Traffic-Aware Dynamic Spectrum Access

Lei Yang, Lili Cao, Heather Zheng, Elizabeth Belding
Department of Computer Science, University of California, Santa Barbara, CA 93106
leiyang,lilicao,htzheng,ebelding@cs.ucsb.edu

Abstract— **Demand-driven spectrum allocation can drastically improve performance for WiFi access points struggling under increasing user demands. While their frequency agility makes cognitive radios ideal for this challenge, performing adaptive spectrum allocation is a complex and difficult process. In this work, we propose FLEX, an efficient spectrum allocation architecture that efficiently adapts to dynamic traffic demands. FLEX tunes network-wide spectrum allocation by access points coordinating with peers, minimizing network resets through local adaptations. Through detailed analysis and experimental evaluation, we show that FLEX converges quickly, provides users with proportional-fair spectrum usage and significantly outperforms existing spectrum allocation proposals.**

I. Introduction

As WiFi networks become a pervasive last mile connectivity tool, WiFi users are suffering from poor performance at crowded hotspots. Take for example recent experiences at ACM conferences such as SIGCOMM and MobiCom. When more than 200 attendees compete for wireless access on the same network, each user obtains minimal bandwidth barely enough to support email access. As bandwidth-hungry devices such as AppleTV and iPhone join the fray, these frustrating experiences will become an increasingly common part of our daily lives. The fundamental observation is that increasing user participation leads to greater variability in traffic density and demands, and consequently more unpredictable user experiences.

To improve the user experience, WiFi access points (AP) must adjust their allocated bandwidth based on varying traffic demands. Existing networks exploit MIMO techniques to improve AP bandwidth, but the improvement is limited by the number of antennas. On the other hand, varying APs' spectrum allocation would be a natural and highly effective approach. Unfortunately, current WiFi networks cannot exploit this approach because of the following restrictions:

- Each AP's spectrum usage is fixed by its radio hardware. Because a WiFi radio can only access one channel at any time, each AP's spectrum usage is defined by the number of radios equipped, typically less than 2 in practical deployments.
- Each AP's channel is statically assigned, but its traffic demand varies across time. Upon a meeting or a class, a lightly-loaded AP will become a hotspot within a few minutes. More importantly, network measurements have shown that traffic dynamics are unpredictable [8], making static channel planning infeasible.

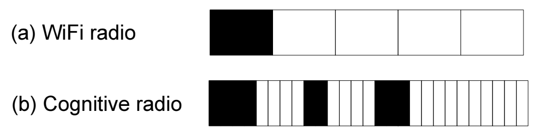

Fig. 1. Comparing WiFi and CR radios. CRs can use multiple channels.

Lacking the flexibility to adapt spectrum usage, WiFi networks face great challenges in providing satisfactory user performance. Cognitive radios (CR) are the ideal solution to this problem. Unlike WiFi radios, CRs can access spectrum flexibly. CRs partition spectrum into a large number of orthogonal channels, and transmit using a flexible set of channels simultaneously as shown in Figure 1. Existing CR testbeds and on-going research have demonstrated the hardware feasibility of this technique [5], [18]. By adjusting the amount and the frequency location of spectrum usage, CRs can adapt their bandwidth on-the-fly.

Intuitively, user performance in WiFi networks can be significantly improved by equipping APs and end-devices with CRs. APs can quickly adapt their spectrum usages to varying traffic demands while minimizing the interference with nearby peers. The fundamental challenges are how to determine *which* and *how many* channels each AP should use to *maximize user satisfaction*, and how to adapt AP spectrum usages to their dynamic traffic demands. Prior work [1], [20] develops centralized algorithms to allocate each CR a spectrum block of variable size based on traffic demands. Such centralized architecture, while simplifying algorithm design, can suffer from significant adaptation delay and communication cost in large-scale wireless networks. To cope with fast-growing WiFi deployments, APs should also determine their spectrum allocations in a timely manner, requiring minimum management.

In this paper, we propose a distributed approach for CR-equipped APs to dynamically access spectrum based on their traffic demands. Our design considers the following three primary goals:

- Quickly adapting to varying traffic demands.
- Capable of maximizing user proportional-fairness.
- Distributed, scalable, requiring minimum management.

More specifically, we present FLEX, a distributed architecture for APs to adapt spectrum usage. Using FLEX, APs coordinate with peers and iteratively apply local improvements to tune network-wide spectrum allocation. Upon traffic dynamics, instead of performing another round of global optimization, only affected APs and their close neighbors perform local improvements. These local actions occur in parallel across

Digital Object Identifier: 10.4108/ICST.WICON2008.4953
http://dx.doi.org/10.4108/ICST.WICON2008.4953

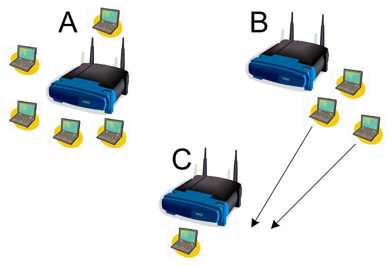

Fig. 2. An example scenario which shows the benefits of fast traffic-aware adaptation. At time T_0, two wireless users move from AP B to AP C.

the network, resulting in very low adaptation delay. We demonstrate analytically and show experimentally that this new architecture quickly adapts to network dynamics and significantly outperforms priori spectrum allocation proposals.

Our key contributions are of three-fold:

- We propose a novel distributed algorithm that applies local improvements to tune network-wide spectrum allocation considering traffic demands. We prove its convergence and show that it provides guaranteed spectrum usage.
- We implement the FLEX algorithm using a light-weight coordination protocol and analyze the adaptation delay. APs coordinate in the background without disrupting existing user activity, and utilize spatial parallelism to reduce adaptation delay.
- We perform extensive experiments to evaluate FLEX. Compared to prior proposals, FLEX improves the spectrum utilization and fairness by more than 30%. FLEX quickly adapts to traffic variations, requiring only 2-6 seconds for a large WiFi network of 400 access points.

II. Motivation and Problem Model

FLEX targets WiFi deployments in large corporate or city campuses, which place APs densely to provide seamless connectivity. As user activity changes, each AP's traffic demand varies unpredictably [8]. The fundamental problem is how to allocate spectrum channels to APs based on their traffic demands to maximize user satisfaction while minimizing conflict. In this section, we demonstrate the benefits of fast traffic-aware adaptation and formally present the FLEX problem model. We will present the FLEX distributed allocation algorithm in Section III and implement it as the FLEX coordination protocol in Section IV.

A. Benefits of Fast Traffic-Aware Adaptation

To demonstrate the benefits of fast traffic-aware spectrum allocation and adaptation, we consider the following example. As shown in Figure 2, three APs A, B and C with different numbers of associated users are within the interference range of each other. We assume there are $M = 9$ channels and all users have the same traffic demand.

Without considering traffic heterogeneity, each AP should get the same amount of the spectrum: 3 channels. Since each AP equally divides its spectrum to its associated users in time, individual users associated with A, B and C get 0.6, 1, and 3 channels, respectively. Such unfairness among users can lead to either user starvation or wasted spectrum resource. Being traffic-aware, the system allocates 5 channels to A, 3 channels to B, and 1 channel to C, so that each user gets the same amount of spectrum: 1 channel. Therefore, being traffic-aware can significantly improve user fairness and hence satisfaction.

Fast adaptation is critical for traffic-aware spectrum allocation. Since wireless networks are dynamic in natural, each AP's traffic load varies quickly over time. Only if spectrum allocations adapt quickly to traffic dynamics, can the benefits of traffic-aware allocation be achieved. We show this using the same example in Figure 2: at time T_0, two users disassociate with B and associate with C, changing the traffic loads of B and C significantly. Unless AP spectrum allocations adapt quickly to this new traffic load, the user of B gets 3 channels while others get $1/3$ channel.

Fast adaptation in large-scale wireless networks, however, is challenging given the network scale. For example, metropolitan wireless networks have thousands of APs concentrated in a small area. To achieve fast traffic-aware spectrum allocation in this type of networks, we need a distributed solution that has minimum management overhead, and adapts quickly to traffic dynamics.

B. Problem Model

We formally define the traffic-aware spectrum allocation as a combinatorial optimization problem. For simplicity, we describe the problem model and FLEX design within the context of binary pairwise interference condition. In this case, we group each AP and its users into a super-node, and two super-nodes either conflict and cannot use the same channel concurrently or do not conflict. We also assume channels are homogeneous with the same bandwidth and interference property. Our algorithm and analytical conclusions can be extended to complex interference and channel conditions, as discussed in Section VI.

We first define the following notations:

Nodes We group each AP and its users into a single node i, $i \in [1, N]$, where N is the number of APs in the campus.

Channels The spectrum is divided into a large set (M) of orthogonal channels*, indexed 1 to M.

Interference Constraints We model the interference condition as a binary metric between any two nodes n, k:

$$c_{n,k} = \begin{cases} 1, & \text{node } n \text{ and } k \text{ conflict with each other} \\ 0, & \text{node } n \text{ and } k \text{ can reuse the same channel.} \end{cases}$$

Neighbors Node k is the "neighbor" of node n if $c_{n,k} = 1$.

Conflict-free Channel Allocation We represent an allocation as

$$a_{m,n} = \begin{cases} 1, & \text{channel } m \text{ assigned to node } n \\ 0, & \text{otherwise.} \end{cases}$$

*Using multicarrier modulation with proper guardband, channels in cognitive radio networks are orthogonal [5], [18].

Digital Object Identifier: 10.4108/ICST.WICON2008.4953
http://dx.doi.org/10.4108/ICST.WICON2008.4953

Under the interference constraints, an allocation is conflict-free if $a_{m,n} \cdot a_{m,k} = 0,\ if\ c_{n,k} = 1,\ \forall\ n,k \in [1,N], m \in [1,M]$. Let $S_n = \sum_{m=1}^{M} a_{m,n}$ represent AP n's spectrum usage.

Fairness-driven Optimization FLEX maximizes user satisfaction by maximizing user *proportional fairness*. Let b_i represent the number of channels allocated to user i. Then the allocation vector $\mathbf{b} = (b_i, i \in I)$ is proportionally fair if for any other feasible vector $\mathbf{b}'$, the aggregate of the proportional changes is not positive:

$$\sum_{i \in I} \frac{b_i' - b_i}{b_i} \leq 0.$$

It was shown by [14] that the maximum proportional fairness is achieved by maximizing $\sum_{i \in I} \log(b_i)$. Assuming each AP n's bandwidth scales linearly with its spectrum usage S_n, and is divided equally among its associated users[†], we model each user's bandwidth as $b_i = S_n/t_n$ where t_n is the number of users associated to AP n. As a result, we can rewrite the proportional fairness metric as

$$U_{fair} = \sum_{n=1}^{N} t_n \log \frac{S_n}{t_n} = \sum_{n=1}^{N} t_n \log \sum_{m=1}^{M} a_{m,n} - \sum_{i=1}^{N} t_n \log(t_n)$$

For easy notation, we remove the second item in U_{fair} since it does not depend on the allocation $\{a_{m,n}\}$.

From the above, we can represent FLEX's spectrum allocation problem as

$$\begin{array}{ll} \text{Find} & \{a_{m,n}\}_{n \in [1,N], m \in [1,M]} \\ \text{Maximize} & \displaystyle\sum_{n=1}^{N} t_n \log\left(\sum_{m=1}^{M} a_{m,n}\right) \\ \text{Subject to} & a_{m,n} \cdot a_{m,k} = 0,\ if\ c_{n,k} = 1 \\ & \forall\ n,k \in [1,N], m \in [1,M]. \end{array} \quad (1)$$

This is a constrained non-linear optimization problem and has been shown to be NP-complete [9]. In large-scale WiFi networks, each AP's number of users (t_i) changes over time, which raises the challenge to quickly adapt the channel assignment to optimize the fairness. To deal with this challenge, FLEX introduces efficient heuristics to approximate the global optimum while minimizing the computation overhead.

III. FLEX Allocation Algorithm

Motivated by the idea in Section II, we design FLEX to allow APs to coordinate and quickly adapt to varying traffic demands. FLEX differs significantly from prior graph coloring schemes [2], [7], [10] that allocate channels to achieve a given spectrum usage while minimizing the number of channels used. Instead, FLEX targets typical WiFi scenarios:

Given M spectrum channels, how do APs determine which and how many channels to use in order to maximize user fairness while being conflict-free?

The critical challenge is how should APs determine a fairness-maximizing channel allocation with only local information. FLEX introduces a novel distributed strategy using

[†]For simplicity we assume users have equal traffic demands. In practice, APs can assign users with spectrum proportional to their traffic demands.

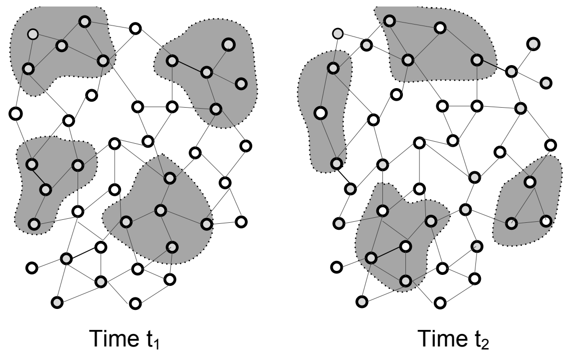

Fig. 3. FLEX local improvement framework – Nodes form local events with their conflicting neighbors recursively. FLEX schedules multiple local improvements in parallel to minimize adaptation delay.

iterative local improvements. Starting from an initial allocation, APs iteratively improve local spectrum usage towards an allocation with higher system utility among its current solution neighborhood. The process repeats until an efficient global allocation is found. More importantly, by regulating the format of local improvements, FLEX quickly leads the system to a stable state where each AP acquires spectrum usage that is lower-bounded.

A. Adaptation via Local Coordination

FLEX iteratively modifies local spectrum allocation to tune network-wide spectrum allocation. Define a local area that contains a set of nodes: $\mathbb{P} = \{n_1, n_2, ...n_P\}$ (shown as a shaded area in Figure 3). We can rewrite the optimization function in eq. (1) into:

$$\begin{aligned} U_{fair} &= \sum_{n \in [1,N] \setminus \mathbb{P}} t_n \log S_n + \sum_{n \in \mathbb{P}} t_n \log S_n \\ &= U(\setminus \mathbb{P}) + U(\mathbb{P}). \end{aligned} \quad (2)$$

A local improvement over $\mathbb{P}$ is to modify $\{\{a_{m,n}\}_{m \in [1,M]}\}_{n \in \mathbb{P}}$ (hereby referred to as $\{S_n\}_{n \in \mathbb{P}}$) to improve U_{fair}. By applying local improvements recursively across the network, FLEX gradually improves U_{fair} to approximate the global optimum.

The fundamental challenge is how to ensure each local improvement can improve U_{fair}. Because of interference, changes to $\{S_n\}_{n \in \mathbb{P}}$ may lead to conflicts at nodes outside $\mathbb{P}$ and degrade $U(\setminus \mathbb{P})$. Furthermore, it may lead to a subsequent set of local improvements across the area, destabilizing the system. FLEX eliminates such conflict by limiting the set of usable channels in $\mathbb{P}$, preventing any conflict to nodes outside $\mathbb{P}$. That is, nodes in $\mathbb{P}$ will not use any channel that is used by a conflicting neighbor inside and outside of $\mathbb{P}$. In this way, modifying $\{S_n\}_{n \in \mathbb{P}}$ can improve $U(\mathbb{P})$ without changing $U(\setminus \mathbb{P})$. Hence, any local improvement over $U(\mathbb{P})$ leads to a global improvement in U_{fair}.

FLEX keeps $\mathbb{P}$ small to simplify the local optimization procedure. The simplest format is a single node, $\mathbb{P} = \{n\}$, *i.e.* a node gets one more channel only if it is not used by any neighbors. This format, commonly found in cellular networks [11] to minimize voice blocking, is extremely limited in maximizing fairness for data services. We analyze the local improvement format and adopt a different strategy: ***Any node in need of spectrum can request its conflicting neighbors to give up some channels in order to maximize the local fairness***. Hence, each local improvement contains a node n and its immediate neighbors: $\mathbb{P} = \{n, \mathbb{N}(n)\}$ where $\mathbb{N}(n) = \{k\,|c(n,k) = 1\}$. Node n will modify $\{a_{n,m}\}_{n\in\mathbb{P},m=1..M}$ to maximize $U(\mathbb{P})$ while maintaining the same $U(\backslash\mathbb{P})$.

Our algorithm is supported by the following theorem, which shows that the system converges[‡] after a limited number of local improvements. Each AP obtains a guaranteed number of channels, proportional to the ratio of its traffic demand and those of its neighbors.

Theorem 1: The system will converge within a finite number of iterations. If APs have equal traffic demands, it will converge with an expected number of O(N^2) iterations. When the system converges, AP n's spectrum usage is lower bounded by

$$S_n > t_n \cdot (\left\lfloor \frac{M}{t_n + \sum_{k\in\mathbb{N}(n)} t_k} \right\rfloor - 1) \qquad (3)$$

When $M \gg t_n + \sum_{k\in\mathbb{N}(n)} t_k$, $S_n \geq \frac{t_n \cdot M}{t_n + \sum_{k\in\mathbb{N}(n)} t_k}$, referred to as the relaxed lower bound.

The proof is in Appendix A. Note that this bound is similar to the local fairness constraint defined in [13], [20] and in the classical weighted fair queuing problem [6]. However, the main difference is that this fairness expression is derived as a performance guarantee as the result of the global fairness optimization in (1). It is a *lower bound* on AP's spectrum usage, not a preset constraint.

B. Organizing Multiple Local Improvements

As traffic and network topology vary, many local improvements are required to tune network-wide spectrum allocation. As shown in Figure 3, FLEX schedules multiple local improvements to execute in parallel to minimize adaptation delay. FLEX implements these strategies as a coordination protocol executed by each AP. We will present the protocol design and analyze its performance in Section IV.

IV. Using FLEX in Practice: An Access Point Coordination Protocol

Aside from its algorithmic advantage, we show that FLEX can be implemented in practice as a coordination protocol. In corporate or city WiFi campuses, APs are cooperative and execute the FLEX coordination protocol to adapt spectrum allocation to traffic dynamics and maximize user satisfaction. APs exchange spectrum usage and traffic load information with peers periodically to identify sub-optimality in spectrum allocation. Upon detecting a sub-optimal allocation, APs communicate with neighboring peers to apply FLEX local improvement. We note that APs coordinate in the background without disrupting existing users communications. Upon identifying an efficient allocation, APs inform users to modify their spectrum usage. The primary goal of our protocol design is to minimize the system-wide adaptation delay.

We assume APs exchange coordination information through a dedicated control radio, which is widely used in existing CR testbeds [16], [19]. The protocol sits on top of the MAC and produces coordination messages as application packets. For simplicity, we describe the protocol assuming APs can reliably identify and directly communicate with their conflicting neighbors *i.e.* $\mathbb{N}(n)$. We discuss practical implications of these assumptions in Section VI.

A. Executing A Local Improvement Event

To perform a local improvement over $\mathbb{P} = \{n, \mathbb{N}(n)\}$, APs execute the following procedures.

- ***Request*** – AP n initiates the local improvement by broadcasting a "request" to its conflicting neighbors. AP n initiates a local improvement if 1) it identifies conflicts in channel usage with its neighbors $\mathbb{N}(n)$, or 2) it identifies sub-optimality in local spectrum allocation.
- ***Acknowledgement*** – Upon receiving a request, an AP responds with an ACK if it is not currently participating in any local improvement event, otherwise a NACK.
- ***Announcement*** – Without collecting all the $|\mathbb{N}(n)|$ ACK acknowledgements within a certain time, AP n cancels the coordination event. Otherwise, n will execute a local improvement and broadcast the local allocation adjustment to its neighbors.
- ***Improvement and Release*** – Upon receiving the announcement, nodes in $\mathbb{P}$ will record their new spectrum allocation and release themselves from the current local improvement event.

Figure 4 presents the state transition diagram for APs running the FLEX protocol. Each AP transits among "idle", "initiator" and "responder". To avoid conflict, FLEX requires that at any given time, each node can only participate in one local improvement event.

We can derive the duration of a local improvement event as the sum of these four stages:

$$T = T_{request} + T_{ack} + T_{announce} + T_{release}$$

where $T_{request}$ and $T_{announce}$ are the time to broadcast a single coordination packet, T_{ack} is the time to collect acknowledgements from multiple neighbors and $T_{release}$ is a small constant factor. The value of T depends significantly on the underlining MAC protocol. Using a 802.11 MAC protocol, the dominating component is T_{ack} because multiple neighbors of n will contend to send back acknowledgements which may collide with each other. In Section IV-C we analytically evaluate the delay.

B. Organizing Multiple Coordination Events

The system's adaptation delay represents the total time required for the FLEX algorithm to converge, *i.e.* no AP initiates

[‡]The system converges when no local improvement of the given format can improve the system utility.

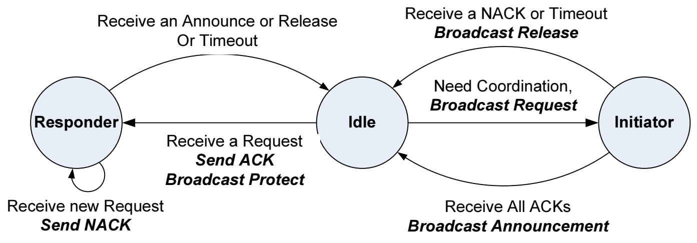

Fig. 4. FLEX State Transition Diagram

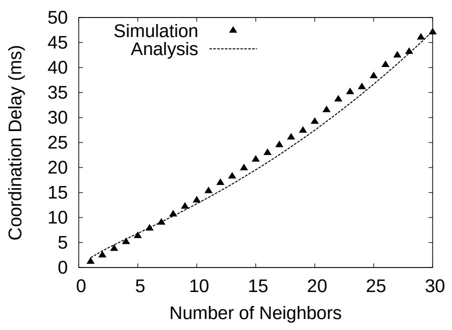

Fig. 5. Time needed to perform a local improvement in a circular topology.

local improvement. To minimize the adaptation delay, FLEX allows multiple local improvements to execute in parallel. The example in Figure 3 shows that at time t_1, 4 local improvement events execute in parallel without any conflict. This is achieved naturally using local requests.

On the other hand, while maximizing parallelism, the FLEX protocol also places conflicting coordination events to different times. Because APs only have local view of the network, conflicts will likely occur when multiple local improvements contend for transmission medium or an AP. For example, when APs near an active local improvement event start to initiate new events, their coordination packets will disrupt existing coordinations. Similarly, when neighboring APs initiate new events simultaneously, their request packets contend. Even if their requests were sent successfully, they are likely to target common neighbors. Because each node can only ACK to one event, most requests fail, leading to unnecessary waste of resource and higher coordination delay.

The FLEX protocol minimizes conflicts by organizing the timing of local improvement events. First, to minimize contention among requests, APs apply a *request backoff* to randomize their timing. Using an exponential increase mechanism similar to the 802.11 protocol, APs double request backoff window when its request fails. Second, to minimize disruption to active events, APs broadcast a *protection* packet immediately before responding ACK to a request. The packet carries a *protection period* (T_P) field that estimates the time required to finish the current event, $T_P = T_{ack} + T_{announce} + T_{release}$. Neighbors use this information to delay their future requests/coordinations to the estimated finish time. Finally, APs use *timeouts* to release themselves from unsuccessful local improvement events. APs in the initiator state estimate the maximum waiting time for acknowledgements; APs in the responder state estimate the maximum waiting time for announcements.

C. Analytical Models of T_P and T

As discussed earlier, the dominating component in T_P (and T) is T_{ack}. In this section, we show that we can analytically derive T_{ack} as a function of $|\mathbb{N}(n)|$, assuming the control radio uses the 802.11 MAC protocol.

To model the random backoff and contention among $\mathbb{N}(n)$ nodes who respond to the coordination request, we adopt the method in [3]. Let t_v represent the time gap between two consecutive successful transmissions. From [3],

$$t_v = E[N_c](E[Coll]+\tau+DIFS)+E[Idle](E[N_c]+1)+E[S]$$

where $E[N_c]$ is the average number of collisions within t_v, $E[Coll]$ is the average collision length, $E[S]$ is the average successful transmission time, τ is maximum propagation delay and $E[Idle]$ is the average number of idle slots within t_v. For the FLEX protocol, we can compute them as [3]:

$$\begin{aligned} E[N_c] &= \frac{1-(1-p)^K}{Kp(1-p)^{K-1}} - 1 \\ E[Coll] &= \overline{m}, \qquad E[Idle] = \frac{(1-p)^K}{1-(1-p)^K} \cdot t_{slot} \\ E[S] &= \overline{m} + 2\cdot\delta + SIFS + ACK + DIFS \quad (4) \end{aligned}$$

where δ is the packet transmission delay, K is the number of transmitting nodes, and $\overline{m}$ is the MAC transmission time for FLEX acknowledgement packets. Using these results, we can compute the average time required for $\mathbb{N}(n)$ nodes to acknowledge the initiator: $E[T_{ack}(\mathbb{N}(n)] = \sum_{K=1}^{|\mathbb{N}(n)|} t_v(K)$.

In Figure 5 we verify the proposed model using Qualnet simulations. We place X nodes around a center node n (with the same distance to n). Each node needs to send an ACK to n. We measure the average time required for n to collect all the ACKs and compare them to the estimated T_{ack}. Results show that the analytical result closely approximate the simulated protocol performance. Hence, APs can use this analytical result to optimize the request backoff window size, the protection period and the values of timeouts.

V. Experiments

In this section, we evaluate FLEX in terms of its allocation algorithm and coordination protocol. We implement the FLEX coordination protocol using the Qualnet simulator. We examine the adaptation delay of FLEX's coordination protocol using different network and traffic scenarios.

A. FLEX Protocol Performance

We evaluate the protocol performance by measuring:

(1) System convergence time which is the time required to finish the last local improvement. It represents the adaptation delay over traffic dynamics.

(2) Coordination overhead which is the total number of local improvement events, including both successful and failed ones. It estimates the protocol communication overhead.

Because the coordination delay depends heavily on the number of conflicting neighbors per AP, we first examine the protocol performance assuming a grid topology where APs have 4 and 8 conflicting neighbors, respectively. To examine the protocol scalability, we expand the grid size to increase the number of APs while keeping the same number of conflicting neighbors. We also examine the impact of random topologies where APs are placed randomly on the grid with up to 4 and 8 neighbors. We assume each data channel provides 1Mbps bandwidth in average.

We assume APs can identify and communicate directly with conflicting peers using a 802.11 control radio. The simulation configuration is listed in Table I. Using the default configuration, the analytical estimation for T is $E[T] = 12ms$. We set the initial request backoff window size to $10E[T]$. We set $T_P = 4E[T]$ and the timeouts to $4E[T]$ to schedule local events. Each simulation is 100s. The result is averaged over 30 random seeds.

TABLE I

QUALNET SIMULATION PARAMETERS

Parameter	Default	Range
# of APs (N)	400	9–400
# of data channels	30	5–80
# of neighbors per AP	8	4,8, random in [0,4][0,8]
Traffic load per APs	10Mbps	random in [5,15]Mbps
Control channel rate	1Mbps	1,2,5.5Mbps

*1) **Static Traffic Load**:* We start from a simple scenario where APs have static traffic load. We use this experiment to examine the system optimization time at initialization. We assume APs start from an empty allocation $\mathbb{S} = \{0, 0, ...0\}$ and coordinate to reach an efficient allocation.

Figure 6(a) shows that the coordination overhead (the number of local improvements) scales linearly with the network size and is insensitive to the degree of conflict. This result confirms our analysis on the system convergence. Figure 6(b) shows that the system convergence time flattens quickly as the network scale increases. The different trends in Figure 6(a) and (b) demonstrate the power of parallel execution of local improvement events. As the degree of conflict increases from 4 to 8, the amount of time to finish a single local improvement increases, and hence the system convergence time increases as well. But most importantly, we see that the proposed system converges quickly, requiring only 7-15s for a network of 400 APs.

We are also interested in understanding how local improvements iteratively refine the network-wide spectrum allocation. Figure 6(c) plots the cumulative distribution of each AP's allocated channels at different stages of the entire process (50% represents half into the coordination process). Starting from an empty allocation, APs gradually acquire channels and balance their allocations. Halfway into the process, 75% of APs obtain more than 6 channels, while the rest have 0-4 channels. At the end of the process, the spectrum usage becomes well balanced.

*2) **Dynamic Traffic Load**:* We now examine the FLEX protocol when APs have dynamic traffic load. Instead of starting from an empty allocation, APs will adjust from their present allocations. Therefore, only APs experiencing traffic variations and their neighbors will perform local improvements, significantly reducing adaptation delay. In this case, the convergence time depends on the percentage of APs with traffic variations, their location, and the degree of traffic variations.

Similar to [17], we examine the convergence time using two types of traffic patterns: 1) uniform traffic patterns where APs' traffic varies randomly between [5,15]Mbps; and 2) hotspot traffic patterns where a small percentage of APs located in the center have traffic varying randomly between [5,15]Mbps and the others have static traffic load.

Uniform Traffic Figure 7(a) plots the convergence time for uniform dynamic traffic over a network of 400 APs. As expected, the convergence time increases with the number of APs with traffic variations. Compared to Figure 6(b), the convergence time reduces from 17s to 5.5s if 50% of APs change traffic load. Even at a 100% rate, the delay of 8s is also significantly smaller because APs adjust from a non-empty allocation.

Hotspot Traffic Figure 7(b) plots the convergence time when a number of APs in the center of a 20x20 network change traffic dynamically. We measure the result over different percentage of APs in the hotspot. Compared to the uniform traffic scenario, the system requires slightly higher adaptation time when the same percentage of APs change traffic. This is because local improvement events are densely packed in the center (shown in Figure 7(c)), which reduces the level of the parallelism. On the other hand, Figure 7(c) also shows that the local improvement events are self-contained in the hotspot area, indicating a powerful property of FLEX's local actions. Overall, the adaptation delay of 2-6s is still significantly small for a large WiFi deployment of 400 APs.

*3) **Sensitivity Analysis**:* In this section we study how different system settings affect FLEX. Previous figures have shown the impact of network scale and conflict degree. We now examine the impact of the number of data channels M, the control radio data rate and the network topology.

Varying M Figure 8(a) plots the convergence time for a 400 AP network over different M values. The convergence time scales linearly with M. For a conflict degree of 4, the curve flattens after $M > 50$ because of usage saturation.

Varying Control Radio Rate We compare the convergence time when the control radio data rate varies between 1-5.5Mbps, shown in Figure 8(b). As expected, FLEX benefits from a higher control data rate. More importantly, the results indicate the convergence time increases only linearly as the control data rate drops.

Random AP Topology The above simulations use a grid topology to control the conflict degree. We also evaluate FLEX when APs are randomly deployed. We assume APs are randomly scattered in an area of 4000mx4000m. By adjusting transmission/interference ranges, we produce various network topologies of different conflict conditions. Figure 8(c) shows the result when each AP has up to 4 or 8 conflicting neighbors. Similarly, the convergence time increases linearly with the network scale and is significantly small (0.5–6s).

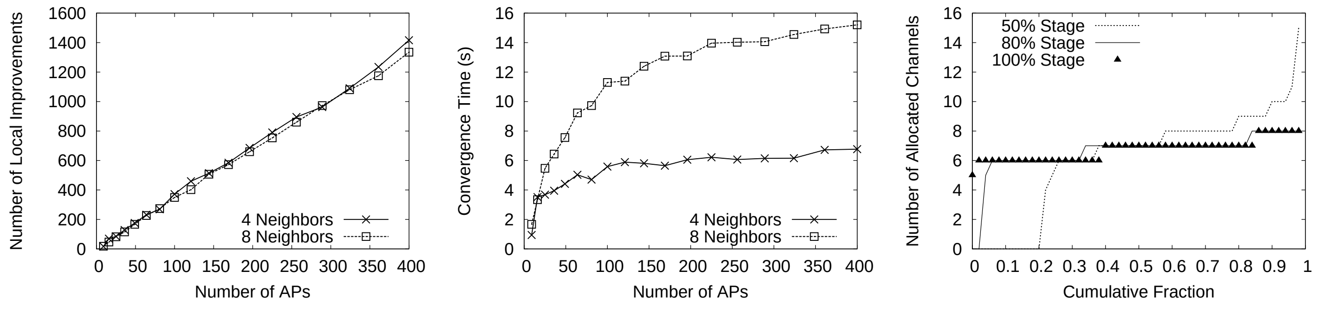

Fig. 6. Coordination protocol performance under static traffic patterns, each AP has the same traffic demand 10. From left to right: (a) Coordination overhead as the number of local events, (b) System convergence time and (c) CDF of AP spectrum usage at different coordination stage.

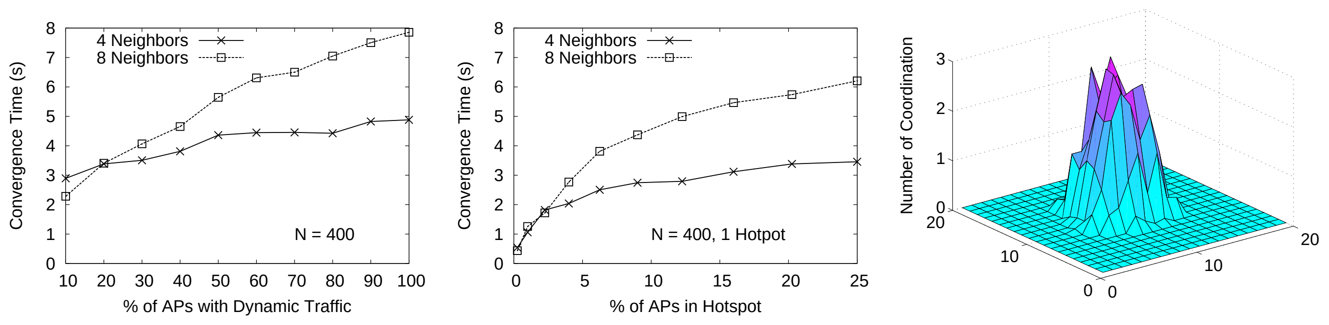

Fig. 7. FLEX's convergence time under dynamic traffic patterns, 400 APs. From left to right: (a) Uniform traffic scenario, (b) Hotspot traffic scenario, (c) Distribution of local improvement events for the hotspot scenario (one hotspot containing 25 APs).

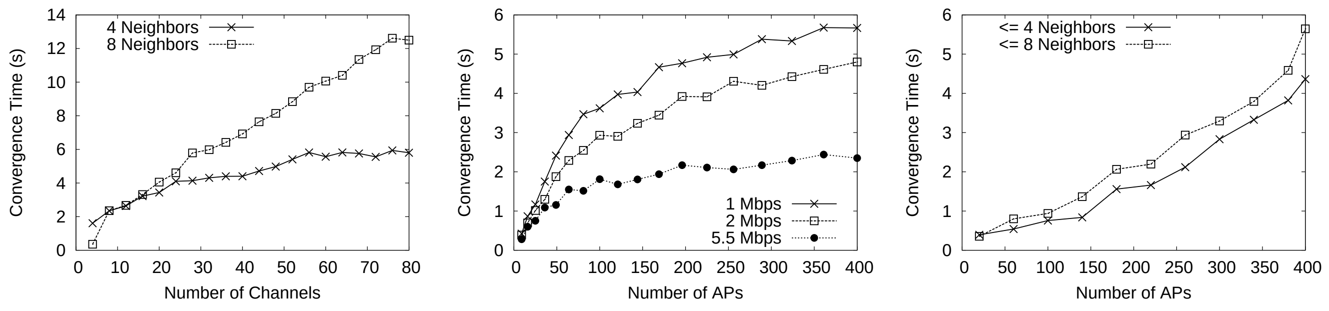

Fig. 8. FLEX's convergence time under various system settings. Assuming dynamic traffic demand, 50% APs change traffic demand uniformly from 5 to 15. From left to right: (a) Varying the number of data channels M, (b) Varying the control radio data rate and (c) Random AP placement.

VI. Practical Considerations

In this section, we discuss practical issues associated with our network model and problem definition.

Identification and Communication with Conflicting APs Because APs are statically placed, each AP can periodically perform interference measurements to identify conflicting peers using the techniques proposed in [12], [17]. FLEX's local improvements can naturaly adapt spectrum allocation to varying conflict conditions across APs as their associated users move. The FLEX coordination protocol assumes APs can directly communicate with conflicting peers to broadcast channel usage and exchange coordination messages. This can be done by configuring the communication range of the control radio as the interference range of the cognitive radio. As a refinement, users associated with APs can also relay these messages between interfering APs.

Extension to Complex Interference Characterizations. In this paper, we use a widely used binary matrix [9], [12], [15] to model the pair-wise interference among access points. Recent work [17] refines this model using probabilistic pair-wise model to approximate packet losses. Our work can be extended to this model by adding weights to channels. On the other hand, many practical interference models are based on aggregated SNR measurements, namely the physical model [9]. The channel allocation problem under this model, however, becomes extremely complex because in order to determine a node's incoming interference, every node in the whole network needs to be considered. We are investigating efficient allocation schemes by reducing the global interference constraint into a set of local constraints.

VII. Related Work

The idea of traffic-aware spectrum allocation has also been considered in existing literatures [13], [17]. In [13], the authors propose an algorithm to assign dynamic-width channels to APs, and improve the system fairness by assigning wider channels to APs with more users. In [17], the authors propose to assign different weights to APs according to their traffics, such that APs with more traffics can be farther separated in frequency to minimize interference. While both [13], [17] propose centralized algorithms that perform well in small scale networks, our distributed algorithm and protocol design targets large-scale dynamic-traffic networks. Our results show that our solution is able to scale to a large network size, and can perform fine grained spectrum allocation by fast local adaptation.

Our work differs from [4], a distributed algorithm that assigns channels to maximize system fairness. First, while [4] assumes backlogged traffic and focuses on AP-level fairness, our work extends the problem scope significantly by considering heterogeneity in traffic demands and addressing user-level fairness. Results in Section V show that our approach significantly outperforms [4] under real traffic dynamics. Second, our work uses a very different evaluation methodology. Instead of focusing on algorithmic complexity such as iterations, we build a highly efficient coordination protocol to realize the proposed algorithms and focus on complexity metrics like system adaptation delay. By enabling spatial parallelism, we conclude that the complexity remains flat as network size grows, rather than the linear growth trend as suggested by [4].

VIII. Conclusion

In this paper we present FLEX, a distributed architecture for WiFi access points to dynamically access spectrum to adapt to varying user traffic demands and maximize user satisfaction. Using cognitive radios, APs coordinate to apply local improvements recursively and tune network-wide spectrum allocation to maximize proportional fairness. We implement FLEX as a light-weight coordination protocol for practical deployment. Through detailed analysis and experimental evaluation, we show that FLEX converges quickly, provides users with proportional-fair spectrum usage and significantly outperforms existing spectrum allocation proposals.

Acknowledgement

We gratefully acknowledge the support from NSF through grant CNS-0721961 and CNS-0832090.

References

[1] Bahl, P., Chandra, R., Moscibroda, T., Wu, Y., and Yuan, Y. Load aware channel-width assignments in wireless lans. Tech. Rep. 2007-79, Microsoft Research, 2007.

[2] Buchsbaum, A. L., Karloff, H., Kenyon, C., Reingold, N., and Thorup, M. Opt versus load in dynamic storage allocation. In *Proc. of STOC* (2003).

[3] Calì, F., Conti, M., and Gregori, E. Dynamic tuning of the ieee 802.11 protocol to achieve a theoretical throughput limit. *IEEE/ACM Trans. Netw. 8*, 6 (2000), 785–799.

[4] Cao, L., and Zheng, H. Spectrum allocation in ad hoc networks via local bargaining. In *Proc. of SECON* (2005).

[5] Challapali, K., Cordeiro, C., and Birru, D. Evolution of spectrum-agile cognitive radios: first wireless internet standard and beyond. In *Proc. of WICON* (2006).

[6] Demers, A., Keshav, S., and Shenker, S. Analysis and simulation of a fair queueing algorithm. In *Proc. of SIGCOMM* (1989).

[7] Gergov, J. Algorithms for compile-time memory optimization. In *Proc. of SODA* (1999).

[8] Henderson, T., Kotz, D., and Abyzov, I. The changing usage of a mature campus-wide wireless network. In *Proc. of MobiCom* (2004).

[9] Jain, K., Padhye, J., Padmanabhan, V., and Qiu, L. Impact of interference on multi-hop wireless network performance. In *Proc. of MobiCom* (2003).

[10] Jansen, K., and Porkolab, L. On preemptive resource constrained scheduling: Polynomial-time approximation schemes. *SIAM J. Discret. Math. 20*, 3 (2006).

[11] Katzela, I., and Naghshineh, M. Channel assignment schems for celluar mobile telecommunication systems. *IEEE Personal Communications 3*, 3 (June 1996), 10–31.

[12] Mishra, A., Shrivastava, V., Agarwal, D., Banerjee, S., and Ganguly, S. Distributed channel management in uncoordinated wireless environments. In *Proc. of MobiCom* (2006).

[13] Moscibroda, T., Chandra, R., Wu, Y., Sengupta, S., Bahl, P., and Yuan, Y. Load-aware spectrum distribution in wireless lans. In *Proc. of ICNP* (2008).

[14] Nandagopal, T., Kim, T.-E., Gao, X., and Bharghavan, V. Achieving mac layer fairness in wireless packet networks. In *Proc. of MobiCom* (2000).

[15] Peng, C., Zheng, H., and Zhao, B. Y. Utilization and fairness in spectrum assignemnt for opportunistic spectrum access. *Mobile Networks and Applications (MONET) 11* (May 2006), 555–576.

[16] Raychaudhuri, D., et al. CogNet - an architecture for experimental cognitive radio networks within the future internet. In *Proc. of MobiArch* (2006).

[17] Rozner, E., Mehta, Y., Akella, A., and Qiu, L. Traffic-aware channel assignment in enterprie wireless networks. In *Proc. of ICNP* (2007).

[18] Wyglinski, A. M. Effects of bit allocation on non-contiguous multicarrier-based cogntiive radio transceivers. In *Proc. of VTC* (2006).

[19] Yuan, Y., Bahl, P., Chandra, R., Chou, P. A., Ferrell, J. I., Moscibroda, T., Narlanka, S., and Wu, Y. Knows: Kognitiv networking over white spaces. In *Proc. of IEEE DySPAN* (2007).

[20] Yuan, Y., Bahl, P., Chandra, R., Moscibroda, T., Narlanka, S., and Wu, Y. Allocating dynamic time-spectrum blocks in cognitive radio networks. In *Proc. of MobiHoc* (2007).

Appendix

A. Proof of Theorem 2

The convergence of the system follows from the fact that the system utility increases every time and there are only a finite number of possible assignments in the system. Next, when traffic demands are equal, following a similar proof in [4], we can prove that after an expected number of O(N^2) iterations, the system will converge. Finally, the third part of Theorem 1 follows from the following lemma:

Lemma 1: Given a $P \times Q$ binary matrix $B_{P\times Q}$, and a list of (traffic) numbers $t_0, t_1, \cdots, t_{Q-1}$, let $r = P/\Sigma t_i$. Define $S_n = \sum_{m=0}^{P-1} B_{m,n}$, for $0 \le n \le Q-1$. Then there exists $0 \le c \le P-1$, s.t.

$$L_c = \prod_{0\le i\le Q-1, B_{c,i}=1} \left(\frac{S_i - 1}{S_i}\right)^{t_i} \ge \frac{r-1}{r}. \qquad (5)$$

Proof: We prove the Lemma by an induction on Q.

When $Q = 1$, $r = P/t_0$. If for some $0 \le c \le P-1$, $B_{c,0} = 0$, then c satisfies (5). Otherwise, for all $0 \le c \le P-1$, $B_{c,0} = 1$. Then $S_0 = r \cdot t_0$, and for arbitrary c, the left side is $(\frac{r\cdot t_0 - 1}{r \cdot t_0})^{t_0}$, by Lemma 2, greater than the right side.

Assuming the lemma holds for all $k < Q$ ($Q \ge 2$), we derive two cases of Q:

• If there exists $0 \le i \le Q-1$, *s.t.* $S_i/t_i \le r$. Then we can find the c required in the Lemma from those with $B_{c,n_i}=0$, with the help of induction hypothesis.

• If for all $0 \le i \le Q-1$, $S_i/t_i > r$, then

$$\begin{aligned}
\prod_{c=0}^{P-1} L_c &= \prod_{0\le c\le P-1}\ \prod_{0\le i\le Q-1, B_{c,i}=1} \left(\frac{S_i-1}{S_i}\right)^{t_i} \\
&= \prod_{0\le i\le Q-1}\ \prod_{0\le c\le P-1, B_{c,i}=1} \left(\frac{S_i-1}{S_i}\right)^{t_i} \\
&= \prod_{0\le i\le Q-1} \left(\frac{S_i-1}{S_i}\right)^{S_i\cdot t_i} \\
(Lemma2)\quad &\ge \prod_{0\le i\le Q-1} \left(\frac{r\cdot t_i-1}{r\cdot t_i}\right)^{r\cdot t_i\cdot t_i} \\
(Lemma2)\quad &\ge \prod_{0\le i\le Q-1} \left(\frac{r-1}{r}\right)^{r\cdot t_i} \\
&= \left(\frac{r-1}{r}\right)^{r\cdot\Sigma t_i} = \left(\frac{r-1}{r}\right)^{P} \qquad (6)
\end{aligned}$$

Hence $\prod_{c=0}^{P-1} L_c \ge (\frac{r-1}{r})^P$. Because $L_c \ge 0$ for all $0 \le c \le P-1$, there must exist $0 \le c \le P-1$, s.t. $L_c \ge \frac{r-1}{r}$. ■

In the following, we will be using these results:

Lemma 2: $f(x) = (1-\frac{1}{x})^x$ is monotonically increasing in $[1,+\infty)$. $f(x) = (1-\frac{1}{r\cdot x})^x$, $r \ge 1$ is monotonically increasing in $[1,+\infty)$. $f(x) = (1+\frac{1}{r\cdot x})^x$, $r \ge 1$, is monotonically increasing in $[1,+\infty)$.

Next, we prove Theorem 1 using LEMMA 1. Let $M = (t_n + \sum_{k\in\mathbb{N}(n)} t_k)\cdot r + r_0,\ \ r, r_0 \in \mathcal{Z}, 0 \le r_0 \le d$, $r = \lfloor \frac{M}{t_n+\sum_{k\in\mathbb{N}(n)} t_k} \rfloor$. If $r=0$, the theorem is trivial. In the following we assume $r>0$. Suppose on the contrary that for some n,

$$S_n \le \left(\left\lfloor \frac{M}{t_n + \sum_{k\in\mathbb{N}(n)} t_k} \right\rfloor - 1\right)\cdot t. \qquad (7)$$

Suppose there are d elements in $\mathbb{N}(n)$, and suppose W.L.O.G. they are indexed by $\{0,1,\cdots,d-1\}$ (*i.e.* $\mathbb{N}(n) = \{0,1,\cdots,d-1\}$), and n is indexed by d. Also suppose W.L.O.G. that the channels assigned to n are indexed by $\{M-1, M-2, \cdots, M-S_n\}$. We can represent the allocation matrix $\{a_{m,n}\}$ by:

Index	0	$\cdots$	d-1	d	$\cdots$
0	$\cdots$	$\cdots$	$\cdots$	0	$\cdots$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\cdots$	$\vdots$
$M-S_n-1$	$\cdots$	$\cdots$	$\cdots$	0	$\cdots$
$M-S_n$	0	$\cdots$	0	1	$\cdots$
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
M-1	0	$\cdots$	0	1	$\cdots$

Now $a_{(0:M-S_n-1),(0:d-1)}$ satisfies the conditions in Lemma 1, with $P = M - S_n$ and $Q = d$. By Lemma 1, there exists $0 \le c \le M - S_n - 1$, s.t.

$$\prod_{0\le i\le d-1, a_{c,i}=1} \left(\frac{(\sum_{m=0}^{M-S_n-1} a_{m,i}) - 1}{(\sum_{m=0}^{M-S_n-1} a_{m,i})}\right)^{t_i} \ge \frac{r'-1}{r'}. \qquad (8)$$

$$\begin{aligned}
\text{where } r' &= \frac{M-S_n}{\Sigma t_i} \ge \frac{M-(r-1)\cdot t}{\Sigma t_i} \\
&= \frac{((t+\Sigma t_i)\cdot r + r_0) - (r\cdot t - t)}{\Sigma t_i} > r. \qquad (9)
\end{aligned}$$

Because $a_{m,i} = 0$ for $M - S_n \le m \le M-1$ and $0 \le i \le d-1$, we have $\sum_{m=0}^{M-S_n-1} a_{m,i} = \sum_{m=0}^{M-1} a_{m,i} = S_i$, $0 \le i \le d-1$. Next, from (8),

$$\prod_{0\le i\le d-1, a_{c,i}=1} \left(\frac{S_i-1}{S_i}\right)^{t_i} \ge \frac{r'-1}{r'} > \frac{r-1}{r}. \qquad (10)$$

• $S_n = 0$. Then

$$\begin{aligned}
&(S_n+1)^t \cdot \prod_{i\in\mathbb{N}(n)\wedge a_{c,i}=1} (S_i-1)^{t_i} \\
&> 0\ (by(10)) \\
&= (S_n)^t \cdot \prod_{i\in\mathbb{N}(n)\wedge a_{c,i}=1} (S_i)^{t_i}. \qquad (11)
\end{aligned}$$

• $S_n > 0$. Then

$$\begin{aligned}
&\frac{(S_n+1)^t \cdot \prod_{i\in\mathbb{N}(n)\wedge a_{c,i}=1} (S_i-1)^{t_i}}{(S_n)^t \cdot \prod_{i\in\mathbb{N}(n)\wedge a_{c,i}=1} (S_i)^{t_i}} \\
&= \left(\frac{S_n+1}{S_n}\right)^t \cdot \prod_{0\le i\le d-1, a_{c,i}=1} \left(\frac{S_i-1}{S_i}\right)^{t_i} \\
&(by(10)) > \left(\frac{S_n+1}{S_n}\right)^t \cdot \frac{r-1}{r} \\
&(by(7)) \ \ \ge \left(\frac{(r-1)\cdot t+1}{(r-1)\cdot t}\right)^t \cdot \frac{r-1}{r} \\
&(Lemma2) \ge \frac{(r-1)+1}{(r-1)} \cdot \frac{r-1}{r} = 1. \qquad (12)
\end{aligned}$$

In both cases, it shows that we can apply local improvements over A by assigning channel c to node n. This contradicts with the assumption that the system has converged and no more local improvements can improve the system utility.