Cooperative Caching in Dynamic Shared Spectrum Networks

Dibakar Das, *Student Member, IEEE,* and Alhussein A. Abouzeid, *Senior Member, IEEE*

***Abstract*—This paper considers cooperation between primary and secondary users in shared spectrum radio networks via caching. We first consider a network with one channel shared between a single macro (primary) base station and multiple micro (secondary) base stations. Secondary base stations can cache some primary files and thereby satisfy content requests generated from nearby primary users. For this cooperative scenario, we develop two caching and scheduling policies under which the set of primary and secondary request generation rates that can be supported increases from the case without cooperation. The first of these algorithms, Fixed Primary Caching Policy (FPCP), provides more gain in the set of supportable request generation rates. However under this algorithm primary packet transmissions from secondary base stations have the same priority of access as secondary packets and thus might suffer in terms of delay. In the second algorithm, Variable Primary Caching Policy (VPCP), primary packet transmissions from the secondary base stations have higher priority of access than that of secondary packets. We find that the set of request generation rate vectors for which all queues in the network are stable under each of these algorithms is greater than that under any non-cooperative algorithm. We conduct extensive simulations to compare the performance of both algorithms against that of an optimal non-cooperative algorithm. Finally, we extend the analysis to a network with multiple channels.**

***Index Terms*—Cognitive radio, caching, small base station, heterogeneous network, cooperation, relay, queuing theory, Lyapunov drift, network stability.**

I. Introduction

In recent years, dynamic shared spectrum or cognitive radio networks have been widely studied to efficiently use the often under-utilized spectrum [1]. In such networks secondary (i.e., unlicensed) users opportunistically transmit on a given channel if the primary (i.e., licensed) users of that channel are inactive. Recent works have also studied cooperation schemes where primary and secondary users assist in each other's transmissions. Such schemes can help both types of users by reducing the duration of primary users' transmission activity, thereby increasing secondary users' transmission opportunities. Such cooperation has been studied from a physical-layer information-theoretic perspective e.g., [2]–[4]. Other recent works, e.g., [5]–[9], study network layer aspects of cooperation such as queuing and prioritized scheduling. Our work in this paper belongs to the latter group of works.

Another solution that has been proposed to support increasing mobile network traffic is the use of base stations with smaller coverage areas (commonly called small-cells [10]). This leads to a higher spatial re-use of the spectrum. However, the limited capacity of backhaul links at the base stations may reduce the impact of this approach [11]. As a result, caching popular files at nearby base stations has been proposed (e.g., in [11]) to reduce back-haul usage as well as improve the delay performance for users. Caching is also an attractive practical solution since storage-capacity is relatively inexpensive compared to other network resources.

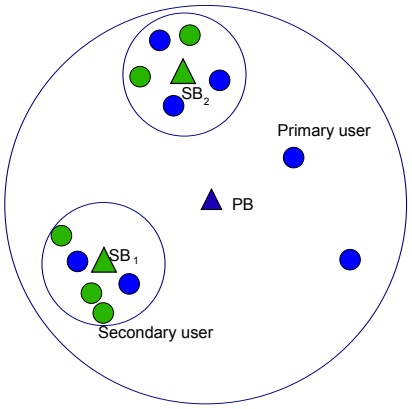


Fig. 1. A network with primary base station (PB) co-existing with secondary base stations: SB_1 and SB_2. The outermost circle and smaller circles indicates the transmission region of PB and SB_1, SB_2 respectively. Number of secondary base stations (M) is 2, number of primary users ($N^{(p)}$) and secondary users ($N^{(s)}$) are 7 and 5 respectively, number of primary users within transmission range of SB_1 ($\phi_1^{(p)}$) is 2, number of secondary users within transmission range of SB_1 ($\phi_1^{(s)}$) is 3.

In this work we explore cooperation between primary and secondary networks via caching. We consider a primary network consisting of a single base station that serves primary users. Co-existing with the primary network is a set of small secondary base stations that serve secondary users. A schematic example of such a network architecture is shown in Fig.1. Content requests from users are queued at base stations. The base stations serve these requests by transmitting packets corresponding to the requested files from their cache if the file is available in the cache. If the requested file is not currently present in its cache (i.e., if it is not 'cached'), then the base station fetches this file, possibly after some delay, so as to satisfy the content request. We assume that content is fetched periodically; we call this period the *cache-refresh* period. Secondary base stations located closer to primary users may have better down-link channels than the primary base station. Therefore, if some of the content requests from primary users are served via these secondary base stations then it might free up spectrum resources for use by the secondary users.

For the above network scenario we study the key problem of developing caching and scheduling policies with performance guarantees. In particular, we design algorithms under which

D. Das and A.A. Abouzeid are with the Department of Electrical and Computer Engineering, Rensselaer Polytechnic Institute, Troy, NY, 12180 USA e-mail: (dasd2@rpi.edu; abouzeid@ecse.rpi.edu).

a centralized network controller determines: (a) which files to cache in every cache-refresh period and, (b) which file requests (referred to as "requests" in the rest of the paper) to admit at a given base station and how to schedule transmissions in each time slot. The objective of the network controller is to maintain the stability of the queues, i.e., to keep the length of all queues in the network bounded.

For a network with one channel we develop two algorithms: Fixed Primary Caching Policy (FPCP) and Variable Primary Caching Policy (VPCP), using Lyapunov-drift techniques [12]. The objective of FPCP is to increase the set of supportable request generation rates while ensuring that secondary base stations can serve at least one type of secondary file request in every period. The FPCP algorithm solves this problem by having each secondary base-station cache a fixed set of primary files and only one secondary file in every period. The objective of the VPCP algorithm is to increase the set of supportable request generation rates such that (a) secondary base stations serve primary user requests with high priority and (b) minimize the number of primary files cached at secondary base stations. The motivation behind the first constraint is to ensure that primary user requests have higher priority of service even at secondary base stations. The motivation behind the second constraint is to ensure that secondary base stations only cache as many primary files as is required to maintain stability of the network. The VPCP algorithm solves this problem by having each secondary base station cache variable sets of primary files in each period based on a penalty parameter. We then prove that the set of primary and secondary request generation rate vectors for which all queues in the network are stable under each of these algorithms is greater than that under any non-cooperative algorithm. Unlike the FPCP algorithm under VPCP algorithm the secondary base-stations can serve more than one type of secondary file requests in each cache refresh period. Simulation results show that this algorithm, with proper selection of a penalty parameter, tends to have better delay performance than the FPCP algorithm when request generation rates are low. In such scenarios it is not necessary, for the purpose of system stability, to cache as much primary files as possible in every secondary base-station. In Section VII we also show that the delay performance of VPCP is not guaranteed to be better than FPCP for any choice of the penalty parameter. However, the guaranteed stability region of the network under VPCP is less than that under FPCP.

Finally, we consider a network with multiple orthogonal channels where all secondary base stations are non-interfering to each other (i.e., they can simultaneously transmit when primary base station is not transmitting) and develop an algorithm for this scenario, referred to as MultiChannel Caching Policy (MCCP).

There is a significant body of works on caching in non-cognitive wireless networks (e.g. [11], [13], [14] and the references therein). Caching in cognitive networks has been studied recently in [15] and [16]. However these works did not consider primary-secondary cooperation. Our periodic cache-refresh policy and the use of Lyapunov drift to develop a scheduling policy is motivated by [17] and [18]. However, their results do not directly apply in the cognitive network setting since here primary users have higher priority of channel access than secondary users.

The novelty of our work lies in showing how Lyapunov drift techniques can be used to design efficient joint cooperative caching and scheduling algorithms in dynamic shared spectrum networks. The difficulty of designing such algorithms lies in higher priority of channel access for primary users. While a simple Lyapunov drift-based algorithm tries to serve the queues with higher backlogs first, in our model the queues in the primary base station would have to be served before those at the secondary base stations even when the former have relatively lower backlog. The difficulty of developing caching and scheduling algorithms is even higher when primary user requests are served with higher priority even at secondary base stations, a scenario considered in the VPCP algorithm. Such complications resulting from higher priority of service for primary users are addressed in this work, and up to our knowledge, have not been addressed before.

The paper is organized as follows. In Section II we describe the system model for a single channel case. In Section III, we define an achievable capacity region consisting of supportable request generation rate vectors (i.e., request generation rate vectors for which all queues in the network are stable). Using this region we measure the stability performance of the FPCP, VPCP and MCCP algorithms. In Section IV we present the FPCP algorithm and show that under FPCP all queues are stable for every request generation rate vector within the achievable capacity region. In Section V we present the VPCP algorithm and find a guaranteed stability region for it. Section VI presents the multichannel network model, an achievable capacity region and the MCCP algorithm that stabilizes the network for all request generation vectors within this region. In Section VII we present simulation results. Section VIII concludes the paper. Due to lack of space, we provide proofs of Lemmas 1, 2 and Theorems 1, 2 and 3 in our technical report [19].

II. System model

The network consists of a macro-cell wherein a single primary base station PB serves $N^{(p)}$ primary users: PU_1,...,$\text{PU}_{N^{(p)}}$. It also contains M secondary small-cells $\text{SC}_1, \ldots, \text{SC}_M$ associated with secondary base stations (SB): SB_1,...,SB_M respectively. There are $N^{(s)}$ secondary users: SU_1,...,$\text{SU}_{N^{(s)}}$. Exactly one small base station serve each secondary user. We consider a discrete time model. Every request is served by successfully transmitting C packets of equal size. A base station attempts a new packet transmission only at the beginning of a time slot; it can attempt at most one such transmission at any time slot. If a packet transmission fails, the packet is re-transmitted at another time slot. Next, we present details about the transmission model, interference model, caching and scheduling model. We summarize important notations in Table I.

A. Transmission Model for Primary and Secondary Base stations

We model a hybrid access cognitive femtocell system wherein SBs transmit only in slots when PB is not transmitting and can serve nearby primary users. All base stations transmit

TABLE I
LIST OF NOTATIONS

$N^{(p)}(N^{(s)})$	number of primary (secondary) users	$\text{PU}_i(\text{SU}_i)$	i-th primary (secondary) user
M	number of small cells	$t_{r,j}$	j'th slot in r'th period
$\lvert F^{(p)}\rvert(\lvert F^{(s)}\rvert)$	set of primary (secondary) files	$\tilde{D}^{(p)}$	set of primary availability matrices
$\lambda_i^{(p)}(\lambda_i^{(s)})$	request generation rate from $\text{PU}_i(\text{SU}_i)$	$D^{(p)}$	primary availability matrix
$A_{f,i}^{(p)}(t)(A_{f,i}^{(s)}(t))$	requests generated at slot t by $\text{PU}_i(\text{SU}_i)$	PB	primary base station
$\text{F}_i^{(p)}(\text{F}_i^{(s)})$	i-th primary (secondary) file	SB_i	i-th secondary base station
$\mu_{f,0}^X(t)$	transmission rate offered to PB for packets of file f under policy X at slot t		
$\mu_{f,k}^X(t)$	transmission rate offered to SB_k for packets of file f under policy X at slot t		
$U_{f,i}^{(p)}(t)(U_{f,i}^{(s)}(t))$	queue length at slot t for primary (secondary) file f in SB_i		
$U_{f,0}^{(p)}(t)$	queue length at slot t for primary file f in PB		
$P_i^{(p)}(P_i^{(s)})$	probability of i-th primary (secondary) file being requested, given primary (secondary) file request		

at fixed power. All SBs have identical transmission range. We consider the case where some primary users have poor downlink channel from PB than from adjacent SBs. This situation can occur (e.g., due to shadowing or near-far effect) in several real life scenarios such as in the case of macro-cell edge users that are far away from PB but happen to be closer to an SB. We are interested in studying the best case gains obtained by offloading primary user traffic to nearby SBs. Hence, for simplicity, we assume non ideal transmissions from PB to primary users, but ideal transmission from SBs to users within their transmission range:

> **A1)** At every slot, a transmission from PB to a primary user located within and out of transmission range of an SB succeeds with probability p and p' respectively (where $0 < p, p' \leq 1$). At any slot an SB transmits to a user within its transmission range only if PB is not transmitting simultaneously; the transmission succeeds with probability 1.

In the rest of the paper, for simplicity, we have assumed that p' equals p. However, the analysis can be easily extended for any other value of p'. Each primary user is served by at most one SB, i.e., no two small-cells contain the same primary user. We denote the set of primary and secondary users in SC_i, $i \in \{1, \ldots, M\}$, as $\phi_i^{(p)}$ and $\phi_i^{(s)}$ respectively. All secondary users and base stations are within the transmission range of PB and share a single channel.

B. Caching Model

Every time an SB fetches a set of files for caching it incurs some overhead cost. We model this cost by requiring that SBs only cache files periodically. Higher frequency of caching reflects higher cost. A cache-refresh period is defined as the number of time slots between two successive caching events. A cache refresh period may also represent the frequency with which contents in files become outdated thereby requiring newer versions to be fetched. It consists of T time slots with the first caching event being at time slot $t = 1$. Henceforth, we refer to the cache refresh period simply as *period* whenever there is no confusion.

C. Content Requests

We consider the case where primary and secondary networks are different, and hence have statistically different content requirements[1]. This can occur, for example, if the small-cells serve an industrial or academic environment while users of the macro-cell are the general public who are typically interested in video content. We denote the library of files requested by primary users as $F^{(p)}$ and individual files in the set as $\text{F}_1^{(p)}, \ldots, \text{F}_{|F^{(p)}|}^{(p)}$. Similarly we denote the library of files requested by secondary users as $F^{(s)}$ and individual files in the set as $\text{F}_1^{(s)}, \ldots, \text{F}_{|F^{(s)}|}^{(s)}$. The sets $F^{(p)}$ and $F^{(s)}$ are mutually exclusive. Henceforth we will call files in $F^{(p)}$ and $F^{(s)}$ as primary and secondary files respectively. Similarly, we call packets corresponding to primary and secondary files as primary and secondary packets respectively. All files are of equal size. We denote the set of primary and secondary files cached by base station SB_k, $k \in \{1, \ldots, M\}$ at time slot τ as $H_k^{(p)}(\tau)$ and $H_k^{(s)}(\tau)$ respectively where $H_k^{(p)}(\tau) \in F^{(p)}$ and $H_k^{(s)}(\tau) \in F^{(s)}$.

Users request files according to a fixed popularity distribution. Given a primary user requests a file, the requested file is $\text{F}_i^{(p)}$ with probability $P_i^{(p)}$ where $i \in \{1, 2, \ldots, |F^{(p)}|\}$. Similarly, given a secondary user requests a file, the file $\text{F}_i^{(s)}$ is requested with probability $P_i^{(s)}$. Without loss of generality, we assume files are indexed according to their popularity, i.e., $P_i^{(p)} \geq P_{i+1}^{(p)}$ and $P_j^{(s)} \geq P_{j+1}^{(s)}$ for every $i \in \{1, 2, \ldots, |F^{(p)}| - 1\}$ and $j \in \{1, 2, \ldots, |F^{(s)}| - 1\}$.

At every time slot, primary user PU_m, $m \in \{1, \ldots, N^{(p)}\}$, requests a file with probability $\lambda_m^{(p)}$. Note that the probability that PU_m requests file $\text{F}_i^{(p)}$ therefore equals $\lambda_m^{(p)} P_i^{(p)}$. Similarly, at every time slot, secondary user SU_l, $l \in \{1, \ldots, N^{(s)}\}$, requests files with probability $\lambda_l^{(s)}$. All request generation processes are identical and independently distributed (iid) from time slot to time slot. We denote primary and secondary request generation rate vectors:$(\lambda_1^{(p)}, \ldots, \lambda_{N^{(p)}}^{(p)})^T$ and $(\lambda_1^{(s)}, \ldots, \lambda_{N^{(s)}}^{(s)})^T$ as $\boldsymbol{\lambda}^{(p)}$ and $\boldsymbol{\lambda}^{(s)}$ respectively.

D. Service Discipline for Primary Users

A request from a primary user is served by either a secondary base station (if the primary user is located in a small-cell and the associated secondary base station contains the file) or by the primary base station. Every base station maintains *queues* for every file. Queues store packet requests that need to be satisfied in response to a request. Whenever a base station admits request for a primary file $f \in F^{(p)}$, C packet requests[2] are queued at the queue for file f. Within each

[1]The case where the users have homogeneous content requirements can be studied in a similar fashion.

[2]Recall, each file request is served by successfully transmitting C packets.

queue packet requests are satisfied in a first-come first-serve (FCFS) manner.

In order to focus only on the effect of cooperative caching by the secondary network, we assume the primary base station can cache all the $|F^{(p)}|$ primary files. On the other hand, the size of cache at each secondary base station is finite. Each such cache can store at most B files where $0 \leq B \leq \min(|F^{(p)}|, |F^{(s)}|)$. A secondary base station only admits a primary request if the requested file is currently present in its cache[3]. We denote a request generated from PU_i, $i \in \{1, \ldots, N^{(p)}\}$, at time slot t for file f (where $f \in F^{(p)}$) by variable $A^{(p)}_{f,i}(t)$. This variable equals C if such a request is indeed generated at t; otherwise it equals 0.

We denote the queue length of primary file f (where $f \in F^{(p)}$) in SB_k, $k \in \{1, \ldots, M\}$, at time slot t as $U^{(p)}_{f,k}(t)$. If PU_i is within transmission range of a secondary base station SB_k containing the file f requested by PU_i at t, then SB_k *may* admit this request[4]; C packet requests are queued at SB_k and $U^{(p)}_{f,k}(t)$ is incremented by C. Otherwise, this request will be served by PB and the length of the associated queue, denoted as $U^{(p)}_{\text{f},0}(t)$, will be incremented by C. The system begins at time slot t=1 with all queues being initially empty.

On successful transmission of a packet corresponding to primary file f (where $f \in F^{(p)}$) from PB at t, $U^{(p)}_{\text{f},0}(t)$ is decremented by 1. We denote the transmission rate offered to base station PB and SB_k for packets of file f', where $f' \in F^{(p)} \cup F^{(s)}$, under caching and scheduling policy X at time slot t by the binary variables $\mu^X_{f',0}(t)$ and $\mu^X_{f',k}(t) \in \{0, 1\}$ respectively. Therefore the queue length at PB under policy X evolve as:

$$\begin{aligned} U^{(p)}_{\text{f},0}(t+1) &= U^{(p)}_{\text{f},0}(t) - \mu^X_{f,0}(t) I_{\text{PB}}(t) \\ &+ \sum_{i:\text{PU}_i \notin \cup_{j=1}^{M} \phi^{(p)}_j} A^{(p)}_{f,i}(t) \\ &+ \sum_{k=1}^{M} \sum_{i:\text{PU}_i \in \phi^{(p)}_k} A^{(p)}_{f,i}(t) \left(1 - \hat{I}_{f,k}(t)\right) \\ &+ \sum_{k=1}^{M} \sum_{i:\text{PU}_i \in \phi^{(p)}_k} A^{(p)}_{f,i}(t) \left(1 - \chi^{(p)}_{f,k}(t)\right) \hat{I}_{f,k}(t) \\ &\quad \forall f \in F^{(p)}, \end{aligned} \tag{1}$$

where $I_{\text{PB}}(t)$ is an indicator variable representing whether the transmission from PB to the primary user at time slot t was successful or not, $\chi^{(p)}_{f,k}(t)$ is an admission-control variable representing whether SB_k (where $k \in \{1, \ldots, M\}$) admits request at time slot t for cached primary file $f \in H^{(p)}_k(t)$ and $\hat{I}_{f,k}(t)$ is an indicator variable representing whether SB_k (where $k \in \{1, \ldots, M\}$) contains a primary file f at time slot t or not. The variable $I_{\text{PB}}(t)$ equals 1 if the transmission is successful at t and is zero otherwise; $\chi^{(p)}_{f,k}(t)$ equals 1 if SB_k admits request for f at t and is zero otherwise. The variable $\hat{I}_{f,k}(t)$ equals 1 if SB_k contains file f at t and is zero otherwise. No transmission-rate is offered to PB when all its queues are empty.

The queue length of primary files at secondary base stations evolve under policy X as:

$$\begin{aligned} U^{(p)}_{f,k}(t+1) &= \max\left\{U^{(p)}_{f,k}(t) - \mu^X_{f,k}(t), 0\right\} \\ &+ \sum_{i:\text{PU}_i \in \phi^{(p)}_k} A^{(p)}_{f,i}(t) \chi^{(p)}_{f,k}(t) \\ &\quad \forall k \in \{1, \ldots, M\}, f \in H^{(p)}_k(t). \end{aligned} \tag{2}$$

E. Service Discipline for Secondary Users

Requests from a secondary user is submitted and always admitted by the unique secondary base station associated with that user. Similar to the case of primary files, each secondary base station maintains a queue for every secondary file. We denote a request generated from SU_j, $j \in \{1, \ldots, N^{(s)}\}$, at time slot t for a secondary file f (where $f \in F^{(s)}$) by a variable $A^{(s)}_{f,j}(t)$. It equals C if such a request is indeed generated at t; otherwise it equals 0. For every $k \in \{1, \ldots, M\}$ we denote the queue length in SB_k at t of file f as $U^{(s)}_{f,k}(t)$. The length of this queue is incremented by C every time SB_k receives a request for file f. It is decremented by 1 every time SB_k transmits a packet of f. The queue evolve under policy X as,

$$\begin{aligned} U^{(s)}_{f,k}(t+1) &= \max\left\{U^{(s)}_{f,k}(t) - \mu^X_{f,k}(t), 0\right\} \\ &+ \sum_{j:\text{SU}_j \in \phi^{(s)}_k} A^{(s)}_{f,j}(t) \\ &\quad \forall k \in \{1, \ldots, M\}, f \in F^{(s)}. \end{aligned} \tag{3}$$

We refer to the queues corresponding to primary and secondary files as primary and secondary queues respectively. We denote the vector of all primary and secondary queues in the network at slot t as $\mathbf{U}^{(p)}(t)$ and $\mathbf{U}^{(s)}(t)$ respectively.

In this paper we use the notion of strong stability of queues, defined as follows.

Definition 1: A discrete time queue $U(t)$ is strongly stable if $\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t-1} E[|U(\tau)|] < \infty$. The network is strongly stable if every queue in the network is strongly stable.

F. Interference Model

We represent the set of all secondary base stations that can transmit simultaneously by an *activation* vector of length M. An activation vector E is binary and its m'th, $m \in \{1, \ldots, M\}$, component corresponds to SB_m. It is set to 1 if SB_m is transmitting in that time slot; otherwise it is set to zero. The set of all feasible activation vectors is denoted as $\tilde{E}$. No secondary base station can transmit when PB is transmitting.

III. Achievable Capacity Region

In this section we describe an achievable capacity region Λ corresponding to the system model in Section II. The region contains every request generation rate vector for which all queues in the network are stable considering only a *restricted* set of caching and scheduling algorithms. First, we specify this restricted set of algorithms that satisfy a certain caching constraint; this constraint is useful for tractable analysis. Sec-

[3]Secondary base stations do not admit requests for uncached primary files because doing so will likely deteriorate delay performance of primary users. Note that secondary base stations cannot serve such requests at least until the next cache refresh period. But PB can always serve such requests as its cache always contains every primary file.

[4]The admission control decision is taken by the network controller. The controller may decide not to admit a primary request at a secondary base station even when the requested file is present in the latter's cache.

ond, we introduce a new term called the *primary availability matrix* which represents the set of primary files currently cached at secondary base stations. Third, we obtain the set of feasible transmission rates for secondary base stations given the probability with which each primary availability matrix is used in each period. Fourth, we use the set of feasible transmission rates for secondary base stations to describe the region Λ similar to the description of capacity region in [12]. Fifth, we find a simpler description of the region Λ in Lemma 1. Sixth, for the network with non-interfering secondary base stations we compare the region Λ with the actual capacity region in Lemma 2. The actual capacity region consists of all request generation rate vectors for which every queue is stable, considering all stationary caching and scheduling algorithms, not just the restricted ones. Seventh, we discuss extending our analysis for unequal file sizes. Finally, we discuss maximum supportable request generation rates for both types of users for a symmetric network.

In the description of the region Λ we only consider the set of caching and scheduling algorithms that satisfy the following constraint:

C1: Each secondary base station has at least one secondary file in its cache in every slot.

We consider only this restricted set of algorithms instead of all caching and scheduling algorithms because it allows for tractable analysis and is not a huge limitation. In particular, it allows us to determine the stability constraints at secondary base stations without accounting for probability of the event: "the network controller offers transmission rate to a secondary base station which has no cached secondary file". Obtaining probability of such events is cumbersome but is required to analyze stability performance of algorithms which do not satisfy constraint **C1**. Using constraint **C1** does not distort the system model significantly from that in real environments for a couple of reasons. First, this constraint guarantees each secondary base station at least has one secondary file to transmit in any slot. Since the main goal of a secondary base station is to serve secondary users, in a real life system, a constraint similar to C1 (e.g., one that requires each secondary base station to cache at least 2 or some other non-zero number of secondary files per period) might be required to balance the ability of a secondary base station to transmit multiple type of secondary files in a period versus increasing transmission opportunities to secondary users. Second, we show that the performance gap (in terms of supportable rates) between policies satisfying **C1** and other stationary policies corresponds to the rate of requests for the B'th most popular primary file. For large B and $|F^{(p)}|$, as in real systems (e.g., $B = 100$ and $|F^{(p)}| = 1000$ [11]), this rate is small. We quantify this gap in Lemma 2 for the network in which all secondary base stations are non-interfering to each other.

Now we introduce the *primary availability matrix*. A primary availability matrix is a binary matrix that indicates the availability of primary files in SBs at any given time slot. Each such matrix contains the same number of rows and columns as the total number of primary files and number of SBs respectively. The (l, j)'th component of a primary availability matrix $\mathbf{D}^{(p)}$ (where $l \in \{1, \ldots, |F^{(p)}|\}$, $j \in \{1, \ldots, M\}$), denoted as $D_{l,j}$, equals 1 if the file $\mathrm{F}_l^{(p)}$ is present at the cache of SB_j; otherwise it equals zero. Given a primary availability matrix $\mathbf{D}^{(p)}$, we indicate whether or not a primary file f (where $f \in F^{(p)}$) requested by PU_i, $i \in \{1, \ldots, N^{(p)}\}$, is present in a nearby secondary base station by the binary variable $Q(i, f|\mathbf{D}^{(p)})$. In particular, for every primary file $\mathrm{F}_l^{(p)}$, the variable $Q(i, \mathrm{F}_l^{(p)}|\mathbf{D}^{(p)})$ equals 1 if there exists j such that (s.t.) $D_{l,j}^{(p)} = 1$ and $\mathrm{PU}_i \in \phi_j^{(p)}$; it is set to 0 otherwise. Since for tractable analysis we consider only algorithms that satisfy constraint **C1**, we consider only the set of primary availability matrices for which the sum of every column is less than B. This is because each column of the primary availability matrix denotes the set of primary files cached at an SB and under **C1** each SB can have upto $(B-1)$ primary files in its cache at any slot. We denote this set as $\tilde{D}^{(p)}$.

We find the set of feasible transmission rates for SBs, given a probability distribution over the set of primary availability matrices in $\tilde{D}^{(p)}$, in the following manner. Consider the vector $\boldsymbol{q} = (q_{\mathbf{D}^{(p)}})$ where the term $q_{\mathbf{D}^{(p)}}$ denotes the iid probability with which the primary availability matrix $\mathbf{D}^{(p)} \in \tilde{D}^{(p)}$ is selected in each period. For every $\mathbf{D}^{(p)}$ the term $q_{\mathbf{D}^{(p)}}$ should be non-negative and less than 1 (i.e., $0 \leq q_{\mathbf{D}^{(p)}} \leq 1$) and the sum of all $q_{\mathbf{D}^{(p)}}$ terms should equal 1 (i.e., $\sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} q_{\mathbf{D}^{(p)}} = 1$). Clearly, due to assumption A1, given a set of primary files cached at SBs, transmission opportunities for SBs are maximized when each SB admits all requests of a cached primary file. Then, the rate of primary packet transmissions by all SBs, equals $C \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} Q(k, \mathrm{F}_l^{(p)}|\mathbf{D}^{(p)}) q_{\mathbf{D}^{(p)}}$. For a stable system the rate of primary packet transmissions by PB is the rate of all primary packet requests minus the rate of all primary packet requests served by SBs, i.e., $\sum_{k=1}^{N^{(p)}} C\lambda_k^{(p)} - C \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} Q\left(k, \mathrm{F}_l^{(p)}|\mathbf{D}^{(p)}\right) q_{\mathbf{D}^{(p)}}$. The probability that PB is not transmitting in a given time slot is therefore,

$$\left\{1 - \frac{\sum_{k=1}^{N^{(p)}} C\lambda_k^{(p)} - C \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} Q\left(k, \mathrm{F}_l^{(p)}|\mathbf{D}^{(p)}\right) q_{\mathbf{D}^{(p)}}}{p}\right\}.$$

We denote as $\Gamma(\boldsymbol{q})$ the set of feasible secondary transmission-rate vectors for a given vector $\boldsymbol{q}$. The set $\Gamma(\boldsymbol{q})$ is defined in average sense as the convex hull of all feasible activation vectors when PB is not transmitting, multiplied by the probability that PB is not transmitting, as

$$\Gamma(\boldsymbol{q}) = \left\{1 - \frac{\sum_{k=1}^{N^{(p)}} C\lambda_k^{(p)}}{p} - \frac{C \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} Q\left(k, \mathrm{F}_l^{(p)}|\mathbf{D}^{(p)}\right) q_{\mathbf{D}^{(p)}}}{p}\right\} \mathbf{conv}(\tilde{E}) \quad (4)$$

where **conv** of a set of vectors is the set of all possible convex combinations of its elements.

We use the set of feasible transmission rate vectors for SBs to define the region Λ. The region Λ is the set of all possible request generation rate vectors: $(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)})$ for which there exists vector $\boldsymbol{q}$, $(R_1, \ldots, R_M)^T \in \Gamma(\boldsymbol{q})$ and variable π_0 s.t.

$$\frac{\pi_0}{p} \leq 1 \tag{5}$$

$$C \left\{ \sum_{i:\text{SU}_i \in \phi_j^{(s)}} \lambda_i^{(s)} \right. + \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{k:\text{PU}_k \in \phi_j^{(p)}} \left. \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} \lambda_k^{(p)} D_{l,j}^{(p)} q_{\mathbf{D}^{(p)}} \right\} \leq R_j \quad \forall 1 \leq j \leq M, \tag{6}$$

where

$$\pi_0 = C \left\{ \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} - \sum_{k=1}^{N^{(p)}} \lambda_k^{(p)} \sum_{\mathbf{D}^{(p)} \in \tilde{D}^{(p)}} \sum_{l=1}^{|F^{(p)}|} P_l^{(p)} Q(k, \text{F}_l^{(p)} | \mathbf{D}^{(p)}) q_{\mathbf{D}^{(p)}} \right\}. \tag{7}$$

The term π_0 in (7) represents the rate of successful packet transmissions by PB corresponding to the vector $\boldsymbol{q}$. The right hand side (RHS) of (7) is the product of the rate of primary user requests served by PB and C, the number of successful packet transmissions per served request. The constraint (5) is the stability constraint at PB: the rate of packet transmissions by PB cannot exceed 1. Constraint (6) represents the stability requirement at SBs. The left hand side (LHS) of (6) represents total arrival rate into all the queues in a given SB; the RHS of (6) represents a feasible transmission rate offered to that base station.

The region Λ contains the set of $\boldsymbol{\lambda}^{(p)}$ and $\boldsymbol{\lambda}^{(s)}$ supportable without cooperation as the all-zero primary availability matrix (i.e., one whose every component is 0) is in the set $\tilde{D}^{(p)}$.

We now simplify the description of the region Λ. Intuitively, we observe that SBs create maximum transmission opportunities for themselves by transmitting as much primary traffic as possible. This means that the region Λ can be described by only considering policies in which each SB, in every period, caches exactly $B-1$ most popular primary files and admits all user requests for those files. We state this formally in Lemma 1.

Lemma 1: The region Λ can be defined by considering, in (5) and (6), only the vector $\boldsymbol{q}$ for which $q_{\mathbf{D}^*} = 1$ and $q_{\mathbf{D}} = 0$ for every primary availability matrix $\mathbf{D} \neq \mathbf{D}^*$ where

$$D^*_{n,j} = \begin{cases} 1, & \text{if } 1 \leq n \leq B-1, \quad 1 \leq j \leq M \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

Now we compare the achievable capacity region Λ with the general capacity region Λ_{gen} for a network in which all SBs are non-interfering. The region Λ_{gen} consists of all request generation rate vectors for which the network is stable under any stationary scheduling and caching algorithm, and not just the restricted set of algorithms that satisfy constraint **C1**. Clearly, $\Lambda_{\text{gen}} \supseteq \Lambda$. In particular, we compare the total secondary request generation rate that can be maximally supported by each SB under the restricted set of algorithms and that under all stationary algorithms. We quantify the performance gap, measured in terms of supportable secondary request generation rates, between these two sets of policies as follows.

For a given $\boldsymbol{\lambda}^{(p)}$, let $\lambda^{(s)}_{j,\text{max,ach}}(\boldsymbol{\lambda}^{(p)})$ and $\lambda^{(s)}_{j,\text{max,gen}}(\boldsymbol{\lambda}^{(p)})$ denote the maximum total secondary request generation rate that can be supported (i.e., all queues in the network are stable) at base station SB_j, $j \in \{1, \ldots, M\}$, under the restricted set of policies and under all policies respectively. Then $\lambda^{(s)}_{j,\text{max,ach}}(\boldsymbol{\lambda}^{(p)})$ is lower bounded as follows.

Lemma 2: Consider a network in which all SBs are non-interfering. For any primary request generation rate vector $\boldsymbol{\lambda}^{(p)}$ which is present in Λ (i.e., there exists $\boldsymbol{\lambda}^{(s)}$ such that $\left(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)}\right) \in \Lambda$) we have for every $j \in \{1, \ldots, M\}$,

$$\begin{aligned} \lambda^{(s)}_{j,\text{max,ach}}(\boldsymbol{\lambda}^{(p)}) \geq\ & \lambda^{(s)}_{j,\text{max,gen}}(\boldsymbol{\lambda}^{(p)}) \\ & - (\frac{1}{p} - 1) \sum_{k:\text{PU}_k \in \phi_j^{(p)}} \lambda_k^{(p)} P_B^{(p)} \\ & - \frac{\sum_{n=1, n \neq j}^{M} \sum_{k:\text{PU}_k \in \phi_n^{(p)}} \lambda_k^{(p)} P_B^{(p)}}{p}. \end{aligned} \tag{9}$$

Extension to case of different file sizes

The analysis can be extended to the case of unequal file sizes as follows. First, in the system model, the queue evolution will change. For every new request, the queue size will be incremented by a parameter different from C corresponding to the size of the file. Second, we will need to redefine the achievable capacity region by modifying constraint **C1**. For example, we can redefine **C1** so that any secondary file can always be stored at an SB independent of state of queues for primary files. Third, we will need to modify (4)-(7) that define the region Λ. In particular, we replace each occurrence of the terms $C\lambda_i^{(p)}$ and $C\lambda_k^{(s)}$ (where $1 \leq i \leq N^{(p)}$, $1 \leq k \leq N^{(s)}$) in (4)-(7) with terms $\sum_{m=1}^{|F^{(p)}|} P_m^{(p)} C_m^{(p)} \lambda_i^{(p)}$ and $\sum_{n=1}^{|F^{(s)}|} P_n^{(s)} C_n^{(s)} \lambda_k^{(s)}$ where $C_m^{(p)}$ and $C_n^{(s)}$ denote the number of packets required to serve file $\text{F}_m^{(p)}$ and $\text{F}_n^{(s)}$ respectively ($1 \leq m \leq |F^{(p)}|$, $1 \leq n \leq |F^{(s)}|$). Fourth, we will modify Lemma 1 based on the assumption used in modified **C1**. For example, suppose SBs always reserve a fixed amount of cache space for secondary files. Then we can restate Lemma 1 as one in which each SB always stores the most popular primary files in its remaining cache space. Using Lemma 1 we can derive the counterparts of FPCP, VPCP and MCCP with performance guarantees similarly as in this paper.

Gain due to Cooperation

In order to quantify the gain due to cooperation we consider a symmetric network where every primary (respectively secondary) user generate requests at same rate, all small cells contain equal number of primary (respectively secondary) users, there is no primary user which is outside all small cells and all secondary base stations are non-interfering. Given SU_n (where $1 \leq n \leq N^{(s)}$) requests content at rate $\lambda_n^{(s)}$, the maximum request generation rate for PU_m (where $1 \leq m \leq N^{(p)}$) with cooperation, denoted as $\lambda^{(p)}_{m,\text{coop}}(\lambda_n^{(s)})$ can be obtained, using

(4)-(7) and Lemma 1, as,

$$\lambda^{(p)}_{m,\text{coop}}(\lambda^{(s)}_n) = \max\{(\frac{1}{CN^{(p)}} - \frac{\lambda^{(s)}_n\rho}{M})\frac{1}{\frac{1}{p}(1 - \sum_{l=1}^{B-1} P^{(p)}_l) + \frac{1}{M}\sum_{l=1}^{B-1} P^{(p)}_l}, 0\} \quad (11)$$

and that without cooperation, denoted as $\lambda^{(p)}_{m,\text{no-coop}}(\lambda^{(s)}_n)$ can be obtained as,

$$\lambda^{(p)}_{m,\text{no-coop}}(\lambda^{(s)}_n) = \max\{p(\frac{1}{CN^{(p)}} - \frac{\lambda^{(s)}_n\rho}{M}), 0\}$$

where $\rho = \frac{N^{(s)}}{N^{(p)}}$ is the ratio of number of secondary to primary users. For fixed $N^{(p)}$ both $\lambda^{(p)}_{m,\text{coop}}(\lambda^{(s)}_n)$ and $\lambda^{(p)}_{m,\text{no-coop}}(\lambda^{(s)}_n)$ decreases with increasing ρ. This is expected as higher ρ implies larger number of secondary users transmitting at rate $\lambda^{(s)}_n$ leading to lower supportable $\lambda^{(p)}_{m,\text{coop}}$. Cooperation benefits a primary user for given $\lambda^{(s)}_n$ if $\lambda^{(p)}_{m,\text{coop}}(\lambda^{(s)}_n) \geq \lambda^{(p)}_{m,\text{no-coop}}(\lambda^{(s)}_n)$.

Similarly, given PU_m requests content at rate $\lambda^{(p)}_m$, the maximum request generation rate for SU_n with cooperation, denoted as $\lambda^{(s)}_{n,\text{coop}}(\lambda^{(p)}_m)$ is,

$$\lambda^{(s)}_{n,\text{coop}}(\lambda^{(p)}_m) = \max\{\frac{M}{CN^{(s)}} - \frac{M\lambda^{(p)}_m}{\rho p}(1 - \sum_{l=1}^{B-1} P^{(p)}_l) - \frac{\lambda^{(p)}_m}{\rho}\sum_{l=1}^{B-1} P^{(p)}_l, 0\}$$

and that without cooperation, denoted as $\lambda^{(s)}_{n,\text{no-coop}}(\lambda^{(p)}_m)$ is,

$$\lambda^{(s)}_{n,\text{no-coop}}(\lambda^{(p)}_m) = \max\{\frac{M}{CN^{(s)}} - \frac{M\lambda^{(p)}_m}{\rho p}, 0\}.$$

Again, $\lambda^{(s)}_{n,\text{coop}}(\lambda^{(p)}_m)$ and $\lambda^{(s)}_{n,\text{no-coop}}(\lambda^{(p)}_m)$ decreases with increasing number of secondary users. This is expected as higher number of secondary users means lower transmission opportunities per user. Clearly, cooperation benefits a secondary user for given $\lambda^{(p)}_m$ if $\lambda^{(s)}_{n,\text{coop}}(\lambda^{(p)}_m) \geq \lambda^{(s)}_{n,\text{no-coop}}(\lambda^{(p)}_m)$.

IV. Fixed Primary Caching Policy

In this section we present FPCP, an algorithm that stabilizes all queues in the network for every request generation rate vector in the interior of the achievable capacity region Λ. The algorithm is constructed using Lyapunov drift techniques that are widely used to develop efficient scheduling algorithms in communication networks. In order to construct the algorithm we first find the set of files cached at the SBs in each period. Given the set of files currently cached, in FPCP we schedule packet transmissions according to a modified backpressure policy. We formally state the algorithm in Table 2 and analyze its stability performance in Theorem 1. Finally, we briefly discuss a drawback of the FPCP algorithm.

We obtain the set of files cached at the SBs in each period as follows. First, recall, according to Lemma 1, the achievable capacity region Λ can be described by considering only those policies under which, in every period, each SB caches exactly $B-1$ most popular primary files and always admits requests corresponding to those files. Therefore, in the construction of FPCP we consider only those policies. Next we refer to the aggregate of all primary queues in SB_k as the *aggregate primary queue* in SB_k for every $k \in \{1,\ldots,M\}$. Clearly, the length of the aggregate primary queue in SB_k, denoted as $U^{(p)}_k(t)$, is the sum of the length of all primary queues i.e., $U^{(p)}_k(t) = \sum_{f\in F^{(p)}} U^{(p)}_{f,k}(t)$. We then apply the Lyapunov drift technique to find the set of cached secondary files in every period. We denote as $t_{r,j}$ the j'th slot in r'th period (where $r \in \{1,2,\ldots,\}$ and $j \in \{1,\ldots,T\}$) and form the Lyapunov function $L_1(r) \triangleq \sum_{k=1}^{M}\{\sum_{f\in F^{(s)}} (U^{(s)}_{f,k}(t_{r,1}))^2 + (U^{(p)}_k(t_{r,1}))^2\}$. We find the set of secondary files to be cached at each SB in the r'th period by minimizing the upper bound of the conditional drift $\Delta_1(r) \triangleq E[L_1(r+1) - L_1(r)|\mathbf{U}^{(p)}(t_{r,1}), \mathbf{U}^{(s)}(t_{r,1})]$. We perform this minimization over all polices X under which each secondary base station caches exactly $B-1$ most popular primary file in every period. Therefore under FPCP, for every strictly positive integer r, in the r'th period each base station SB_k (where $k \in \{1,\ldots,M\}$) can cache only one secondary file, denoted as $f^*_k(r)$. The expression of $f^*_k(r)$ is obtained as,

$$f^*_k(r) \in \operatorname*{argmax}_{f\in F^{(s)}} U^{(s)}_{f,k}(t_{r,1}) \quad \forall k = 1,2,\ldots,M. \quad (12)$$

Given the set of files cached at each base station in any period, in FPCP we schedule packet transmissions from base stations according to a modified backpressure policy. At any time slot when PB is not transmitting, we schedule packet transmissions by solving a max-weight optimization problem of instantaneous queue lengths of the cached files in all base stations.

Table 2: FPCP Algorithm

For every strictly positive integer r following steps are performed in the r'th period:

1) **Caching Scheme:** At the beginning of the period, every secondary base station caches the $B-1$ most popular primary files. In addition, every secondary base station SB_k (where $k \in \{1,\ldots,M\}$) caches the secondary file $f^*_k(r)$.
2) **Scheduling for PB:** At any time slot t PB transmits a packet corresponding to the Head-of-Line (HOL) packet request at the queue of highest length (in case of ties, pick one arbitrarily). Mathematically, $\mu^{\text{FPCP}}_{f,0}(t) = 1$ if $U^{(p)}_{f,0}(t) > 0$ and $U^{(p)}_{f,0}(t) \geq U^{(p)}_{\bar{f},0}(t)$ for every $\bar{f} \in F^{(p)} - f$ and is 0 otherwise.
3) **Request Admission and Scheduling Policy at SBs:** Each secondary base station SB_k, $k \in \{1,\ldots,M\}$, admits every request for a cached primary file from a primary user within its transmission-range, i.e., for all t, $\chi^{(p)}_{f,k}(t)$ equals 1 if and only if (iff) $f \in \{\text{F}^{(p)}_1,\ldots,\text{F}^{(p)}_{B-1}\}$. If PB is not transmitting at $t_{r,j}$, then obtain the set of base stations that can transmit at $t_{r,j}$, denoted as $E^*_{\text{FPCP}}(t_{r,j})$, by solving the following max-weight problem:

$$E^*_{\text{FPCP}}(t_{r,j}) \in \operatorname*{argmax}_{E\in\tilde{E}} \left(\left(\max\left\{ U^{(p)}_k(t_{r,j}), U^{(s)}_{f^*_k(r),k}(t_{r,j}) \right\} \right)_{k=1}^{M} \right)^T E. \quad (13)$$

Suppose, according to $E^*_{\text{FPCP}}(t_{r,j})$ some SB_k is allowed to transmit at this time slot. Then we determine which packet to transmit from SB_k in the following manner.

We transmit the packet corresponding to the HOL packet request at the queue of secondary file $f_k^*(r)$ if $U^{(s)}_{f_k^*(r),k}(t_{r,j})$ is greater than or equal to $U_k^{(p)}(t_{r,j})$. Otherwise transmit the packet corresponding to the HOL packet request at the aggregate primary queue in SB_k. Mathematically, given k'th component of $E^*_{\text{FPCP}}(t_{r,j})$ is 1, $\mu_k^{\text{FPCP},(p)}(t_{r,j}) = 1$, $\mu^{\text{FPCP}}_{f_k^*(r),k}(t_{r,j}) = 0$ if $U_k^{(p)}(t_{r,j}) \geq U^{(s)}_{f_k^*(r),k}(t_{r,j})$; $\mu_k^{\text{FPCP},(p)}(t_{r,j}) = 0$, $\mu^{\text{FPCP}}_{f_k^*(r),k}(t_{r,j}) = 1$ otherwise. Here, $\mu_k^{X,(p)}(\tau)$ denotes the transmission rate offered to SB_k, under policy X, to transmit a packet from its aggregate primary queue at time slot τ.

Note that FPCP does not require knowledge of request generation rates.

We show that FPCP stabilizes all queues in the network for every request generation rate vector in the interior of the achievable capacity region:

Theorem 1: FPCP stabilizes all queues in the network for each $(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)}) \in$ Interior(Λ).

While the FPCP algorithm stabilizes every queue for all request generation rate vectors in the interior of Λ, it allows each SB to cache only one secondary file per period. Our simulation results in Section VII show that this can adversely affect the delay performance of secondary users, especially when primary request generation rates are low and it is not necessary to cache $B-1$ primary files per SB per period in order to stabilize the queues. Hence, in Section V we present the VPCP algorithm which dynamically varies the number of cached primary files at SBs in every period according to instantaneous queue lengths. However, the guaranteed stability region under VPCP is lower than that under FPCP.

V. Variable Primary Caching Policy

In this section we present VPCP, an algorithm under which the network controller dynamically determines the number of primary files to be cached in each period. Furthermore, primary packet requests are satisfied ahead of secondary ones at every SB. We first provide an overview of the motivation behind construction of this algorithm; in the following subsections we discuss the construction methodology and present a guaranteed stability region under VPCP.

The VPCP algorithm addresses the difficult problem of serving primary packets at each base station ahead of secondary ones while also attempting to minimize the number of cached primary files and maintain stability of all queues. The difficulty of the problem lies in the coupled nature of the evolution of primary and secondary queues. In particular, scheduling decisions for secondary packets from any base station depend on whether or not the primary queues in that base station are empty. Evolution of primary queues in turn depend on cooperative caching decisions since any base station can only transmit packets of a cached primary file. If we view the primary queue lengths as the system state, then the problem is a constrained Markov Decision Problem where the system state itself evolves according to the control decisions. One way to solve such problems efficiently is by using renewal frame based optimization techniques. Such techniques can solve certain constrained Markov Decision Problems without suffering from the pitfalls associated with conventional solutions to those problems, e.g., requiring extensive knowledge of system dynamics, large convergence times [9]. Renewal frame based techniques are based on partitioning the time-line into distinct collection of contiguous time slots called "frames" and then making control decisions in the beginning of each new frame. Length of each frame is variable and depends on the control decisions taken at the beginning of the frame. In Section V-A we discuss how we partition the timeline into renewal frames in our model.

However, we cannot use the renewal frame based optimization techniques directly due to some restrictions imposed by the system model. In particular, in renewal frame based methods, control decisions (e.g. caching) are made only at the beginning of every variable length frame. On the other hand, according to our system model, caching decisions take place at the beginning of every fixed length cache-refresh period which may not occur at the beginning of a new frame.

We address the challenge of applying renewal frame method to this scenario by constructing the VPCP algorithm in two steps. First, as discussed in Section V-B, in each period we estimate, the number of primary files that would be cached at SBs by a renewal frame based caching and scheduling policy *had* the beginning of the period coincided with beginning of a new frame. This renewal frame based method makes caching decisions only at the beginning of a frame by minimizing the ratio of a drift plus penalty expression over a frame, and the expected length of the frame. Then, as discussed in Section V-C, we use those estimates to construct VPCP by first determining the actual number of primary files cached at each SB in every period and then the scheduling policy. Because of this two-step construction process, VPCP is not an optimal renewal frame based optimization policy, i.e., it does not provide any optimality guarantee on the average number of primary files cached at SBs.

In order to render tractable analysis of its stability performance, the VPCP algorithm is constructed under certain restrictions. First, under this algorithm only a fixed set of non-interfering SBs, denoted without loss of generality as $\text{SB}_1, \ldots, \text{SB}_G$ respectively (where $G \leq M$), cooperate with the primary network by caching primary files, admitting and serving primary requests. Secondly, the policy VPCP is constructed such that it satisfies constraints **C1** (described in Section III) as well as two additional scheduling constraints described in Section V-B. In Section V-D we present a guaranteed stability region for this algorithm.

A. Renewal Frame Techniques for Cognitive Networks

In this subsection we discuss how we apply the renewal frame technique. We define the system state to be the sum of the length of all primary queues in the network, similar to the way primary user's channel occupancy process was used as system state in [9][5]. Defining the system state in this way is useful because scheduling decision for secondary packets depends on whether this system state is zero or not. Each

[5]Note that in [9] the authors did not study cooperative caching in cognitive radio networks.

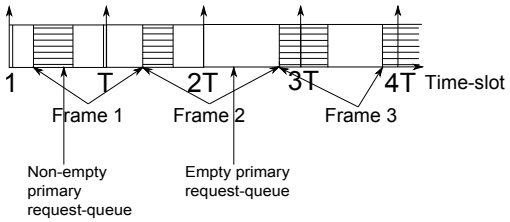


Fig. 2. Partition of time-line into frames. Shaded region shows time slots for which at least one primary queue in the network is non-empty. An arrow indicates beginning of a new period.

frame begins when the system state changes, i.e., the sum of all primary queue lengths transitions from zero to a non-zero value. Each frame consists of two distinct phases, one in which at least one primary queue in the network is non-empty, and one in which all primary queues in the network are empty. Fig. 2 shows an example of such partition.

B. Caching Decisions by a Renewal Frame Based Optimization Policy

In this subsection we estimate the number of primary files, $\hat{n}_k^*(r)$, that would be cached at each cooperative secondary base station SB_k (where $k \in \{1,\ldots,G\}$) at the beginning of the r'th period (where $r \in \{1,2,\ldots,\}$) under a renewal frame based optimization policy. The optimization policy tries to minimize the average number of primary files cached while maintaining stability of all queues. The policy also transmits primary packets with higher priority than secondary ones from each cooperative SB. The policy assumes a new frame began at time slot $t_{r,1}$, irrespective of the actual length of primary queues, and subsequent caching decisions take place only at the beginning of every frame. In the rest of this sub-section, we assume those assumptions are indeed true and that the r''th frame (where $r' \in \{1,2,\ldots,\}$) began at time slot $t_{r,1}$. Next we discuss how we obtain the $\hat{n}_k^*(r)$ variables.

We obtain the $\hat{n}_k^*(r)$ variables by minimizing the ratio of a conditional drift plus penalty expression to expected frame length as follows. First, we consider the Lyapunov function of secondary queue lengths at beginning of each frame, $L_2(r') \triangleq \sum_{l=1}^{M} \sum_{f\in F^{(s)}} (U_{f,l}^{(s)}(\tilde{t}_{r',1}))^2$ and its conditional drift, $\Delta_2(r') \triangleq E[L_2(r'+1) - L_2(r')|\mathbf{U}^{(s)}(\tilde{t}_{r',1})]$. Here, $\tilde{t}_{n,j}$ denotes the j'th, $j \in \{1,\ldots,\tilde{t}_{n+1,1}-1\}$, time slot in the n'th, $n \in \{1,2,\ldots,\}$, frame. Next we form the sum of $\Delta_2(r')$ and a penalty term: $VE[\sum_{k=1}^{G} \sum_{\tau=\tilde{t}_{r',1}}^{\tilde{t}_{r'+1,1}-1} \hat{n}_k(r')|\mathbf{U}^{(s)}(\tilde{t}_{r',1})]$, where V is a positive constant and $\hat{n}_k(r')$ denotes the number of primary files cached at SB_k in the r''th frame. The penalty parameter V reflects the trade-off between queue stability and number of primary files cached. High V implies less primary files should be cached and vice-versa. We then obtain a simplified upper bound to this sum by considering only those $\boldsymbol{\lambda}^{(p)}$ for which the primary network can stabilized even without cooperation. Finally, we obtain the $\hat{n}_k^*(r)$ variables by finding the set of primary files cached at base stations in the r''th frame, and the scheduling scheme used throughout the r''th frame, that minimizes the ratio of this upper bound to that of the expected frame-length. By simplifying this ratio we obtain the following expression under policy X:

$$V\sum_{k=1}^{G}\hat{n}_k(r') - \frac{E[\sum_{l=1}^{M}\sum_{f\in F^{(s)}} U_{f,l}^{(s)}(\tilde{t}_{r',1}) \sum_{\tau=\tilde{t}_{r',1}}^{\tilde{t}_{r'+1,1}-1} \mu_{f,l}^{X}(\tau)|\mathbf{U}^{(s)}(\tilde{t}_{r',1})]}{E[(\tilde{t}_{r+1,1}-\tilde{t}_{r,1})|\mathbf{U}^{(s)}(\tilde{t}_{r',1})]}. \quad (14)$$

Minimizing the expression in (14) reflects the tradeoff between stability of queues versus caching less primary files at SBs. Caching more primary files at SBs minimizes the fraction term in (14), while it increases the penalty term: $V\sum_{k=1}^{G}\hat{n}_k(r')$.

In order to precisely define a guaranteed stability region for the VPCP algorithm (described in Section V-C), we minimize the expression in (14) by considering only caching and scheduling algorithms that satisfy **C1** and two scheduling constraints **C2** and **C3** that we state next.

> **C2:** At every slot PB transmits a packet iff it has at least one non-empty primary queue and all cooperative SBs have only empty primary queues. A cooperative SB transmits a secondary packet only if it has all empty primary queues.

Constraint **C2** ensures that primary packets are transmitted ahead of secondary packets from each cooperative SB. Furthermore, this constraint also simplifies construction of a guaranteed stability region for VPCP as it ensures that the events of primary packet transmission from cooperative SBs are independent from base station to base station.

> **C3:** No non-cooperative SB simultaneously transmits secondary packets when some cooperative SB is transmitting a primary packet.

Constraint **C3** simplifies construction of a guaranteed stability region for VPCP by ensuring the following event cannot occur: "concurrent transmission of a secondary and primary packet from a non-cooperative SB and a cooperative SB respectively".

In Appendix C of [19] we find an approximate solution to the problem of minimizing the expression in (14) by considering only policies that satisfy constraints **C1**, **C2**, **C3** and under some additional assumptions. In particular, we obtain an estimate of the number of most popular primary files, $\hat{n}_k^*(r)$, that ought to be cached at each cooperative secondary base station SB_k in the r'th period if the expression in (14) is minimized.

C. Description of the VPCP Algorithm

In this subsection we use the $\hat{n}_k^*(r)$ variables to construct the VPCP algorithm. First, we discuss the problem with simply caching $\hat{n}_k^*(r)$ most popular primary files at each cooperative base station SB_k, $k \in \{1,\ldots,G\}$, in the r'th period. Then we discuss how the caching scheme in VPCP resolves this problem. Finally, we state the VPCP algorithm in Table 3.

We cannot simply cache $\hat{n}_k^*(r)$ most popular primary files at each cooperative base station SB_k , $k \in \{1,\ldots,G\}$, in the r'th period for every $r \in \{1,2,\ldots\}$. This is because we calculated $\hat{n}_k^*(r)$ estimates were assuming that a new frame began at time slot $t_{r,1}$ regardless of the actual length of primary queues at $t_{r,1}$. Hence, it is possible that at $t_{r,1}$, the queue length in SB_k for some primary file f, not one of the $\hat{n}_k^*(r)$ most popular primary files, is non-zero. Not caching file f in the r'th period can lead to loss of transmission opportunities for SB_k since a

cooperative SB transmits a secondary packet only if all of its primary queues are empty.

The VPCP algorithm resolves the above problem of lost transmission opportunities due to lack of cached primary files by ensuring that in the r'th period, each cooperative SB caches every primary file corresponding to its non-empty primary queues. We guarantee this by having the network controller cache $n_k^*(r)$ most popular primary files and admit primary requests at each SB_k, $k \in \{1, \ldots, G\}$, in the r'th period (for every $r \in \{1, \ldots, \}$) as follows:

1) Suppose no new frame began during the $(r-1)$'th period, i.e., at least one primary queue in the network was non-empty for all time slots in the $(r-1)$'th period. Then, at time slot $t_{r,1}$, cache at least as many most popular primary files at SB_k as cached in the $(r-1)$'th period, i.e., $n_k^*(r) = \max\{n_k^*(r-1), \hat{n}_k^*(r)\}$. For the very first period we have, $n_k^*(1) = \hat{n}_k^*(1)$.
2) Otherwise, suppose a new frame began during $(r-1)$'th period, i.e., at some time slot $t_{r-1,j}$ (where $j \in \{1, 2, \ldots, T\}$) all primary queues in the network became empty for the first time in $(r-1)$'th period. Then from time slot $t_{r-1,j+1}$ till the end of the $(r-1)$'th period, SB_k only admits primary requests corresponding to $\hat{n}_k^*(r-1)$ most popular primary files. By admitting only those requests we ensure that only queues of $\hat{n}_k^*(r-1)$ most popular primary files can be non-empty in SB_k at $t_{r,1}$. At $t_{r,1}$, SB_k caches at least $\hat{n}_k^*(r-1)$ most popular primary files i.e $n_k^*(r) = \max\{\hat{n}_k^*(r-1), \hat{n}_k^*(r)\}$, if at least one primary queue is non-empty at $t_{r,1}$; otherwise, it caches $\hat{n}_k^*(r)$ most popular primary files i.e, $n_k^*(r) = \hat{n}_k^*(r)$.

After obtaining the set of cached primary files, the network controller fills rest of the cache at each SB with secondary files based on their instantaneous queue lengths.

In Table 3 we formally present the VPCP algorithm. In VPCP, each cooperative SB serves queued primary requests with higher priority than the secondary ones; the algorithm satisfies scheduling constraints **C2** and **C3**. When all primary queues are empty, VPCP schedules transmissions from SBs according to a modified backpressure rule.

Table 3: VPCP Algorithm

For every strictly positive integer r following steps are performed in the r'th period:

1) **Caching Policy:** At the beginning of the period, for every cooperative base station SB_k, $k \in \{1, \ldots, G\}$, compute $\hat{n}_k^*(r)$ and then $n_k^*(r)$. Each cooperative base station SB_k caches the $n_k^*(r)$ most popular primary files; it also caches the $B - n_k^*(r)$ secondary files with highest queue lengths in SB_k at $t_{r,1}$. Every non-cooperative secondary base station SB_l, $l \in \{G+1, \ldots, M\}$, caches the B secondary files with highest queue lengths in SB_l at $t_{r,1}$.
2) **Request Admission and Scheduling Policy for Primary Users:** Suppose, at least one primary queue is non-empty at $t_{r,1}$. Consider all time slots in the r'th period until either the end of the period or the first time slot in which all primary queues become empty, whichever is earliest. In these time slots each SB_k, $k \in \{1, \ldots, G\}$, admits requests corresponding to $n_k^*(r)$ most popular primary files from the $\phi_k^{(p)}$ primary users. Suppose at time slot t_{r,j^*} (where $1 < j^* \leq T$) all primary queues become empty for the first time. Then from time slots t_{r,j^*} until end of the period, SB_k admits requests corresponding to $\hat{n}_k^*(r)$ most popular primary files from the $\phi_k^{(p)}$ primary users.
In any time slot $t_{r,j}$ (where $j \in \{1, \ldots, T\}$), SB_k transmits the packet corresponding to the HOL packet request at its primary queue of highest length; in case of ties, pick one arbitrarily. Mathematically for every $k \in \{1, \ldots, G\}$ and primary file f, $\mu_{f,k}^{\text{VPCP}}(t) = 1$, if $U_{f,k}^{(p)}(t) > 0$, $U_{f,k}^{(p)}(t) \geq U_{\bar{f},k}^{(p)}(t)$ for every $\bar{f} \in F^{(p)} - f$; it equals 0 otherwise.
If all primary queues in each SB is empty and the queue in PB is non-empty, identify the queue of highest length in PB and transmit the packet corresponding to its HOL packet request from PB. Mathematically, $\mu_{f,0}^{\text{VPCP}}(t) = 1$, if $U_{f,0}^{(p)}(t) > 0$, $U_{f,0}^{(p)}(t) \geq U_{\bar{f},0}^{(p)}(t)$ for every $\bar{f} \in F^{(p)} - f$ and $U_{f,k}^{(p)}(t) = 0$ for every $k \in \{1, \ldots, G\}$; it is 0 otherwise.
3) **Scheduling Policy for Secondary Users:** Suppose, at time slot $t_{r,j}$ all the primary queues of cooperative base station SB_k are empty but not at another cooperative base station. Then find the queue of the cached secondary file with largest instantaneous backlog in SB_k; SB_k transmits the packet corresponding to the HOL packet request in this queue. Mathematically, for every $k \in \{1, \ldots, G\}$, we have $\mu_{f,k}^{\text{VPCP}}(t_{r,j}) = 1$ if $f \in \underset{f_1 \in H_k^{(s)}(t_{r,j})}{\text{argmax}} U_{f_1,k}^{(s)}(t_{r,j})$, $\max_{f_2 \in F^{(p)}} U_{f_2,k}^{(p)}(t_{r,j}) = 0$ and $\max_{f_3 \in F^{(p)}} U_{f_3,l}^{(p)}(t_{r,j}) > 0$ for some $l \in \{1, \ldots, G\}$.
Otherwise, if all the primary queues are empty at $t_{r,j}$, then transmit secondary packets according to a modified backpressure rule. First find the activation vector $E_{\text{VPCP}}^*(t_{r,j})$ as

$$E_{\text{VPCP}}^*(t_{r,j}) \in \underset{E \in \tilde{E}}{\text{argmax}} \left((\max_{f \in H_l^{(s)}(t_{r,j})} U_{f,l}^{(s)}(t_{r,j}))_{l=1}^{M} \right)^T E. \quad (15)$$

If base station SB_l is scheduled to transmit according to $E_{\text{VPCP}}^*(t_{r,j})$, then SB_l transmits a packet corresponding to the HOL packet request in the secondary queue of highest length (in case of ties, pick one arbitrarily) i.e., $\mu_{f,l}^{\text{VPCP}}(t_{r,j}) = 1$ if $f \in \underset{f_1 \in H_l^{(s)}(t_{r,j})}{\text{argmax}} U_{f_1,l}^{(s)}(t_{r,j})$.

D. Guaranteed Stability Region

Next, we find a guaranteed stability region for the VPCP algorithm. First we construct this region, denoted as Λ^{VPCP}. Then, in Theorem 2 we show that VPCP algorithm indeed stabilizes all queues for every request generation rate vector within the guaranteed stability region.

The region Λ^{VPCP}, consists of all vectors $\boldsymbol{\lambda}^{(p)}$

supportable without cooperation[6] and a set of secondary request generation rate vectors $\Lambda^{(s)}(\boldsymbol{\lambda}^{(p)})$. Mathematically, the region Λ^{VPCP} is defined as the set $\left\{(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)}) : \boldsymbol{\lambda}^{(p)} \in \Lambda_0^{(p)}, \boldsymbol{\lambda}^{(s)} \in \text{ Interior } \left(\Lambda^{(s)}(\boldsymbol{\lambda}^{(p)})\right)\right\}$ where $\Lambda_0^{(p)}$ denotes the set of $\boldsymbol{\lambda}^{(p)}$ supportable without cooperation. For a given vector $\boldsymbol{\lambda}^{(p)}$, the set $\Lambda^{(s)}(\boldsymbol{\lambda}^{(p)})$ defines the set of all $\boldsymbol{\lambda}^{(s)}$ supportable under caching and scheduling algorithms that satisfy constraints **C1**, **C2** and **C3**. The exact definition of the region $\Lambda^{(s)}(\boldsymbol{\lambda}^{(p)})$ is provided in Appendix C of [19].

Theorem 2: VPCP stabilizes all queues in the network for all $(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)}) \in \Lambda^{\text{VPCP}}$.

Clearly, the guaranteed stability region is greater than the capacity region without cooperation but is smaller than the achievable capacity region Λ. However, with respect to $\boldsymbol{\lambda}^{(p)}$, the guaranteed stability region under VPCP remains the same as for the case without cooperation.

VI. Network with Multiple Channels

In this section we extend our analysis to a network where all base stations can transmit on multiple orthogonal channels. For simplicity we consider the case where all SBs are non-interfering. First in Section VI-A we present the system model. Then in Sections VI-B and VI-C we discuss an achievable capacity region and the MCCP algorithm that stabilizes all queues for every request generation rate vector within this region respectively.

A. System Model

There are J orthogonal channels denoted as $c_1, \ldots, c_J$. Every base station also has J antennas and can transmit on each channel simultaneously. Each user has one antenna and can only communicate with one base station on a single channel at any time slot. The number of primary and secondary users in each small-cell is greater than the number of channels. Next, we present details about the transmission model, interference model, caching and scheduling model.

1) Transmission Model

All channels have identical quality. For each channel, the transmission model is same as that of the single channel network. Primary users can not switch channels using software-defined radio as they are non-cognitive. Hence, each primary user is associated with a unique channel on which it receives packets in all time slots. Let γ_j denote the set of primary users associated with channel c_j (where $\gamma_j \subseteq \{\text{PU}_1, \ldots, \text{PU}_{N^{(p)}}\}$ and $j \in \{1, \ldots, J\}$). Each secondary user can receive packets from SBs on different channels in different time slots.

2) Interference Model

The interference model slightly differs from that in the single channel case as each base station may transmit on multiple channels at any time slot but each user can receive transmission on only one channel. So, we define a link between base station a (where $a \in \{\text{PB}, \text{SB}_1, \ldots, \text{SB}_M\}$) and user b (where $b \in \{\text{PU}_1, \ldots, \text{PU}_{N^{(p)}}, \text{SU}_1, \ldots, \text{SU}_{N^{(s)}}\}$), located within a's transmission range, on channel c (where $c \in \{c_1, \ldots, c_J\}$) as the 3-tuple (a, b, c). Note that since primary users can communicate on a fixed unique channel, the 3-tuple (a, b, c) is not a feasible link for a primary user $b \in \gamma_j$ (where $j \in \{1, \ldots, J\}$) if channel c is not c_j. A link (a, b, c) is said to be active if a transmits to b on channel c; otherwise it is inactive. The link (a, b, c) is active iff:

1) Base station a does not transmit to user b on another channel, i.e., $(a, b, \bar{c})$ is inactive for every $\bar{c} \in \{c_1, \ldots, c_J\} - c$.
2) Base station a does not transmit to another user on the same channel, i.e., $(a, \bar{b}, c)$ is inactive for every user $\bar{b} \in \{\text{PU}_1, \ldots, \text{PU}_{N^{(p)}}, \text{SU}_1, \ldots, \text{SU}_{N^{(s)}}\} - b$ within a's transmission range.
3) If a is an SB, then PB is not transmitting on channel c.

3) Service Discipline for Primary Users

The network controller queues user requests at either an SB or at the primary base station in the same way as the single channel model in Section II. However now each base station maintains queues for every (primary user, primary file) pair corresponding to each user within its transmission range. This is because now a base station can use multiple antennas to transmit packets, possibly of the same file, to more than one user at a given time slot.

We denote the length of the queue at time slot t for the pair (PU_i, f) in SB_k, where $\text{PU}_i \in \phi_k^{(p)}$, $k \in \{1, \ldots, M\}$ and $f \in F^{(p)}$, as $U_{f,k,i}^{(p)}(t)$. We denote transmission rate offered to SB_k to transmit packet from this queue at time slot t as $\mu_{f,k,i}(t)$ where $\mu_{f,k,i}(t) \in \{0, 1\}$. Note that $\mu_{f,k,i}(t)$ equals 1 iff at t the link $(\text{SB}_k, \text{PU}_i, c_j)$ is active for some $j \in \{1, \ldots, J\}$ and $\mu_{\bar{f},k,i}(t) = 0$ for every primary file $\bar{f} \neq f$. Similarly, we denote the length of queue at time slot t for the pair (PU_l, f) (where $l \in \{1, \ldots, N^{(p)}\}$ and $f \in F^{(p)}$) in PB as $U_{f,0,l}^{(p)}(t)$ and the transmission rate offered to PB to transmit packet from this queue at time slot t as $\mu_{f,0,l}(t)$ where $\mu_{f,0,l}(t) \in \{0, 1\}$.

4) Service Discipline for Secondary Users

The network controller queues secondary user requests at an SB in the same way as in the single channel model. However, we maintain separate queues for every (secondary user, secondary file) pair. We denote the length of the queue at time slot t for the pair (SU_i, f) in SB_k (where $\text{SU}_i \in \phi_k^{(s)}$, $k \in \{1, \ldots, M\}$ and $f \in F^{(s)}$) as $U_{f,k,i}^{(s)}(t)$. As in the primary user case, we denote transmission rate offered to SB_k to transmit packet from this queue at time slot t as $\mu_{f,k,i}(t)$ where $\mu_{f,k,i}(t) \in \{0, 1\}$; $\mu_{f,k,i}(t)$ equals 1 iff at time slot t the link $(\text{SB}_k, \text{SU}_i, c_j)$ is active for some $j \in \{1, \ldots, J\}$ and $\mu_{\bar{f},k,i}(t) = 0$ for every secondary file $\bar{f} \neq f$.

5) Caching Model

The caching model is same as that of the single channel case.

B. Achievable Capacity Region

The achievable capacity region is the set of all request generation rate vectors for which each queue in the network is stable under any caching and scheduling policy that satisfies constraint **C1**. It is a generalization of the achievable capacity region for the single channel system model in section III. With slight abuse of notation we denote the achievable capacity region for multiple channels as Λ as well. We describe the construction of this region in Appendix E of [19].

[6]Recall, in Section V-B we derived the upper bound to the drift plus penalty expression by considering only those $\boldsymbol{\lambda}^{(p)}$ for which all primary queues are stable without cooperation.

C. Description of the MCCP Algorithm

In this section we propose MCCP, an algorithm under which primary packet transmissions from a secondary base station on any channel have higher priority over secondary packet transmissions on that channel. In other words, a secondary base station transmits secondary packets on a given channel only if it has no queued primary packet requests yet to be served via that channel. The algorithm is constructed using Lyapunov drift techniques similar to the FPCP algorithm. In order to construct the algorithm we first find the set of files cached at the secondary base stations in each period. Given the set of files currently cached, in the MCCP algorithm we schedule packet transmissions according to a modified backpressure policy. We formally state the algorithm in Table 4 and analyze its stability performance in Theorem 3.

We obtain the set of files cached at the SBs in each period as follows. First, similar to Lemma 1 we can show that the achievable capacity region Λ can be described by only considering those policies under which, in every period, each SB caches exactly the $B-1$ most popular primary files and always admits requests corresponding to those files. Therefore, in the construction of the MCCP algorithm we consider only those policies. Next we apply the Lyapunov drift technique to find the set of cached secondary files in every period. We form the Lyapunov function $L_3(r) \triangleq \sum_{k=1}^{M} \sum_{i:\mathrm{SU}_i \in \phi_k^{(s)}} \sum_{f \in F^{(s)}} (U_{f,k,i}^{(s)}(t_{r,1}))^2$ for every strictly positive integer r. We find the set of secondary files to be cached in the r'th period by minimizing the upper bound of the conditional drift $\Delta_3(r) \triangleq E[L_3(r+1) - L_3(r) | \mathbf{U}^{(s)}(t_{r,1})]$. We perform this minimization over all polices X in which each SB caches $B-1$ most popular primary files in each period and admits requests for all those files. Moreover, in order to transmit primary packets with high priority from both PB and SBs, we only consider policies X satisfying the following scheduling constraint:

> **C4)** PB transmits packet on any channel whenever it has non-empty primary queue for users associated with that channel. No SB transmits a secondary packet on any channel if it has non-empty queues for primary users associated with that channel.

Therefore, in the r'th period each secondary base station SB_k (where $k \in \{1,\ldots,M\}$) can cache only one secondary file, denoted with slight abuse of notation as $f_k^*(r)$. A simplified expression for $f_k^*(r)$ is provided in Appendix E of [19].

Table 4: MCCP Algorithm

Following steps are performed in the r'th period for every strictly positive integer r:

1) **Caching Policy:** Each base station SB_k (where $k \in \{1,\ldots,M\}$) caches $B-1$ most popular primary files in every period and admits every request for a cached primary file from a primary user within its transmission-range, i.e., for all $t \in \{1,2,\ldots,\}$, $\chi_{f,k}^{(p)}(t)$ equals 1 iff $f \in \{\mathrm{F}_1^{(p)},\ldots,\mathrm{F}_{B-1}^{(p)}\}$. At SB_k cache the secondary file $f_k^*(r)$ in r'th period.
2) **Scheduling Policy for Primary Users:** At each time slot t for each $j \in \{1,\ldots,J\}$ find the queue in PB of highest length, among all primary queues corresponding to users associated with channel c_j. PB transmits a packet corresponding to the HOL packet request from this queue on channel c_j. Mathematically, for some $\mathrm{PU}_i \in \gamma_j$, $\mu_{f,0,i}(t) = 1$ if $U_{f,0,i}^{(p)}(t) > 0$ and $U_{f,0,i}^{(p)}(t) \geq U_{\bar{f},0,\bar{i}}^{(p)}(t)$ for every $\bar{f} \in F^{(p)} - f$ and $\mathrm{PU}_{\bar{i}} \in \gamma_j - \mathrm{PU}_i$; it is 0 otherwise.
   If all queues corresponding to channel c_j at PB are empty but that at some base station SB_k is non-empty, then SB_k transmits a packet on channel c_j. This transmission corresponds to the HOL packet request at the queue of highest length in SB_k, among all primary users associated with channel c_j. Mathematically, for some $\mathrm{PU}_i \in \gamma_j$, $\mu_{f,k,i}(t) = 1$ if $U_{f,k,i}^{(p)}(t) > 0$ and $U_{f,k,i}^{(p)}(t) \geq U_{\bar{f},k,\bar{i}}^{(p)}(t)$ for every $\bar{f} \in F^{(s)} - f$ and $\mathrm{PU}_{\bar{i}} \in \gamma_j - \mathrm{PU}_i$; it is 0 otherwise.
3) **Scheduling Policy for Secondary Users:** At any given time slot τ in the r'th period, every secondary base station SB_k (where $k \in \{1,\ldots,M\}$) transmits secondary packets on those channels in which there is no primary packet transmission by either PB or SB_k. Starting from c_1 to c_J identify the m'th such available channel (where $m \in \{1,\ldots,J\}$). On this channel, SB_k transmits a packet corresponding to the HOL packet request in the m'th longest queue of the cached secondary file $f_k^*(r)$, i.e., from the queue associated with pair $(\mathrm{SU}_{\theta_{f_k^*(r),k,m}(\tau)}, f_k^*(r))$. The term $\theta_{f,k,m}(\tau)$, where $\theta_{f,k,m}(\tau) \in \{1,\ldots,N^{(s)}\}$, denotes the index of the secondary user with m'th highest queue length among all queues in SB_k associated with secondary file $f \in F^{(s)}$ at time slot τ.

We state the stability performance of the MCCP algorithm in Theorem 3.

Theorem 3: MCCP stabilizes all queues in the network for all $(\boldsymbol{\lambda}^{(p)}, \boldsymbol{\lambda}^{(s)}) \in$ Interior(Λ).

Proof: We only outline the proof in [19] as it is similar to proof of Theorem 1. ■

VII. Simulation Results

In this section we observe the performance of the FPCP and VPCP algorithms through simulations using C-programming language for a single channel network. We consider a network with 3 small SBs that are non-interfering to each other[7]. We assume there is one primary and secondary user in each of these 3 small cells; there is no other primary user in the network. We consider symmetric request generation: request generation rates by primary (and secondary) users in each small cell are equal. For convenience, we denote the sum of

[7]We do not observe performance of MCCP as for the network considered in simulations the delay performance under MCCP is likely to be similar to that of VPCP with small V; however, the supportable rates under MCCP is higher than under VPCP.

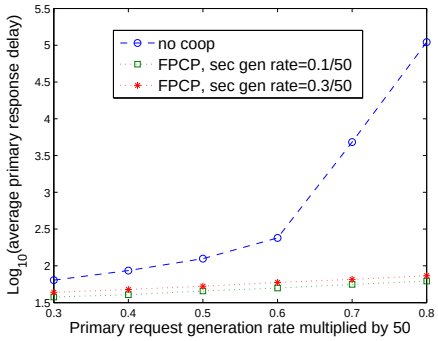


Fig. 3. Plot of base-10 logarithms of time-averaged primary response delay versus net primary request generation rate $50\lambda^{(p)}$, under no cooperation and the FPCP algorithm when $\lambda^{(s)}$ is $\frac{0.1}{50}$, $\frac{0.3}{50}$.

request generation rates of all primary users in the network as $\lambda^{(p)}$ and the request generation rates of every secondary user as $\lambda^{(s)}$ respectively. We use default values of B, C, T, p, $|F^{(p)}|$ and $|F^{(s)}|$ as 200, 50, 100, 0.7, 400 and 400 respectively. Popularity of primary and secondary files have a Zipf distribution[8] with parameter 0.8. All simulations are run for 2000000 time slots.

For comparison we use a non-cooperative protocol wherein PB serves all primary user requests. The algorithm is described as follows. At every time slot, PB transmits a packet from the queue in PB of highest length. Every SB caches B files corresponding to highest queue lengths at the beginning of every cache refresh period. An SB transmits only if all queues in PB are empty; it transmits a packet from the queue of highest length, among the set of all queues corresponding to files currently cached in that base station.

In Fig. 3 we compare the base-10 logarithm of time-averaged *response delay* for primary packets under both FPCP and the non-cooperative algorithm for different $\lambda^{(p)}$. The response delay for every transmitted packet is measured as the time between generation of the corresponding request by a user and the time slot when the packet is successfully transmitted to that user. Average primary (respectively secondary) response delay is obtained by averaging response delays of all primary (resp. secondary) packets transmitted during simulation runtime[9]. Plotting base-10 logarithm values allows us to view high values of the time-averaged delay along with smaller values. We use two values of $\lambda^{(s)}$: $\frac{0.1}{50}$ and $\frac{0.3}{50}$; for each $\lambda^{(s)}$, we vary $\lambda^{(p)}$ from $\frac{0.1}{50}$ to $\frac{0.8}{50}$. Note that maximum $\lambda^{(p)}$ supportable without cooperation is $\frac{0.7}{50}$; hence, average delay is very high without cooperation for $\lambda^{(p)} = \frac{0.7}{50}$ and $\frac{0.8}{50}$. From Fig. 3 we observe that for those $\lambda^{(p)}$, average primary response delay is lowered under FPCP as the system is stable under FPCP.

In Fig. 4 we validate the stability performance of the FPCP algorithm within the achievable capacity region Λ. First, in Fig. 4a we plot $\lambda^{(s)}_{1,\max,\text{ach}}$ and an upper bound on $\lambda^{(s)}_{1,\max,\text{gen}}$ versus $50\lambda^{(p)}$ using Lemma 2. Then in Fig. 4b we plot base-10 logarithm of time-averaged secondary response delay under FPCP for two values of $\lambda^{(p)}$. Specifically, we chose $\lambda^{(p)}$ as $\frac{0.65}{50}$ and $\frac{0.7}{50}$ as for those values there is a wide gap between $\lambda^{(s)}_{1,\max,\text{ach}}$ and upper bound on $\lambda^{(s)}_{1,\max,\text{gen}}$, as can be observed from Fig. 4a . We notice that for $\lambda^{(s)}$ lower than $\lambda^{(s)}_{1,\max,\text{ach}}$, the average secondary response delay is relatively small (i.e., network is stable) while it increases steeply for higher values.

In Fig. 5 we plot base-10 logarithm of time-averaged primary and secondary response delay and the average number of cached primary files versus $\lambda^{(s)}$ when $\lambda^{(p)}$ is $\frac{0.2}{50}$. We plot these values for the case without cooperation, under the FPCP algorithm, and under the VPCP algorithm with different values of parameter V. From Fig. 5a we observe that response delay for primary users does not improve under FPCP compared with no cooperation. This is due to the following reasons. First, SBs do not serve primary user requests with high priority under FPCP. Second, due to relatively high value of $\lambda^{(s)}$ as compared to $\lambda^{(p)}$, SBs do not serve primary user requests very often under FPCP. Under the VPCP algorithm, average primary response delay is small as primary user requests are served with higher priority over secondary ones by secondary users. Further, this delay improves with higher $\lambda^{(s)}$ as higher $\lambda^{(s)}$ causes higher backlog of queued secondary user requests. In this case, the VPCP algorithm prioritizes stability of queues (by increasing secondary packet transmission opportunities) over minimizing the number of cached primary files. Hence, each SB caches more primary files with increasing $\lambda^{(s)}$ which will create more secondary packet transmission opportunities by increasing the fraction of primary user requests served by SBs[10].

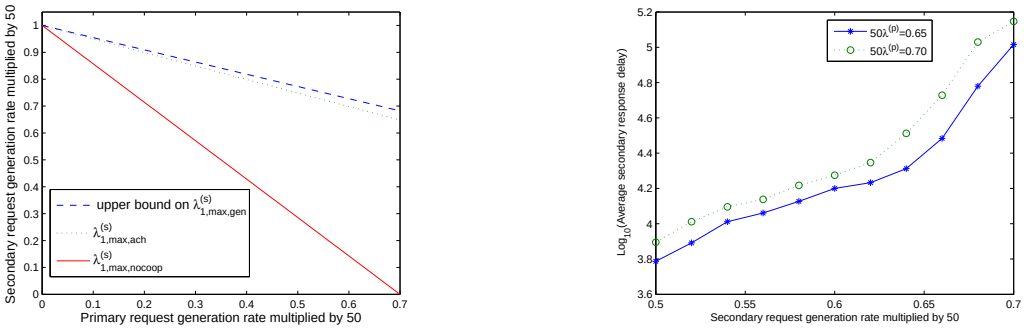
We observe from Fig. 5b that when $50\lambda^{(s)}$ is less than 0.75, average secondary response delay decreases under cooperation when FPCP or VPCP with penalty parameter $V = 0.01$ are used. This is because, under both these algorithms 199 out of every 200 cache positions at each SB are filled by primary files[11]. This reduces opportunities for SBs to satisfy requests for different secondary files in a period. For higher

[8]Typically in works on caching, Zipf distribution is used to model the popularity distribution of files.

[9]Note that average response delay is directly proportional to length of queues, by Little's law.

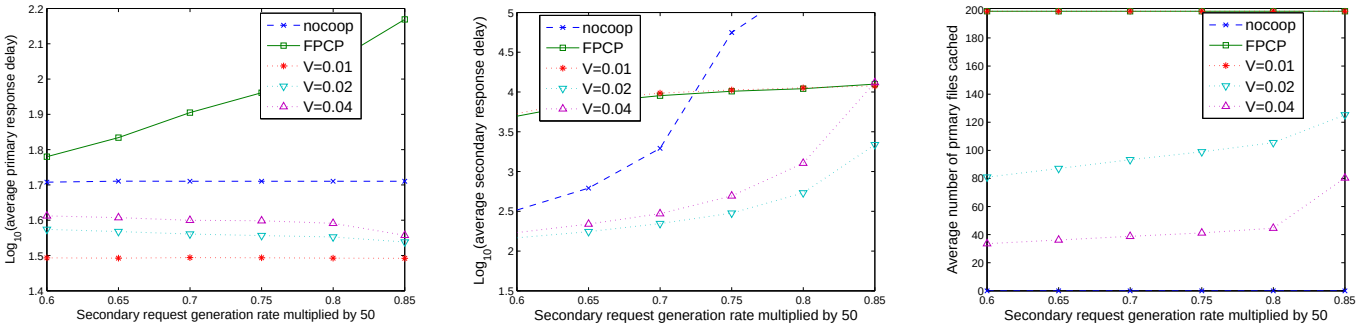
[10]Note that each primary packet transmission requires, on average, $\frac{1}{p}$ slots (where $0 < p < 1$) via the primary base station and 1 slot via any secondary base station. Hence for a fixed $\lambda^{(p)}$, each primary packet transmission via a secondary base station frees up, at least, $(\frac{1}{p} - 1)$ slots on average to transmit secondary packets.

[11]Recall, lower V means higher number of primary files to be cached; exactly 199 primary files are cached when V is 0.



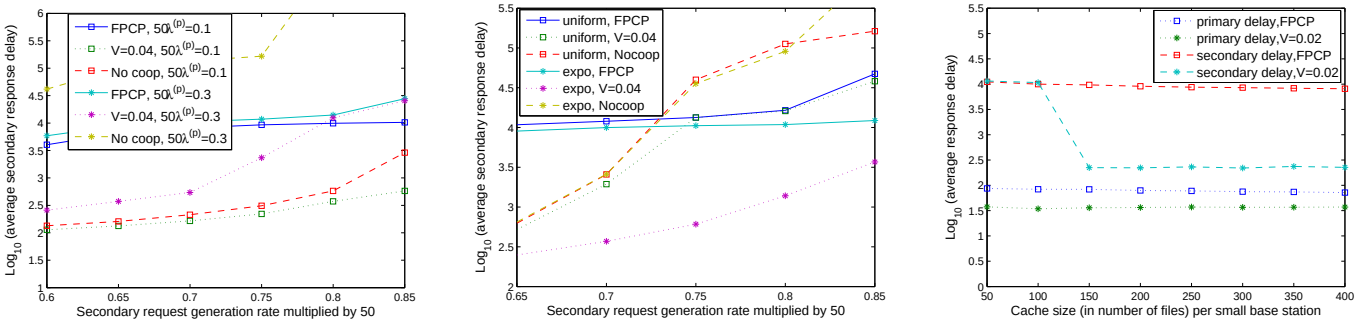
(a) Achievable capacity region (b) Average secondary response delay

Fig. 4. Comparison of $\lambda^{(s)}_{1,\max,\text{ach}}$ and upper bound on $\lambda^{(s)}_{1,\max,\text{gen}}$, and plot of base-10 logarithms of time-averaged secondary response delay under FPCP versus $50\lambda^{(s)}$ when $50\lambda^{(p)}$ is 0.65 and 0.70 respectively.



(a) Average primary response delay (b) Average secondary response delay (c) Average number of primary files cached

Fig. 5. Plot of base-10 logarithms of time-averaged primary and secondary response delay and the average number of primary files cached versus $50\lambda^{(s)}$ when $\lambda^{(p)}$ is $\frac{0.2}{50}$ under no cooperation, FPCP algorithm, and the VPCP algorithm for parameter V=0.01, 0.02 and 0.04.



(a) Variable request generation rates (b) Different distributions (c) Variable cache size

Fig. 6. Plot of base-10 logarithms of time-averaged response delay versus $50\lambda^{(s)}$ for different values of $\lambda^{(p)}$, for uniform and approximate gaussian distribution and versus different cache sizes (B) respectively.

$\lambda^{(s)}$, the system is unstable without cooperation and both algorithms perform better than the non-cooperative policy. Comparing the average secondary response delay under VPCP with $V = 0.01$ and that under FPCP shows that the average secondary response delay is not necessarily lower under VPCP for any value of V. However, for some choices of V (e.g., $V = 0.02$ in Fig. 4) the VPCP algorithm can lower average secondary response delay by striking a balance between the number of cached primary files versus system stability. We observe that, for this network, VPCP with $V = 0.02$ reduces average response delay of both primary and secondary users. For this value of V, the gain in secondary packet transmission opportunities (compared to FPCP or VPCP with $V = 0.01$) due to higher number of cached secondary files, offsets the reduction in secondary packet transmission opportunities due to higher rate of primary packet transmissions from PB.

In Fig. 6a, 6b and 6c we plot time-averaged response delays versus $\lambda^{(s)}$ for variable $\lambda^{(p)}$, for different file popularity distributions and for variable B respectively. In Fig. 6a and 6b we plot time-averaged secondary response delays versus $\lambda^{(s)}$ under FPCP, VPCP with $V = 0.04$ and no cooperation. In Fig. 6a we plot for two values of $\lambda^{(p)}$: $\frac{0.1}{50}$ and $\frac{0.3}{50}$ with other

parameters same as used for Fig. 5. We observe that the VPCP algorithm causes lower secondary response delay than the two other algorithms. In Fig. 6b we plot for two primary and secondary file popularity distributions: uniform and approximate Gaussian with $\lambda^{(p)}$ as $\frac{0.2}{50}$, $|F^{(p)}| = |F^{(s)}| = 401$ and other parameters same as for Fig. 5. Due to lack of space, details of this approximate Gaussian distribution is provided in [19]. We observe that under both distributions the VPCP algorithm with $V = 0.04$ performs better at higher $\lambda^{(s)}$. The delay is higher under uniform distribution as for this distribution, the rate of primary user request rates served via SBs is lower. In Fig. 6c we plot both primary and secondary average response delays under FPCP and VPCP with $V = 0.02$ as B is varied; $\lambda^{(p)}$ is $\frac{0.2}{50}$, $\lambda^{(s)}$ is $\frac{0.7}{50}$ and other parameters same as for Fig. 5. We observe that delay response in general improves with higher cache size. With VPCP the delay decreases until B is 150 at which point each small base station caches about 65 files on average. Increasing cache size B beyond 150 does not significantly decrease average delay because each small base station does not cache anymore primary file (and thereby serve more primary user requests).

VIII. Conclusion

In this work we studied cooperative caching in cognitive radio networks. Using Lyapunov drift techniques we proposed a caching and scheduling algorithm FPCP that increases the set of request generation rates that can be supported. We also proposed an alternative algorithm VPCP whereby SBs serve primary files requests with higher priority. In future we will extend this analysis to more general settings such as multiple SBs per primary user and multiple channels. Another interesting avenue of research is to find efficient values for the penalty parameter V in VPCP.

References

[1] F. S. P. T. Force, "Report of the spectrum efficiency working group," Nov. 2002.

[2] I. Maric, R. Yates, and G. Kramer, "Capacity of interference channels with partial transmitter cooperation," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3536–3548, 2007.

[3] I. Marić, A. Goldsmith, and G. Kramer, "On the capacity of interference channels with one cooperating transmitter," *European Transactions on Telecommunications*, vol. 19, no. 4, pp. 405–420, 2008.

[4] G. Kramer, M. Gastpar, and P. Gupta, "Cooperative strategies and capacity theorems for relay networks," *IEEE Trans. Inf. Theory*, vol. 51, no. 9, pp. 3037–3063, 2005.

[5] O. Simeone, Y. Bar-Ness, and U. Spagnolini, "Stable throughput of cognitive radios with and without relaying capability," *IEEE Trans. Commun.*, vol. 55, no. 12, pp. 2351–2360, 2007.

[6] I. Krikidis, N. Devroye, and J. Thompson, "Stability analysis for cognitive radio with multi-access primary transmission," *IEEE Trans. Wireless Commun.*, vol. 9, no. 1, pp. 72–77, 2010.

[7] A. Fanous and A. Ephremides, "Stable throughput in a cognitive wireless network," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 3, pp. 523–533, 2013.

[8] A. El-Sherif, A. Sadek, and K. Liu, "Opportunistic multiple access for cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 4, pp. 704–715, 2011.

[9] R. Urgaonkar and M. Neely, "Opportunistic cooperation in cognitive femtocell networks," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 3, pp. 607–616, 2012.

[10] J. Andrews, "Seven ways that hetnets are a cellular paradigm shift," in *IEEE Commun. Magazine*, vol. 51, no. 3, March 2013, pp. 136–144.

[11] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *Proc. of IEEE Infocom*, 2012, pp. 1107–1115.

[12] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems," *Synthesis Lectures on Communication Networks*, vol. 3, no. 1, pp. 1–211, 2010.

[13] N. Golrezaei, A. G. Dimakis, and A. F. Molisch, "Wireless device-to-device communications with distributed caching," in *IEEE International Symposium on Information Theory Proceedings (ISIT)*, 2012, pp. 2781–2785.

[14] N. Golrezaei, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," in *IEEE International Conference on Communications (ICC)*, 2012, pp. 7077–7081.

[15] J. Zhao, W. Gao, Y. Wang, and G. Cao, "Delay-constrained caching in cognitive radio networks," in *Proc. of IEEE Infocom*, 2014, pp. 2094–2102.

[16] J. Zhao and G. Cao, "Spectrum-aware data replication in intermittently connected cognitive radio networks," in *Proc. of IEEE Infocom*, 2014, pp. 2238–2246.

[17] M. Amble, P. Parag, S. Shakkottai, and L. Ying, "Content-aware caching and traffic management in content distribution networks," in *Proc. of IEEE Infocom*, 2011, pp. 2858–2866.

[18] N. Abedini and S. Shakkottai, "Content caching and scheduling in wireless broadcast networks with elastic and inelastic traffic," in *WiOpt*, May 2011, pp. 125–132.

[19] D. Das and A. A. Abouzeid, "Cooperative caching in dynamic shared spectrum networks," Tech. Rep., 2013. [Online]. Available: http://www.ecse.rpi.edu/homepages/abouzeid/preprints/2015cachingjrnl.pdf