We want to reduce the maximum AP load as much as possible. Therefore, the MU Removing Problem is to minimize $\max_i G_i$, given by the following ILP.

Sub-problem 1 (MRP):

$$\min \max_i G_i,$$

$$\sum_j z_j c_j \leq K, \tag{7}$$

$$z_j \in \{0, 1\}. \tag{8}$$

We present MRemove (MU Removal Algorithm) in Algorithm (1) to tackle MRP. We denote the maximum AP load

Algorithm 1: MRemove

```
Input: $A$, $U$, $d_{ij}$, $x$
Output: $D$, $G_i$
1  $lb \leftarrow 0$, $ub \leftarrow max_i L_i$. $D \leftarrow null$.
2  while $ub > (1+\epsilon)lb$ do
3      $I \leftarrow null$. $B \leftarrow 0$.
4      $g = (lb + ub)/2$.
5      foreach $i \in A$ do
6          Construct IMKP($a$, $b$, $V$) on set $U(i)$ via the
           following. Each MU $j$ in $U(i)$ corresponds to
           an item. Set $a_j = d_{ij}$, $b_j = c_j$ for each $j$. Set
           $V = \max\{L_i - g, 0\}$.
7          //$a$ and $b$ are vectors with $a_j$ and $b_j$ as
8          //their $j$th element respectively
9          ($I_i$, $B_i$) = IMKA ($a$, $b$, $V$).
10         $I \leftarrow I \bigcup I_i$.
11         $B \leftarrow B + B_i$.
12     end
13     if $B \leq K$ then
14         $D \leftarrow I$.
15         $ub \leftarrow g$.
16     else
17         $lb \leftarrow g$.
18     end
19 end
```

after MUs removed by MRemove as G, i.e. $G = max_i G_i$. Let set $D^* \in U$ be the optimal set of users to remove and G^* be the optimal value of MRP. Please note that D^* may not be the optimal set of MUs to migrate for the CACP.

In MRemove, we first establish the upper bound ub and lower bound lb of G^*. Then, we make an initial guess of optimal value G^* (say $g = (lb + ub)/2$ is our guess of G^*). For each guess g, we try to achieve it by removing enough MUs in the cheapest way (in terms of migration cost). If we can achieve g, we decrease our guess; otherwise, we increase our guess.

IMKA represents the well-known dynamic programming algorithm for solving IMKP (see Preliminary). Parameter $\epsilon > 0$ can be any small number. It determines the time complexity of Algorithm (1). In each iteration, we guess the optimal value to be g. To achieve the objective value g, we have to reduce the load of each AP i. Therefore, we have an MU removing sub-problem for each AP i. The problem is to remove a subset of $U(i)$ to reduce L_i by at least $\max\{L_i - g, 0\}$ while inducing the least cost. We solve the MU removing problem for each AP i by reducing it to IMKP via the construction in Line 6. Then, we use IMKA (Line 9) to calculate the optimal subset $I_i \in U(i)$ to remove from i. $I = \bigcup_i I_i$ is the optimal subset to achieve objective g. B defined in Line 3 calculates the total cost for removing I. If $B \leq K$, objective g can be achieved. If B exceeds K, there is no feasible MU removal that can achieve g. If g can be achieved with no more than K cost, it implies that a lower objective may be possible. Therefore, we set $ub = g$ and reduce g (by Line 4), when g is achievable; set $lb = g$ and increase g, otherwise. We always use D to record the set whose removal results in achievable maximum AP Load g (Line 14). As achievable g keeps decreasing through algorithm iterations, the removal of D will achieve the lowest objective.

Lemma 2. *MRemove is a* $(1+\epsilon)$ *approximation algorithm to MRP (i.e.* $G \leq (1+\epsilon)G^*$*). (See Appendix for proof.)*

Lemma 3. $G^* \leq OPT$. *(See Appendix for proof.)*

For the case that all MUs have the same migration cost, the constraint given by Inequality (7) becomes $\sum_j z_j \leq \lfloor K/c \rfloor$ (where c is the migration cost of each MU). We denote $\lfloor K/c \rfloor$ as K'. The constraint reduces to that no more than K' MUs can be removed. For this case, we use a greedy algorithm to remove the optimal subset of MUs. We call AP i the maximum loaded AP if $i = \arg\max_k L_k$. Similarly, user j is called the heaviest MU if $j = \arg\max_k d_{ik}$. The algorithm is to remove the heaviest MU from the maximum-loaded AP for K' times. Clearly, this algorithm is optimal for homogeneous case.

D. MU Re-association

As MUs in set D are removed from the current association, we need to re-associate them. Let binary variable γ_{ij} indicate whether MU $j \in D$ is re-associated with i. The load of AP i after MU re-association is given by

$$H_i = G_i + \sum_{j \in D} \gamma_{ij} d_{ij}.$$

H_i is also the load of AP i after the optimization of CACA. Subsequently, $H = \max_i H_i$ is the maximum AP load. Therefore, the MU Re-association Problem (MRAP) is to minimize H, given by the following ILP:

Sub-problem 2 (MRAP):

$$\min H$$

$$\sum_{j \in D} \gamma_{ij} d_{ij} \leq H - G_i, \forall i, \tag{9}$$

$$\sum_{i: i \in A(j)} \gamma_{ij} = 1, \forall j \in D, \tag{10}$$

$$\gamma_{ij} \in \{0, 1\}. \tag{11}$$

Let H^* be the optimal value of MRAP. MRAP is NP-hard. However, we can easily find the lower bound (lb) and upper bound (ub) of H^*. We solve the Linear Programming version of MRAP (by relaxing constraint (11)) to get the fractional optimal value H_f and set $lb = H_f$. The objective value of any feasible solution to MRAP is an upper bound of H^*. A feasible solution γ can be found easily (such as associating each MU in D to a randomly selected AP). For any given value g, the corresponding feasibility problem FMRAP(g) is given by

FMRAP(g):

$$\sum_{j \in D} \gamma_{ij} d_{ij} \leq g - G_i, \forall i,$$

$$\sum_{i: i \in A(j)} \gamma_{ij} = 1, \forall j \in D.$$

It is not hard to observe that FMRAP(g) is the same as the LP(λ) in Theorem 1 (by setting $\lambda_i = g - G_i$). With the help of Theorem 1, we propose MAssoc shown in Algorithm (2)

Algorithm 2: MAssoc

```
    Input: $D$, $G_i$.
    Output: $\gamma^I$.
 1  solve the LP-relaxation of MRAP to get $H_f$, $lb \leftarrow H_f$.
 2  Set $ub$ equal to the value of any feasible solution.
 3  while $ub > (1+\epsilon)lb$ do
 4      $g \leftarrow (lb + ub)/2$.
 5      For all $i$ and all $j$, delete $i$ from $A(j)$, if
        $d_{ij} \geq g - Gi$.
 6      if FMRAP(g) has feasible fractional solution then
 7          $ub \leftarrow g$.
 8      else
 9          $lb \leftarrow g$.
10      end
11  end
12  Assign the fractional solution of FMRAP($ub$) to $\gamma$.
13  Use Shmoys and Tardos' rounding algorithm to round $\gamma$
    to an integer solution $\gamma^I$.
```

to address MRAP. We search the optimal fractional solution via binary search. Then we round the best solution we found to get a factor 2 integer solution. Line 5 ensures the condition (i.e. $\max_j d_{ij} \leq \lambda_i, \forall i$) required by Theorem (1) and does not affect the optimality of the fractional solution. Parameter $\epsilon > 0$ controls the time complexity, and can be arbitrary small.

Lemma 4. *For any i, $\sum_{j\in D} \gamma^I_{ij} d_{ij} \leq 2(1+\epsilon)H^* - 2G_i$. (See Appendix for proof.)*

In Step 1, CACA uses MRemove to reduce G as much as possible by MU removing. In Step 2, CACA uses MAssoc for MU re-association to minimize H. After the optimization by CACA, $H_i = G_i + \sum_{j\in D} \gamma^I_{ij} d_{ij}$.

Theorem 2. *CACA is a $(4+\epsilon)$ approximation algorithm. (See Appendix for proof.)*

We next briefly analyze the time complexity of CACA. In CACA, the MU removing sub-problem for each AP i is reduced to IMKP(a,b,V). The time complexity of solving IMKP is $O(nV)$, where n is the number of items and V is the target sum value. In the IMKP we constructed, n equals to the number of MUs and V is bounded by the maximum AP load. MRemove iterates $\log_2^{(ub-lb)/(lb\epsilon)}$ times. In each iteration, it solves m knapsack problems. Therefore, the complexity of MRemove is $O(\log_2^{(ub-lb)/(lb\epsilon)} mnV)$. As the complexity of MRemove dominates that of MAssoc, the Complexity of CACA is still $O(\log_2^{(ub-lb)/(lb\epsilon)} mnV)$.

V. Illustrative Simulation and Experimental Results

In this section, we present our simulation and experimental results on CACA performance. We discuss our simulation environment and performance metrics in Section V-A, and illustrative simulation results in Section V-B. Experimental validation is covered in Section V-C.

A. Simulation Environment

We conduct simulation on packet-level simulator NS3 to evaluate CACA. In our simulation, APs are randomly deployed in an area (of size 400m $\times$ 400m). To bring heterogeneity to the network, some nodes are equipped with multiple antennas. Some randomly selected nodes are equipped with only one single antenna. We create Communication links connecting two randomly chosen nodes. The Sender and the receiver are assigned randomly.

We expect that even if users are uniformly distributed, user associations under strongest signal association rule are non-uniformly distributed around APs due to random AP placement. For example, APs deployed on the border of the area cover less users, thus have less associated users. In the simulation, an AP sends flows with varying traffic demand to its associated users. We randomly generate current association. CACA migrates users based on current association.

Unless otherwise stated, we use the following as our baseline parameters. We use log-distance path loss model with reference distance 1 meter, reference loss 46.678 dbm and loss exponent 3. Transmit power is set to be 20 dbm. Traffic demand per user is 3.3 Mbps. The number of APs and MUs are 20 and 100 respectively. We consider each user has homogeneous migration cost. CACA migrates no more than 25% of the total number of users to optimize the association.

We are interested in the following performance metrics: 1) *Throughput:* It is user throughput, calculated as the number of bits received divided by transmission duration. 2) *Loss rate (UDP):* It is the ratio between the number of received packets and the number of transmitted packets. 3) *Delay:* It is defined as the end-to-end delay of successfully received packets. 4) *Migration cost:* It is defined as the number of migrated users. We compare CACA with the following approaches:

- *Unconstrained optimization:* It maximizes the worst-case user throughput via global optimization.
- Strongest signal based association (SSA): In the scheme, mobile users always associate with the AP from which they receive the best signal.
- *Proportional fairness association (PFairness) [3]:* In the scheme, user association is optimized by an offline algorithm to achieve proportional fairness in user throughput. It does not consider user migration cost.
- *Game-theoretic association (GameBased) [23]:* In the scheme, association is modeled as a selfish game. Each player in the game aims to maximize its achievable throughput. As players will not select a heavily loaded AP, it addresses the load balancing issue. However users are free to switch between APs, handover cost is overlooked.

B. Illustrative Simulation Results

We plot the worst-case user throughput versus different traffic demand in Figure 2. The worst-case user throughput of both CACA and unconstrained optimization flats off when traffic demand reaches network capacity. The achievable network capacity of CACA is close to that of unconstrained optimization. It justifies that CACA achieves close-to-optimum performance and reduces a substantial amount of migration cost at the same time.

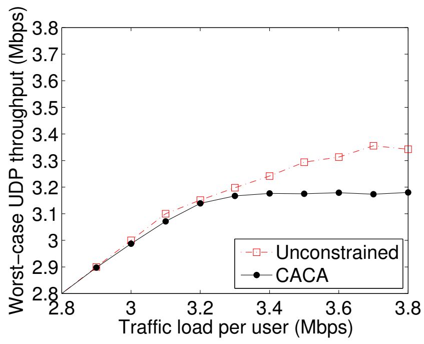

Fig. 2. Worst-case loss rate vs. traffic demand.

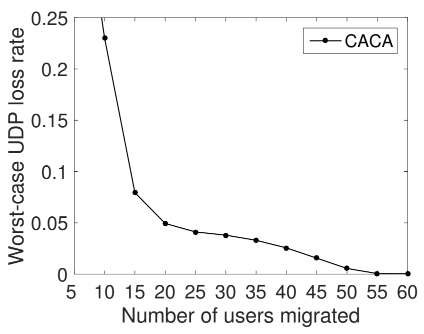

Fig. 3. Worst-case loss vs. num. of migrated users.

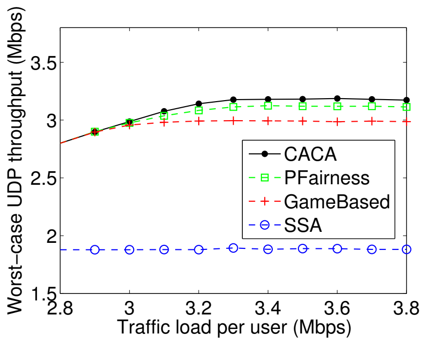

Fig. 4. Worst-case throughput vs. traffic demand.

We study the UDP loss rate of the worst-case user with respect to the number of migrated users in Figure 3. The loss rate drops as we migrate more mobile users because CACA reduces the maximum AP load better by migrating more users. The loss rate converges as the number of migrated users increases, showing that there is little incremental benefit to migrate many users beyond certain number. This is because there are a small number of crucial users whose migration can substantially improve the network performance. This also justifies that CACA can achieve similar performance as the unconstrained scheme by only migrating the most critical users.

In Figure 4, we compare worst-case user throughput of CACA under different traffic requirement with other comparison schemes. User throughput first increases with traffic demand, then flats off as the traffic saturates the network. CACA outperforms both PFairness and GameBased. PFairness performs much worse than its theoretical performance. This is because PFairness assumes that PCF (Point Coordinating Function) of WLANs is enabled. However, in both NS3 and realistic WLANs, DCF rather than PCF is used for media access control. GameBased performs worse than PFairness because it is a distributed scheme that suffers from local optimum.

We show the cumulative distribution function (CDF) of UDP loss rate of different schemes in Figure 5. CACA achieves overall low loss rate, which is substantially lower than Gamebased. A small portion of users in PFairness enjoy lowest loss rate. However, the majority of users in PFairness suffer from higher loss than CACA. This is because Proportional Fairness is the same as the time based fairness. It can make good use of AP throughput if PCF is enabled. On the other hand, CACA achieves throughput fairness by maximizing the worst-case user throughput. Clearly, CACA is a better option for WLANs based on DCF.

We show in Figure 6 worst-case TCP delay versus traffic load. As expected, delay increases with traffic load due to more congestion. The delay of CACA and that of PFairness increase linearly, while the delay of SSA increases drastically. Without AP load balancing, the congestion issue of SSA is much more severe. CACA manages to achieve low delay performance while keeping the migration cost low. The delay performance of GameBased is also poor due to local optimum.

Figure 7 plots the number of migrated users by different schemes versus total number of users in the network. The number of migrated users increases with network size. CACA manages to keep the migration cost low by cost constraint. However, the number of migrated users in PFairness and Unconstrained optimization increases sharply with network size. This shows that without constraining the migration cost, association re-optimization is likely to cause a large number of user migrations. GameBased is fully distributed. Even though it does not require communication with central controller, it indeed incurs a substantial number of control frames in the network due to a large number of MU migrations.

We finally show in Figure 8 the number of migrated users versus total number of APs in the network. The migration cost increases drastically with the number of APs in the network. This is because each MU has more candidate APs as we deploy more APs in the network. The migration cost of CACA keeps low as we put a constraint on the total number of migrations. It is true that one can get a better association by migrating more MUs. However, switching around neighbouring APs brings little performance gain.

C. Experimental Validation

CACA is simple and implementable. In this section, we validate its performance by implementing and testing it in a testbed. As shown in Figure 9, we deploy three 802.11n APs in a 7 meters times 10 meters room. Each AP is implemented by running a hostapd daemon on a Qualcomm JA76PF0 development board. To eliminate interference effect, APs are set to operate on channel 1, 6 and 11 respectively on 2.4GHz frequency band. APs are labelled by A, B and C. We also place five mobile phones running Android close to AP B. According to strongest signal association rule, all mobile phones initially associate with AP B.

The central controller is run on a PC in the same Ethernet with APs. We develop an android APP and install it on each mobile phone. The APP communicates with the central controller. When the central controller detects there is a need for association re-optimization, it requests the APP on migrating phones to perform scanning. To enforce a mobile phone to associate with a specific AP, the APP sets the "BSSID" field of Android class "WifiConfiguration" to be the MAC address of the AP. During the experiment, all mobile phones transmit saturated UDP traffic to a PC in the same Ethernet with APs.

Figure 10 shows the achieved UDP throughput of each mobile phone versus the number of migrated users. User throughput keeps improving as we migrate more users. The marginal throughput gain of the worst-case user decreases with number of migrations. We can see that the user throughput

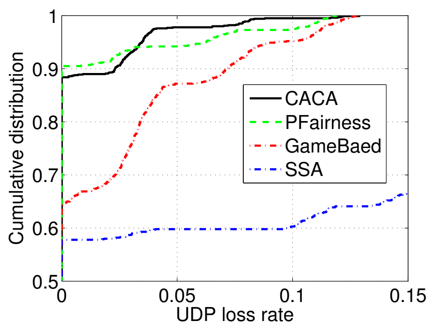

Fig. 5. Cumulative loss rate distribution.

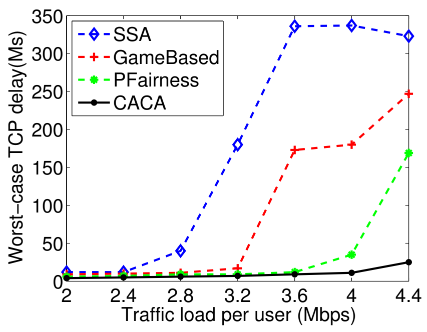

Fig. 6. Delay vs. traffic demand.

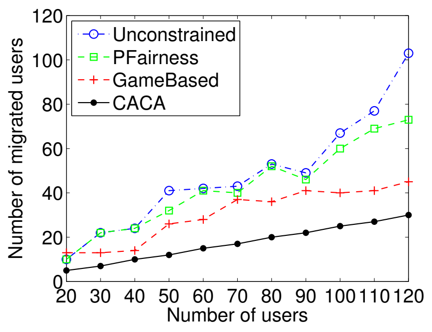

Fig. 7. Number of migrated users vs. number of users.

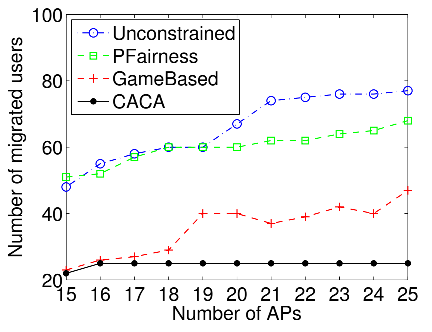

Fig. 8. Number of migrated users vs. number of APs.

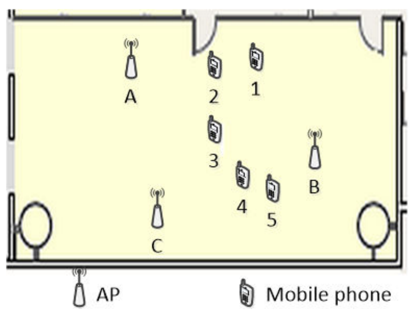

Fig. 9. Locations of APs and mobile phones for the 802.11n testbed.

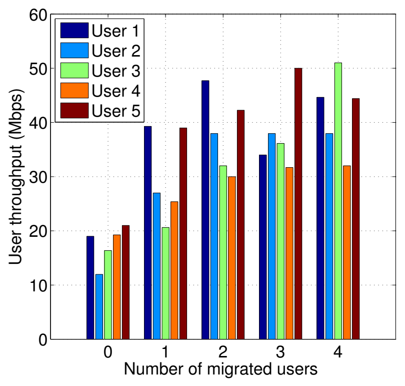

Fig. 10. Throughput vs. number of migrated users.

becomes quite stable after migrating less than half of the total number of users (in our study, 2 users). Therefore, a large number of user migrations is not necessary in reality.

VI. Conclusion

As users join or leave in a WLAN network, AP load may become unbalanced, affecting user throughput. The network throughput can be improved by re-associating users to other APs. As such migration incurs cost, we study the optimization of AP re-association subject to a certain total migration cost constraint in this paper. Given that most WLANs are based on DCF MAC, we study max-min user throughput model as our objective. There has not been sufficient consideration on migration cost, and its introduction as a constraint fundamentally changes the nature of the problem.

We first formulate the problem, and show that it is NP-hard. We present CACA, a simple and efficient approximation algorithm for AP re-association. We prove that CACA achieves $(4+\epsilon)$ approximation. Our extensive simulation results show that CACA efficiently migrates mobile users to substantially improve user throughput. It is closely optimal, i.e., its throughput is close to the case without migration cost constraint. We have implemented CACA. Our experimental results further validate its effectiveness.

References

[1] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley, "Facilitating access point selection in IEEE 802.11 wireless networks," in *Proc. 5th ACM SIGCOMM conf. Internet Meas.*, 2005, pp. 26–26.

[2] Y. Bejerano, S.-J. Han, and L. E. Li, "Fairness and load balancing in wireless lans using association control," in *Proc. ACM MobiCom*, 2004, pp. 315–329.

[3] W. Li, S. Wang, Y. Cui, X. Cheng, R. Xin, M. Al-Rodhaan, and A. Al-Dhelaan, "Ap association for proportional fairness in multirate WLANs," *IEEE/ACM Trans. Netw*, vol. 22, no. 1, pp. 191–202, 2014.

[4] R. Raghavendra, E. M. Belding, K. Papagiannaki, and K. C. Almeroth, "Understanding handoffs in large IEEE 802.11 wireless networks," in *Proc. 7th ACM SIGCOMM conf. Internet Meas.*, 2007, pp. 333–338.

[5] A. Mishra, M. Shin, and W. Arbaush, "Context caching using neighbor graphs for fast handoffs in a wireless network," in *Proc. IEEE INFOCOM*, vol. 1, 2004.

[6] M. Shin, A. Mishra, and W. A. Arbaugh, "Improving the latency of 802.11 hand-offs using neighbor graphs," in *Proc. ACM MobiSys*, 2004, pp. 70–83.

[7] I. Ramani and S. Savage, "SyncScan: practical fast handoff for 802.11 infrastructure networks," in *Proc. IEEE INFOCOM*, vol. 1, 2005, pp. 675–684.

[8] M. Kim, Z. Liu, S. Parthasarathy, D. Pendarakis, and H. Yang, "Association control in mobile wireless networks," in *Proc. IEEE INFOCOM*, 2008.

[9] F. Xu, C. C. Tan, Q. Li, G. Yan, and J. Wu, "Designing a practical access point association protocol," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[10] S. Miyata, T. Murase, and K. Yamaoka, "Novel access-point selection for user QoS and system optimization based on user cooperative moving," *IEICE T COMMUN*, vol. 95, no. 6, pp. 1953–1964, 2012.

[11] F. Xu, X. Zhu, C. C. Tan, Q. Li, G. Yan, and J. Wu, "Smartassoc: Decentralized access point selection algorithm to improve throughput," *IEEE Trans. Parallel Distrib. Syst.*, p. 1, 2013.

[12] D. Gong and Y. Yang, "Ap association in 802.11 n wlans with heterogeneous clients," in *Proc. IEEE INFOCOM*, 2012, pp. 1440–1448.

[13] L. Li, M. Pal, and Y. R. Yang, "Proportional fairness in multi-rate wireless LANs," in *Proc. IEEE INFOCOM*, 2008.

[14] M. Hong, A. Garcia, J. Barrera, and S. G. Wilson, "Joint access point selection and power allocation for uplink wireless networks," *IEEE Trans. Signal Process.*, vol. 61, no. 13, pp. 3334–3347, 2013.

[15] G. Xue, Q. He, H. Zhu, T. He, and Y. Liu, "Sociality-aware access point selection in enterprise wireless lans," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 10, pp. 2069–2078, 2013.

[16] M. Y. Arslan, K. Pelechrinis, I. Broustis, S. Singh, S. V. Krishnamurthy, S. Addepalli, and K. Papagiannaki, "Acorn: An auto-configuration framework for 802.11 n wlans," *IEEE/ACM Transactions on Networking*, vol. 21, no. 3, pp. 896–909, 2013.

[17] M. Hong, A. Garcia, and J. Barrera, "Joint distributed access point selection and power allocation in cognitive radio networks," in *Proc. IEEE INFOCOM*, 2011, pp. 2516–2524.

[18] L.-H. Yen, J.-J. Li, and C.-M. Lin, "Stability and fairness of AP selection games in IEEE 802.11 access networks," *IEEE Trans. Veh. Technol.*, vol. 60, no. 3, pp. 1150–1160, 2011.

[19] Z. Chen, Q. Xiong, Y. Liu, and C. Huang, "A strategy for differentiated access service selection based on application in WLANs," in *IEEE INFOCOM WKSHPS*, 2014, pp. 317–322.

[20] X. Chen, W. Yuan, W. Cheng, W. Liu, and H. Leung, "Access point selection under QoS requirements in variable channel-width WLANs," *IEEE WCL*, vol. 2, no. 1, pp. 114–117, 2013.

[21] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New insights from a fixed point analysis of single cell ieee 802.11 wlans," in *Proc. IEEE INFOCOM*, vol. 3. IEEE, 2005, pp. 1550–1561.

[22] J. K. Lenstra, D. B. Shmoys, and É. Tardos, "Approximation algorithms for scheduling unrelated parallel machines," *Mathematical programming*, vol. 46, no. 1-3, pp. 259–271, 1990.

[23] L.-H. Yen, J.-J. Li, and C.-M. Lin, "Stability and fairness of native AP selection games in IEEE 802.11 access networks." in *Proc. IEEE WOCN*, 2010, pp. 1–5.

APPENDIX

Proof of Lemma 2 (See Page 5.): Given any achievable g, we first prove that I outputted by MRemove is the optimal subset achieving g with the least cost. We show this by contradiction. To achieve the objective value g, we have to reduce the load of each AP i. Therefore, we have an MU removing subproblem for each AP i. The problem is to remove a subset of $U(i)$ to reduce L_i by at least $\max\{L_i - g, 0\}$ while inducing the least cost. This problem reduces to IMKP via the construction shown in Line 6 of Algorithm (1). The corresponding IMKP of each AP i is solved optimally by IMKA and I_i is the optimal set to remove. $I = \bigcup_i I_i$. The removal of I_i incurs B_i cost. The total cost incurred by removing I is hence $B = \sum_i B_i$.

Let us assume that I is not optimal. Then, there must be another subset $I^{'}$ whose removal incurs $B^{'} = \sum_i B_i^{'}$ cost. $B^{'} \leq B$. Similarly, $I_i^{'}$ with removing cost $B_i^{'}$ is the set of MUs removed from i. $B^{'} \leq B$ implies that there exists an AP i such that $B_i^{'} \leq B_i$. However, we have shown that B_i is the optimal value, a contradiction.

At the termination of MRemove, ub is set to a value that can be achieved by MRemove within the cost budget K. This is true because only achievable g will be set as ub according to Algorithm (1).

The lower bound lb is always not achievable by removing any feasible subset of MUs. Therefore, $G^* \geq lb$. When MRemove terminates, $ub < (1+\epsilon)lb$. We hence have $G \leq ub \leq (1+\epsilon)lb \leq (1+\epsilon)G^*$. This completes the proof. □

Proof of Lemma 3 (See Page 5.): We know that D^* is the optimal solution of MRP (MU Removing Problem). The removal of D^* results in optimal value G^*. We denote the set of MUs migrated by the optimal solution of CACA as $D^{'}$. Let $G^{'}$ be the load of the heaviest AP after $D^{'}$ is removed. Since G^* is the optimal value, $G^* \leq G^{'}$. Furthermore, OPT is the heaviest AP load after re-associating MUs in set $D^{'}$. Therefore, we have $OPT \geq G^{'} \geq G^*$. This completes the proof. □

Proof of Lemma 4 (See Page 6.): In MAssoc, we use binary search to find the smallest value g^* such that FMRAP(g^*) has a feasible solution. Clearly, $g^* \leq H^*$ (because FMRAP is the LP Relaxation of MRAP). Every lb that we set in MAssoc is infeasible. Therefore, $lb < g^*$. When the MAssoc terminates, $ub < (1+\epsilon)lb$. γ^I, the output of MAssoc is obtained from rounding the fractional solution of FMRAP(ub). According to Theorem 1, $\sum_{j\in D}\gamma_{ij}^I d_{ij} \leq 2(ub - G_i), \forall i$. Therefore, $\sum_{j\in D}\gamma_{ij}^I d_{ij} \leq 2(ub - G_i) \leq 2((1+\epsilon)lb - G_i) \leq 2((1+\epsilon)g^* - G_i) \leq 2(1+\epsilon)H^* - 2G_i$. □

Proof of Theorem 2 (See Page 6.): Please recall that L^* and x^* denote the optimal value and optimal solution to the MinmaxLP (Definition 3) respectively. We define the following auxiliary problem.

$$\min F$$
$$\sum_{j\in U}\gamma_{ij}d_{ij} \leq F - G_i, \forall i, \tag{12}$$
$$\sum_{i:i\in A(j)}\gamma_{ij} = 1, \forall j \in U \tag{13}$$
$$\gamma_{ij} \in \{0,1\}$$

Let F^* be the optimal value of the auxiliary problem. The auxiliary problem is similar to the MRAP except that the constraint (13) in auxiliary problem is for all $j \in U$. The auxiliary problem is to re-associate all users in U and MRAP is to re-associate users in D. Since $D \subseteq U$, we have $H^* \leq F^*$.

$$\begin{aligned}
H &= \max_i \left(\sum_{j\in D}\gamma_{ij}^I d_{ij} + G_i\right) && (14)\\
&\leq \max_i(2(1+\epsilon)H^* - G_i) && (15)\\
&\leq 2(1+\epsilon)H^* && (16)\\
&\leq 2(1+\epsilon)F^* && (17)\\
&\leq 2(1+\epsilon)\max_i\left(\sum_{j\in U}x_{ij}^* d_{ij} + G_i\right) && (18)\\
&\leq 2(1+\epsilon)\left(\max_i(\sum_{j\in U}x_{ij}^* d_{ij}) + \max_i(G_i)\right) && (19)\\
&\leq 2(1+\epsilon)L^* + 2(1+\epsilon)(1+\epsilon)G^* && (20)\\
&\leq 2(1+\epsilon)L^* + 2(1+\epsilon)(1+\epsilon)OPT && (21)\\
&\leq 2(1+\epsilon)OPT + 2(1+\epsilon)(1+\epsilon)OPT && (22)\\
&\leq (4+6\epsilon+2\epsilon^2)OPT, && (23)\\
&\leq (4+\epsilon)OPT, && (24)
\end{aligned}$$

(replacing $6\epsilon + 2\epsilon^2$ by ϵ as they both are small numbers)

where (15) is due to Lemma 4. Clearly, x^* is a feasible solution to the auxiliary problem and the objective value we get by plugging x^* into the Auxiliary problem is $\max_i\left(\sum_{j\in U}x_{ij}^* d_{ij} + G_i\right)$. F^* is the optimal value. Therefore, we get (18) from (17). We have (20) due to Lemma 2 and (21) due to Lemma 3. We have (22) due to Lemma 1. □