Global Control Plane Architecture for Cognitive Radio Networks

Xiangpeng Jing and Dipankar Raychaudhuri
WINLAB, Rutgers University
671 Route 1 South
North Brunswick, NJ 08902
{xjing, ray}@winlab.rutgers.edu

***Abstract* — This paper presents an architecture for adaptive cognitive radio networks based on the concept of a "global control plane". The proposed control architecture uses a predetermined common coordination channel for spectrum etiquette, network establishment and adaptation to changing interference environments. The focus of this work is on design and evaluation of three key components of the control protocol - bootstrapping, discovery and naming/addressing. The bootstrapping protocol uses beacons to inform neighboring nodes about a node's PHY/MAC capabilities and current status. The network discovery protocol helps nodes to obtain a global view of reachability and end-to-end paths in the network by exchanging and propagating local link states. Further, nodes obtain their IP addresses and perform name to network address translations using a distributed naming/addressing scheme. An *ns2* simulation model of the cognitive radio network with global control has been developed and used to evaluate performance in terms of network setup time, control overhead and achievable data throughput.**

Keywords — Cognitive Radio Network, Bootstrapping, Discovery, Spectrum Etiquette Protocol, CSCC

I. Introduction

Recent "Moore's law" advances in programmable integrated circuits have created an opportunity to develop a new class of intelligent or "cognitive" radios [1] which can adapt to a wide variety of radio interference conditions and multiple protocol standards for collaboration between otherwise incompatible systems. Such a cognitive radio would be capable of very dynamic physical layer adaptation via scanning of available spectrum, selection from a wide range of operating frequencies (possibly non-contiguous), rapid adjustment of modulation waveforms and adaptive power control. In addition, a suitably designed cognitive radio with a software-defined physical layer would be capable of collaborating with neighboring radios to ameliorate interference using higher-layer protocols. Intelligent and reasoning methods [2-3] can also be introduced to the cognitive radio architecture for adaptation. Thus, suitably designed cognitive radios have the potential for creating a next-generation *adaptive wireless network* [4] in which a single universal radio device is capable of operating in a variety of spectrum allocation and interference conditions by selecting appropriate physical and network layer parameters in collaboration with other radios operating in the same region.

We propose and evaluate a specific architecture for adaptive wireless networks based on the concept of a "global control plane (GCP)" [5]. Because cognitive radio nodes have considerable flexibility in their choice of PHY and MAC parameters, their data planes must be configured dynamically on startup or entry into a new operating region. This motivates the use of a well-known common control channel (such as the CSCC described in our earlier work in [6]) for spectrum coordination and establishment of PHY, MAC and network layer functions. Extending the idea of the CSCC protocol, it is possible to design a more general GCP framework which includes protocols required for radio bootstrapping and network formation or adaptation. The control and data planes are generic to allow for implementation on a variety of radios with different available resources.

This paper introduces a GCP design for cognitive radios, including a bootstrapping protocol, a discovery protocol and distributed naming/address. The bootstrapping protocol enables nodes to be aware of surrounding nodes and direct communication links during startup or entry into a new physical region. The protocol helps new nodes to discover available networks and services by listening for bootstrapping beacons in the control plane. Wireless link quality and achievable link rates are estimated during the bootstrap and then local link states are aggregated throughout the network by a discovery protocol. In order to form adaptive networks, a node should be able to quickly setup and configure optimal links/paths to reach a destination, where the global awareness of the network is essential to achieve that. During discovery, optimal end-to-end paths are discovered with routing metrics obtained using estimated achievable end-to-end rate. The self-organizing and discovery involve multi-hop collaborations of cognitive radio nodes and negotiation of radio configuration parameters for data plane, which are distinct from the work in MANET such as zero-config working group [7], which only addresses one-hop networks with direct wireless links. To support applications communicating by permanent node names, a distributed naming/addressing scheme is also proposed for multi-hop cognitive radio networks, where unique IP address is assigned to each node and the name to network address translation service is provided in the self-organized network.

The proposed network architecture and control protocols are evaluated with *ns2* simulations of a typical adaptive wireless network scenario. The GCP is implemented using known 802.11b channel while data is carried over more general variable bandwidth OFDM channels with specifiable MAC layer protocols. An ad hoc network where cognitive radio nodes randomly boot up using the proposed bootstrapping and discovery protocols is evaluated in terms of network setup time, control overhead and estimated achievable end-to-end rate for discovered paths.

In the rest of the paper, Sec II introduces the network architecture; the bootstrapping and discovery protocols are presented in Sec III and IV; naming/addressing scheme is discussed in Sec V; then simulation results are demonstrated in Sec VI and we conclude with future work in Sec VII.

Research supported by NSF grant CNS-0626740.

II. NETWORK ARCHITECTURE

The proposed GCP-based network architecture is shown in Fig. 1 where the network is logically divided into separate control and data planes.

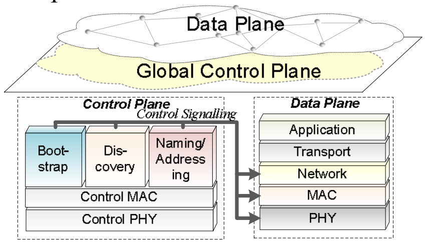


Figure 1: Global control architecture for cognitive radio networks

A. Control Plane

The control plane is made up of several key components: bootstrapping, discovery and naming/addressing. The radio bootstrapping function allows for detecting local links and configuring PHY/MAC parameters when cognitive radio nodes first boot up. After initialization, nodes execute a discovery protocol based on periodic reporting of local link states of neighboring nodes using a one-hop broadcast mechanism. The discovery protocol also interacts with routing protocols for data plane to provide end-to-end reachability and path information across multiple hops. The third key component is the support for naming and addressing by which network nodes map their permanent "names" to dynamically assigned network addresses which may change with network structure and mobility.

To implement the GCP, we extend the concept of CSCC protocol [5] to serve as the control plane for cognitive radio nodes by utilizing a low-cost control radio operating at the edge of the shared spectrum band. The control radio used is a generic low-rate 802.11-type radio fixed at one specific channel to implement the control plane functions and configure data plane which is generic and flexible in adapting to different spectrum and interference scenarios.

B. Data Plane

The data plane protocol stack on each node contains modules needed to support data communication between the wireless nodes and it exposes a set of controls for each module which interact with the control plane through APIs to monitor, configure and adapt the data plane modules. The separation of control and data planes gives the flexibility to optimize each function so that the data plane can use a pipe-like design to fully utilize radio resources and minimize protocol overheads, where data is forwarded in the data "pipe" established and configured by the control plane.

The control plane can set up multi-hop wireless forwarding paths while the data plane simply focuses on forwarding packets on pre-established paths. The control plane generally uses a low-rate radio PHY with wider coverage than the data signal, and can thus be used to efficiently distribute control information with fewer hops than would be required during data transfer. To achieve an efficient radio resource allocation for the data plane, the control plane can help to exploit the entire spectrum to detect spectrum opportunities for dynamic access across the bands, while avoiding the overhead of random channel access for data transmission. Radio parameters used by the data plane can be optimized for end-to-end performance by setting up frequency, power, bandwidth and rate parameters at each data forwarding hop to maximize spectrum space reuse and reduce interference to other nodes.

III. THE BOOTSTRAPPING PROTOCOL

In this section, the bootstrapping protocol for control plane operations is presented, which is aimed for nodes to obtain basic PHY/MAC parameters, local reachability and link state information when they first boot up or move to a network area.

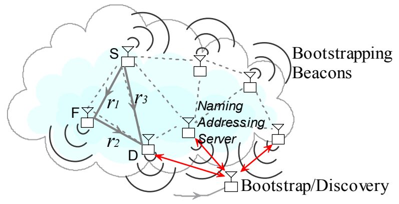
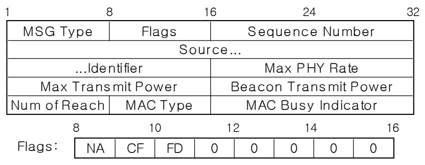


Figure 2: The bootstrapping protocol

In the network of Fig. 2, existing nodes periodically broadcast up-to-date bootstrapping beacons (*BSB*) on a specific control channel. When a new node boots up or moves nearby, it will first listen on the predefined control channel using default control plane radio configuration to collect bootstrapping beacons for a random period of time. A local link state table can thus be built up with the estimation of wireless link quality to neighbor nodes from their bootstrapping beacons. After the beacon collection process, the new node will start discovery process by exchanging all the link states with neighbor nodes to obtain a global view of the network. During the bootstrap, the new node can also detect naming/addressing services if available, which will be discussed in Sec V. After the bootstrap, new nodes begins to periodically broadcast self-states in their own beacons.



1	8	16	24 32
MSG Type	Flags	Sequence Number	
Source...			
...Identifier		Max PHY Rate	
Max Transmit Power		Beacon Transmit Power	
Num of Reach	MAC Type	MAC Busy Indicator	

Flags: 8		10		12		14		16
NA	CF	FD	0	0	0	0	0	

Figure 3: Bootstrapping beacon format

The bootstrapping beacon can be implemented as a low layer (PHY or MAC) broadcast within only one hop, which provides minimal required information of node states shown in Fig. 3. The beacon transmit power (quantized using 16 bits from 0 to 50dBm) is useful for beacon receivers to derive link quality between two nodes. MAC profile includes MAC type and MAC busy indicator, which indicates how busy the sender's data plane MAC is by periodical measurement of data MAC busy time per interval. It is normalized between 0 and 1 (indicated by a 16-bit integer) which is a good estimate of the sender's forwarding ability. A "flags" field usually has control or service information, e.g., "NA" bit indicates naming and addressing service. The collision of beacon messages is resolved by the control plane MAC, e.g. CSMA if 802.11 is used. Control overhead will be evaluated using simulations.

Based on the beacons a node collects, a local link state table is built up with link state vectors (*LSV*) for each direct wireless link. Each *LSV* is a tuple of destination node ID, link (or end-to-end path) weight, next hop ID and hop count, e.g., <$DestID_k$, w_{jk}, $NextHopID_k$, $HopCount_{jk}$> for node j describing the link from node j to k. The link weight is a performance metric assigned to direct links and end-to-end path weight is a metric of paths involving multi-hop relays. During the bootstrap, direct link weight can be obtained, which is an estimate of the maximum achievable PHY bit rate between two nodes by mapping estimated data signal to noise ratio (*SNR*) to physical transmission rate. Node i can estimate the path loss and thus *SNR* for data packets from a beacon of node j by:

$$SNR_{ij} = \frac{Pt_{\max i} \cdot \Pr_{ji}^{(B)}}{Pt_{ji}^{(B)} \cdot N_0} \qquad (1)$$

Pt_{maxi} is the maximum data transmit power of node i, $Pr_{ji}^{(B)}$ and $Pt_{ji}^{(B)}$ are respectively the received and transmit power of the beacon message, and N_0 is the noise power experienced at the data plane (estimated using 20MHz bandwidth). Here we assume the path loss between node i and j is the same as that of node j and i. Note if the data channel is close to the control channel, the path loss estimated by beacon messages is a good estimate for the data channel. Otherwise the path loss estimation is different (e.g., about 8dB more from 2GHz to 5GHz by Friis model), but the estimation at control channel can still be used as a quantity to evaluate the quality of a link. Note at the time of estimation there may not be a data transmission so the frequency to be used by data plane is not determined, and thus interference is not counted either in (1). By orthogonal channel allocation, the interference can be minimized or eliminated. The achievable physical bit-rate for data transmission can be estimated by the *SNR* to rate mapping function f_{map} known to the node's data plane. The maximum achievable link rate $R_{\max ij}$ can be obtained by:

$$R_{\max ij} = \min\{R_{\max i}, R_{\max j}\} = \min\{f_{map}(SNR_{ij}), R_{\max j}\} \qquad (2)$$

Taking MAC busy indicator into consideration, if the available bandwidth ($R_{\max ij}$) at a node is shared by transmissions for different data traffic, we define the link weight L_{ij} from node i to j as the "available" portion of the bandwidth as:

$$L_{ij} = R_{\max ij} \cdot \min\{\rho_{MACi}, \rho_{MACj}\} \qquad (3)$$

where ρ_{MAC} ($0 < \rho_{MAC} < 1$) is the MAC idle ratio (derived from the MAC busy indicator). The link weight L_{ij} is proportional to the maximum achievable rate from node i to j. The larger the weight, the higher data rate can be supported by the link.

IV. The Discovery Protocol

It is important for cognitive radio nodes to discover the network after bootstrap, because in order to quickly setup adaptive links/paths, a node has to have knowledge of a destination node and the optimal path to reach it.

Active discovery can be started by a new node or a node recovered from failure. A link state aggregation (*LSA*) message (Fig. 4) is used to poll neighbor nodes for aggregating local link states. Upon receiving a poll message ("PR" bit disabled), neighbor nodes then send all their *LSVs* in a *LSA* response ("PR" bit enabled). The *LSV* records path metrics to other nodes in the network. For example, *LSV* $<k, w_{jk}, k', C_{jk}>$ sent from node j means, node k can be reached by an end-to-end (E2E) path weight w_{jk} through next hop node k' with a hop count C_{jk}. Note that the requester can also piggyback its own link states in the poll message for suppression. To further reduce control overhead, only changes in link state vectors are propagated to the network in *LSA*.

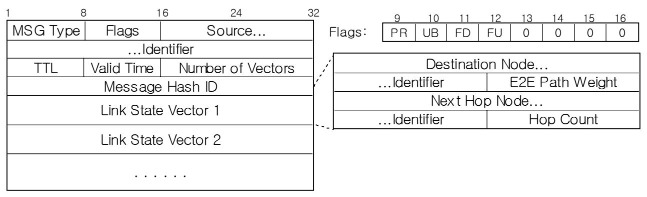


Figure 4: Link state aggregation (*LSA*) message format

When a *LSA* response is received, the link state table is updated and new entries are added by calculating end-to-end path weight if new paths/nodes are discovered. In the network of Fig. 2, when node S wants to transmit data to D, it can either directly reach D by rate r_3 or use node F as relay. The estimated per bit delay for both cases are:

$$E(D_1) = \frac{1}{r_3} \qquad E(D_2) = \frac{1}{r_1} + \frac{1}{r_2} \qquad (4)$$

Compared to transmission delay (especially for large data packets), processing/propagation delay and channel switching delay at node F can be ignored. Channel accessing delay is not counted here either as data forwarding can be completed in consecutive time slots or in orthogonal channels with minimum channel contention. Under the condition $E(D_1) > E(D_2)$, i.e., $r_3 < r_1 r_2 / (r_1 + r_2)$, node S would prefer relay rather than direct communication to D. Based on the analysis above, the end-to-end path weight is defined as the reverse of the summation of the reversing individual link weights along the path, i.e., when node i receives a link state vector $<k, w_{jk}, k', C_{jk}>$ from j, the new end-to-end path weight from node i to k is calculated as:

$$w_{ik} = \frac{1}{\sum_{m \to n \in \Re_{ik}} \frac{1}{L_{m \to n}}} = \frac{1}{\frac{1}{w_{jk}} + \frac{1}{L_{ij}}} \qquad (5)$$

where $\Re_{ik}$ is the link set of all hops (i.e. link $m \to n \in \Re_{ik}$) along the multi-hop path between node i and k. As the direct link weight L_{ij} is an estimate of each hop's link rate between node i and j, the end-to-end path weight w_{ik} will be a good estimate of the achievable end-to-end rate using intermediate traffic relays. From equation (5) we know that the higher each direct link weight, the higher the end-to-end path weight. The algorithm for updating link state table after calculating the new weight is shown in Fig. 5. If node k does not exist in the table, a new entry to destination k will be created and the link state vector $<k, w_{ik}, j, C_{jk}+1>$ is added. If there exists an entry to node k (e.g., $<k, w_{ik}, l, C_{ik}>$), the vector with the higher end-to-end path weight will be kept.

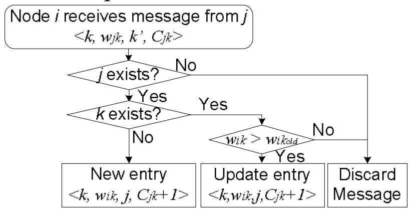

Node i receives message from j
$<k, w_{jk}, k', C_{jk}>$
j exists?
No
Yes
k exists?
Yes
No
$w_{ik} > w_{ikold}$
No
Yes
New entry
$<k, w_{ik}, j, C_{jk}+1>$
Update entry
$<k, w_{ik}, j, C_{jk}+1>$
Discard
Message

Figure 5: Flow chart for processing link state vector message

When there is no link failure, this algorithm is loop-free due to the definition of end-to-end path weight. If a destination node is discovered, the origin node will never update with a path going through itself, because from (5), any looping path going through the same link will cause the weight to decrease, while paths with only higher weight are updated. When there are link failures, the discovery protocol can also guarantee loop-free. When a wireless link is down, according entries in the link state table will not be deleted immediately; instead, the weight will be set to 0 during the next update interval and propagated to the network. When a zero-weight *LSV* is received, the relating path weights will be set to zero and the process is repeated. After the validity interval passes, obsolete *LSVs* will then be deleted. In this way, instant loops may exist but in the long run they will be eliminated after zero-weight *LSVs* are propagated.

The discovery process repeats periodically to keep the consistency and freshness of global information. The aggregation interval (5-10 seconds) is usually designed to be multiples of the BSB interval (2-5 seconds), in order to balance the trade-off between the speed of information propagation and control overhead. Note the aggregation is only a local one-hop broadcast which does not require global flooding. The unique message ID can also prevent re-processing of the same information.

V. NAMING AND ADDRESSING

The control functions for cognitive radio networks will support for naming and addressing of each node. A distributed scheme is proposed to achieve auto-configuration of each node with IP addresses and name-to-address translation.

A. *Distributed Naming/Addressing Server Election*

One of the key ideas of this scheme is to elect distributed naming/addressing servers, which are responsible for allocating unique IP addresses to those nodes covered by a server's control plane while also maintaining node name registration and translation to addresses. The proposed distributed naming/addressing (*NA*) service involving multiple *NA* servers divides the network into logical sub-networks for address allocation and name registration.

As shown in Fig. 6, the election of *NA* servers guarantees that each node in the network has access to at least one server through the control plane. If a new node fails to collect any beacons (with "NA" bit enabled) from *NA* servers during its bootstrap, it will begin to elect itself as an *NA* server by broadcasting *Address Pool Request (APR)* messages to obtain available IP address pools from existing *NA* servers in the network. Upon receiving any *NA* beacons during the election process, the node will cancel its election and register with the detected *NA* server. The *APR* message uses an expanding ring mechanism which starts as a 2-hop broadcast and increases the TTL hop count for subsequent retries. In a network with uniformly distributed nodes, there is a high probability that an *APR* message will reach *NA* servers within two hops. Only non-server nodes rebroadcast *APR* messages. Any *NA* server receiving an *APR* message will use a binary splitting mechanism [8] to tentatively allocate half of its own free IP address pool to the requester by unicasting an *Address Pool Grant (APG)* message. The requester will then accept the pool with the largest space by sending an *Address Pool Accepted (APA)* message. Non-acknowledged pools will be reclaimed by the owner after an *APG* timeout. If no *APG's* are received after several retries, the requester will choose a random IP segment (e.g. 10.31.*.*) to become an *NA* server by enabling "NA" bit in the beacons. Later during the periodical *NA* aggregation process, name and address information will exchanged through all distributed *NA* servers to detect and resolve collisions.

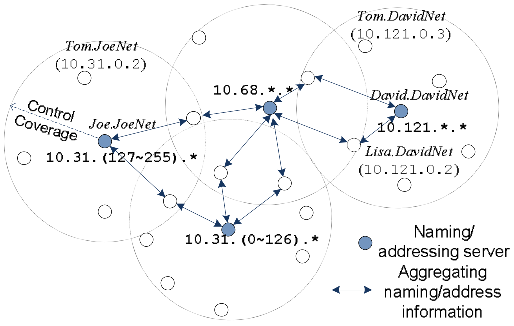


Figure 6: Naming/addressing scheme for cognitive radio networks

B. *Name/Address/ID Translation*

Upon detecting multiple *NA* servers during bootstrap, the new node will register its name to the one with the highest link weight by sending a *Name Registration Request (NRR)* message. The server will maintain the uniqueness of name registration in its control coverage by rejecting requests with conflicting names. The requester will then suffix the name with a random number and retry the registration. The accepted node will be allocated a unique network (e.g. IP) address from the available address pool, and meanwhile name-to-address and physical ID translation is maintained by the server. In this way, multiple logical subnets are formed where each *NA* server organizes names and addresses in one subnet. The resolution of name to address/ID is provided by the distributed *NA* servers. Node name/address/ID information is periodically aggregated and cached between *NA* servers by *Name Address Aggregation (NAA)* messages in order to provide global node name resolutions. The information caching at each server greatly improves the efficiency of name-to-address translation. *NA* servers' names (subnet names) are guaranteed to be unique during the aggregation process and conflict is resolved by granting the name to the server with a higher IP segment. In this way, each node can be uniquely identified by its joint node and subnet names.

VI. SIMULATIONS

The proposed control plane protocols for cognitive radio networks were evaluated using *ns2* simulations. The simulation results are presented in this section.

A. *Simulation Setup*

The control and data planes are implemented using a dual radio structure with a control radio and a data radio for each node. The control radio is an 802.11b radio operating at fixed channel 1 with 2Mbps rate covering about 250m. The control MAC uses the IEEE 802.11 standard without RTS/CTS. The data radio can be implemented quite generically (using varying frequency, bandwidth, modulation, power and rate parameters), but without loss of generality, we utilize 802.11a OFDM radio parameters at 5GHz for data plane with 8 channels of 20MHz each. PHY rates are 6, 9, 12, 18, 24, 36, 28 and 54Mbps and transmit power varies from 0 to 20dBm.

An adaptive network scenario of 1 km x 1 km with varying numbers of cognitive radio nodes is simulated where nodes are randomly placed in the network and boot up at random times. The bootstrapping and discovery protocols are evaluated in terms of network setup time, control overhead used and estimated achievable end-to-end rate. The maximum network setup time is the time from the start of the first node to the time all nodes in the network achieve global awareness by completing the discovery process.

B. *Simulation Results*

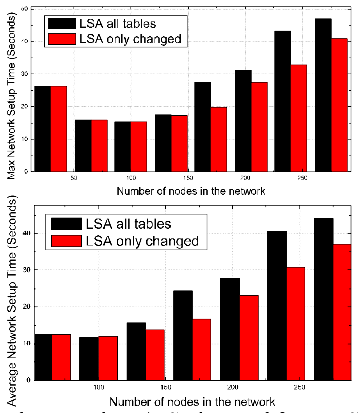


Figure 7: Network setup time (*BSB* interval 2sec, *LSA* interval 5sec)

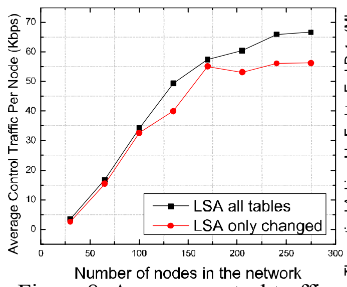


Figure 8: Average control traffic per node for network setup

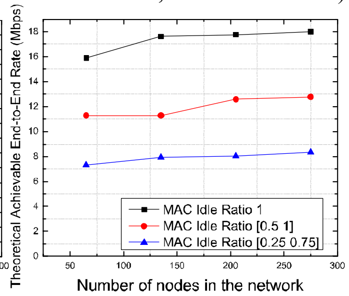


Figure 9: Estimated theoretical achievable end-to-end rate

The simulation results are compared for cases in which all link states are sent periodically ("LSA all tables"), or alternatively only when changes occur ("LSA changes only"). The maximum and average network setup time are shown in Fig. 7 where nodes random boot up from 0 to 4 seconds. With increasing number of nodes in the network, the network setup time first decreases and then increases, reaching its minimum at a node density of about 100nodes/km^2, because when the network is sparse, more *LSA* steps are needed to discover the whole network, while in a very dense network, the size and number of *LSVs* are large, and it takes about 3-8 *LSA* steps to discover the network. It is observed that when only changed link states are propagated, the network converges faster due to reduced control packets contending for the control channel.

The average control traffic per node during discovery process is shown in Fig. 8 with both bootstrapping beacons and *LSA* messages counted as control traffic. The average per node control traffic rate increases as the node number increases but the curve flattens out when the node number becomes large, converging to about 55-65kbps, which is well below the control channel capacity. When only changed link states are propagated, the control traffic rate is about 10kbps less than the case by sending all link states.

The estimated theoretical end-to-end rate is also calculated using equation (5) during discovery. Each node is randomly assigned a MAC idle ratio to simulate its busy condition. From Fig. 9, each node discovers paths to every other node in the network with average achievable end-to-end rate as high as 18Mbps for an 802.11a-type network involving multi-hop relays (usually 1-8 hops). The busier a node, the lower the end-to-end rate achieved due to reduced forwarding ability.

VII. Conclusion And Future Work

In this paper we have proposed and validated a novel network architecture for cognitive radio networks in which control and data plane operations are separated. Control plane protocols for bootstrapping, discovery and naming/addressing have been described. The bootstrapping protocol enables self-organizing of cognitive radio nodes to networks by building up local link state tables. Further, the discovery protocol helps nodes to achieve global awareness by periodically aggregating and propagating link states across the network. The naming/addressing service assigns network addresses to nodes with permanent "names", and maintains name-to-address translations. These control protocols are validated using *ns2* simulations. The preliminary results show the network setup time is only 3-8 steps of *LSA* process and a network with 100nodes/km^2 achieves an acceptably low network setup time. Average control traffic for completing the discovery also converges to about 65kbps per node. Theoretically achievable end-to-end rates are also evaluated for the example network.

In the future work, we plan to evaluate all the protocol modules described including the naming and addressing scheme. Mobility will also be addressed and a complete prototype implementation using USRP2/GNU radio platforms [9] will be realized.

References

[1] J. Mitola III, "Cognitive Radio: An Integrated Agent Architecture for Software Radio," *PhD thesis*, Royal Institute of Technology (KTH), Sweden, May 2000.

[2] R.W. Thomas, D.H. Friend, L.A. Dasilva and A.B. Mackenzie, "Cognitive Networks: Adaptation and Learning to Achieve End-to-end Performance Objectives," *IEEE Communications Magazine*, Dec. 2006.

[3] P. Mähönen, M. Petrova, J. Riihijarvi, and M. Wellens, "Cognitive Wireless Networks: Your Network Just Became a Teenager," *in Proc. of IEEE INFOCOM '06*, April 2006.

[4] D. Raychaudhuri, "Adaptive Wireless Networks Using Cognitive Radios as a Building Block," *MobiCom Keynote Speech*, Sept. 2004.

[5] D. Raychaudhuri, N. B. Mandayam, J. B. Evans, B. J. Ewy, S. Seshan, P. Steenkiste, "CogNet - An Architectural Foundation for Experimental Cognitive Radio Networks within the Future Internet," *in Proc. of MobiArch'06*, Dec. 2006.

[6] D. Raychaudhuri and X. Jing, "A Spectrum Etiquette Protocol for Efficient Coordination of Radio Devices in Unlicensed Bands," *in Proc. of PIMRC'03*, Sept. 2003.

[7] Zero Configuration Networking (Zeroconf), *http://www.zeroconf.org*

[8] M. R. Thoppian and R. Prakash, "A Distributed Protocol for Dynamic Address Assignment in Mobile Ad Hoc Networks," *IEEE Trans. On Mobile computing*, Vol. 5, No. 1, Jan. 2006.

[9] GNU Radio Project, *http://www.gnu.org/software/gnuradio*