This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2009 proceedings

A Neural Network based Cognitive Controller for Dynamic Channel Selection

Nicola Baldo*‡, Bheemarjuna Reddy Tamma†, B. S. Manoj†, Ramesh Rao†, and Michele Zorzi*†
*Department of Information Engineering – University of Padova, Italy
‡Centre Tecnològic de Telecomunicacions de Catalunya – Barcelona, Spain
†California Institute for Telecommunications and Information Technology – UC San Diego, USA
E-mail: {baldo, zorzi}@dei.unipd.it, {btamma, bsmanoj, rrao}@ucsd.edu

***Abstract*—In this paper, we present an application of the Cognitive Networking paradigm to the problem of dynamic channel selection in infrastructured wireless networks. We first discuss some of the key challenges associated with the cognitive control of wireless networks. Then we introduce our solution, in which a Neural Network-based cognitive engine learns how environmental measurements and the status of the network affect the performance experienced on different channels, and can therefore dynamically select the channel which is expected to yield the best performance for the mobile users. We carry out performance evaluation of the proposed system by experimental measurements on a testbed implementation; the obtained results show that the proposed cognitive engine is effective in achieving performance enhancements with respect to state-of-the-art channel selection strategies.**

I. Introduction

In recent years, the Cognitive Networking paradigm [1] has received significant attention by the research community, and is expected to be able to provide wireless devices and networks with enhanced adaptability and reconfigurability to cope with the challenges of radio communications. According to this paradigm, the wireless network becomes an active entity which observes and learns the historical behavior of the communication environment and makes important decisions on the network configuration, as in the particular case of channel selection that we consider in this paper.

We identify the following to be the main technical challenges for the design of a cognitive network controller:

Sensing the Environment: the cognitive controller needs to be able to obtain some sensorial information about the surrounding communication environment. This sensorial information is expected to be available in the form of measurements of different types, e.g., traffic information, signal and noise power measurements, as well as time and location coordinates.

Understanding the Network Status: the network controller needs to be able to identify the status of the network and the impact of different configuration settings on the network performance. It is of course possible to provide directly the controller with this knowledge, e.g., by hard coding the actions to be taken in response to different network conditions. However, a cognitive controller is expected to be able to learn these dependencies, thus relieving the engineering effort in providing the needed knowledge to the controller.

Prediction: environmental measurements provide information about the current and past status of the network; however, the network reconfiguration needs to be selected so that it is optimal with respect to the future status of the network. Determining the future status of the wireless network is not straightforward, since many external factors play an important role in its determination, such as variations in the traffic generated by the users as well as external interference. For this reason, some prediction strategies must be adopted.

Decision Making: once the controller has an understanding of the dependency between the network status and the performance with respect to different network settings, and is provided with suitable means of predicting the evolution of the environment, it still needs to perform its decision, i.e., to select the most desirable network configuration. Depending on the complexity of the configuration of which the network controller is in charge, i.e., depending on the characteristics of the solution space of the optimization problem faced by the controller, a proper search strategy needs to be chosen for decision making to be practical.

While systems adhering to the cognitive paradigm just described have been envisioned several times in the recent literature [1], [2], how to actually implement them has been discussed in only a limited number of publications. One of these is our previous work [3], in which we proposed Neural Networks as a suitable technique for implementing cognitive networking capabilities. The objective of this paper is to extend our previous research effort by presenting its application to a real system, in order to evaluate implementation issues and to be able to assess to what extent the performance gains that have been envisioned in theory and simulation can be achieved in practice.

The particular application we chose for this work is the problem of Dynamic Channel Selection in IEEE 802.11 networks. We made this choice not only because it is an interesting and challenging topic, due to the ubiquity of 802.11 networks and to the overcrowding of the 2.4 ISM bands in which 802.11b/g operates, but also because commercial 802.11 devices, in particular those supporting the MadWiFi Linux driver, offer a very flexible and cost-effective solution for the implementation of a Cognitive Network testbed.

The work at the University of California San Diego was partially supported by NSF under projects Rescue (NSF-0331690), CogNet (NSF-0650048), and Responsphere (NSF-0403433). The work at the University of Padova was partially supported by the European Commission under the ARAGORN project (INFSO-ICT-216856).

978-1-4244-3435-0/09/$25.00 ©2009 IEEE

II. Related Work

Traditionally, Channel Assignment in 802.11 infrastructured networks has always been performed in a static fashion, allocating the APs with techniques such as graph coloring [4]. Other proposals, such as [5], use a more complex approach, in which several pieces of information are supposed to be known a priori, such as the number of users in the network, the amount of traffic they generate, and the propagation characteristics of the network. While this is a reasonable strategy from a theoretical point of view, applying it to network deployments is difficult for several reasons. First, the propagation and interference models commonly used have some subtle differences from propagation in a real environment, and as a consequence the theoretical model for inter-cell interference and the resulting optimization model for channel allocation could fit poorly in a real deployment. Second, a typical assumption which is made is that there are few access points which have a very high load, while in many urban scenarios there are even tens of APs within range of each other, and the traffic per AP can be rather low; as a consequence, while many theoretical proposals focus prominently on load balancing issues among APs in the same network, in a real deployment other issues may be more relevant, such as interference from other wireless networks or microwave ovens. Third, the presence of the users and the amount of traffic they generate varies significantly not only with respect to time, but also with respect to space; for this reason, taking into account the instantaneous load variations at all access points to do a joint channel assignment, while straightforward in theoretical approaches where everything is assumed to be known, is not practical in real scenarios.

The Cognitive paradigm described in the introduction comes to the rescue with respect to this task, in that adaptation by observation and learning is expected to be more effective than theoretical approaches in tackling the subtleties of real scenarios. However, most of the work in this area is rather conceptual [1], [2], [6], [7], and only a few papers in the recent literature have discussed practical implementations, using techniques such as Genetic Algorithms [8], [9] and Fuzzy Logic [10]. However, a common drawback of the two techniques just mentioned is that they do not provide any means of learning from past experience, thus failing to exhibit one of the key properties of cognitive systems.

An interesting approach to the introduction of effective learning strategies in a cognitive network system is to use past experience, i.e., data from past environmental and performance measurements, to model the correlation between environmental conditions, possible network settings, and expected network performance. In this way, the cognitive network will be able to learn which parameter settings are most effective in which conditions. The problem has a strong similarity with the System Identification process which is well understood in Control Theory. A limited number of publications on this subject appeared in recent years. The authors of [11], dealing with the problem of network traffic prediction, evaluate the effectiveness of traditional prediction techniques such as Auto Regressive Integrated Moving Average (ARIMA) and Fractional Auto Regressive Integrated Moving Average (FARIMA), and compare them with Multilayer Feedforward Neural Networks (MFNNs), concluding that MFNNs are more practical due to lower complexity and the ability to model non-linear relationships. In [12] the authors compare the performance of linear regression models with that of MFNNs for the purpose of building models of the performance in mobile ad hoc networks as a function of external factors such as traffic load and configurable parameters such as the routing protocol being used; again, the authors conclude that MFNNs are the best modeling choice for the considered scenario. In a previous work [3], we proposed MFNNs as a general purpose learning and characterization tool to be used for the implementation of intelligent controllers in cognitive radios, with the purpose of adapting the configurable parameters of the radio to the environmental conditions in order to provide enhanced communications performance. In particular, we discussed the case of PHY rate adaptation in an 802.11 system, and reported simulation results showing that a MFNN-based approach was able to outperform state-of-the-art solutions. In this paper, we adopt the approach presented in [3], and design a Neural Network based controller to perform dynamic channel selection in infrastructured wireless networks. We stress that, while the generic approach to cognitive system design is similar, and in spite of the fact that the wireless technology considered is the same (IEEE 802.11), the problem being considered is very different (rate adaptation vs. channel selection) and, most importantly, a major contribution of this paper is to report performance evaluations based on a real testbed, thus being able to confirm that the MFNN-based approach to cognitive radio and networks is practical for real systems.

III. Cognitive Network Controller Design

A. General Concepts

The information which is available to our Cognitive Controller is classified into three different categories: *Environmental Measurements*, i.e., the set of all the measurements gathered by the network controller which can convey information on the external factors which might affect performance; *Parameter Settings*, i.e., the values of the configurable parameters of the system which have been used in the past; *Performance Metrics*, i.e., a measure of the performance of the network.

As we stated in the previous section, the key point of our design is that we require the cognitive controller to be able to learn how Performance Metrics depend on both Environmental Measurements and Parameter Settings. Following the approach that we introduced in [3], we train a MFNN-based predictor using Environmental Measurements and Parameter Settings as inputs, and Performance Measurements as known output values. The purpose of this component is to predict the performance that will be experienced in the future for different values of the configurable parameters in different environmental settings, so that the optimal configuration can be chosen. This is done by solving an optimization problem using the predicted performance as the cost function and the parameters as the variables.

B. Application to channel assignment in 802.11 networks

In order to apply the proposed approach to the particular problem of channel assignment in an 802.11 wireless network, we need to identify what particular pieces of information available in a real 802.11 system can be conveniently used for each of the categories introduced in the previous subsection.

For the *Parameter Setting* we obviously use the channel setting $(1, 2, \ldots, 11)$. For the training of the MFNN, this is the channel for which some particular measurement data has been obtained. At run time, this is the channel whose performance the Cognitive Controller wants to predict.

Since the objective of our cognitive controller will be to provide the channel assignment which maximizes the throughput of the users, for the *Performance Metrics* we use the *Application Layer Throughput* of the mobile users. For the training phase, we exploit measurements performed by a test client node; alternatively, other techniques such as throughput estimation based on traffic sensing or QoS reports fed back by real clients could be used. During the evaluation phase, the estimated application layer throughput is provided as the output of the predictor.

Finally, the most relevant environmental factor determining the performance of an 802.11 cell with respect to a particular channel is how much interference is present in the cell on that channel. Interference actually comes in two forms: interference in the form of radio power added to noise which negatively affects the success probability of frame receptions, as generated by far away 802.11 transmitters as well as by non-802.11 devices such as bluetooth and microwave ovens, and interference in the form of 802.11 transmissions contending for the medium, which trigger the transmission deferral and backoff freeze procedures as per the 802.11 standard, and result in increased medium access times. To characterize both these aspects, we use as *Environmental Measurements* the following metrics: the *Packet Rate*, defined as the number of valid IEEE 802.11 frames received per unit time; the *Data Rate*, defined as the value of the PHY data rate averaged over all successful IEEE 802.11 receptions; the *CRC Error Rate*, defined as the number of incoming IEEE 802.11 transmissions per unit time for which the link-layer CRC check failed; the *PHY Error Rate*, defined as the number of times acquisition was triggered at the PHY layer but reception was aborted because of a failure in the PLCP header check, and the *Packet Size*, defined as the average size of valid IEEE 802.11 frames.

Furthermore, some other *Environmental Measurements* are exploited, in particular information used to characterize the behavior of the system with respect to time: the *Day of Week*, ranging from 0 (Monday) to 6 (Sunday), and the *Hour of Day*, ranging from 0 to 23.

We note that every Cognitive Access Point in the network will collect the above mentioned measurements, so that the characterization will be specific to the position of each cell, thus effectively modeling the dependence of the performance of the communication system on location.

For the implementation of the MFNN-based predictor, if we assume that the environmental conditions vary slowly, it is sufficient to use only the last measurement as input of the neural network in order to have an accurate prediction; this was the approach we followed in [3]. However, this solution is not expected to be effective when the environmental conditions vary significantly between subsequent optimization periods. This aspect has actually been investigated in the past in the context of both generic and MFNN-based control systems [13]. To cope with this issue, we feed past measurements to a delay line with k taps prior to applying them to the MFNN, and feed simultaneously all the measurement values obtained at all the taps of the delay line to the MFNN for prediction. In this way, the NN predictor is expected to be able to identify and learn regular behaviors in the training data, thereby providing a more accurate prediction.

C. Implementation

The system described in the previous section was implemented on a testbed deployed in the Calit2 building at the University of California, San Diego. A collection of Cognitive APs forms the basis of our system. Each Cognitive AP consists of a Soekris Engineering net4521 system board, equipped with two Ubiquity 802.11a/b/g cardbus wireless interfaces based on the Atheros AR5213 chipset. Of the two wireless NICs, one acts as an AP and provides network access to wireless clients, while the other is configured in *monitor* mode to capture all 802.11 packets on the air, thereby acting as a wireless traffic sensor. The testbed is facing two main sources of interference: the production wireless service in the building, made up of Avaya 802.11b/g APs, and several experimental ad-hoc and mesh networks on the sixth floor of the building. For the experiment described in this paper, we have deployed 5 Cognitive APs on the 6th floor of the building among the production APs. Each Cognitive AP runs Voyage Linux OS, and uses the *MadWiFi* driver to drive the Atheros-based wireless interfaces. Each Cognitive AP is connected to the campus intranet via one of its Ethernet interfaces.

The Cognitive APs also feature a time-based sensing control module, which uses the STT sampling scheme described in [14], with a sampling period of 11 s and a sampling duration of 1 s. As discussed in [14], this allows accurate measurement across all 11 channels using a single wireless NIC. The synchronization and control module, on the other hand, performs synchronization of the Cognitive APs with the Cognitive Controller using *NTP* over the wired network.

Using the capture-to-file functionality of the open source *tcpdump* packet sniffer, the traffic sensor module creates capture files and remits them to the Cognitive Controller (Dell PowerEdge 1900 server with two Dual Core Intel Xeon processors operating at 2 GHz with 4 GB RAM and 7.2 TB of storage) via FTP. To further reduce the storage cost, *tcpdump* is configured to capture only the first 250 bytes of each sampled packet. This is a reasonable solution, since all protocol headers we are interested in are located at or near the start of the packet. At the Cognitive Controller, a modified version of *tcpdump* is employed to read the capture file to extract Prism monitoring header fields and header field values from the MAC through transport layers of the TCP/IP protocol stack. These values are stored in our Cognitive Network Database, implemented using MySQL server, from which they can be

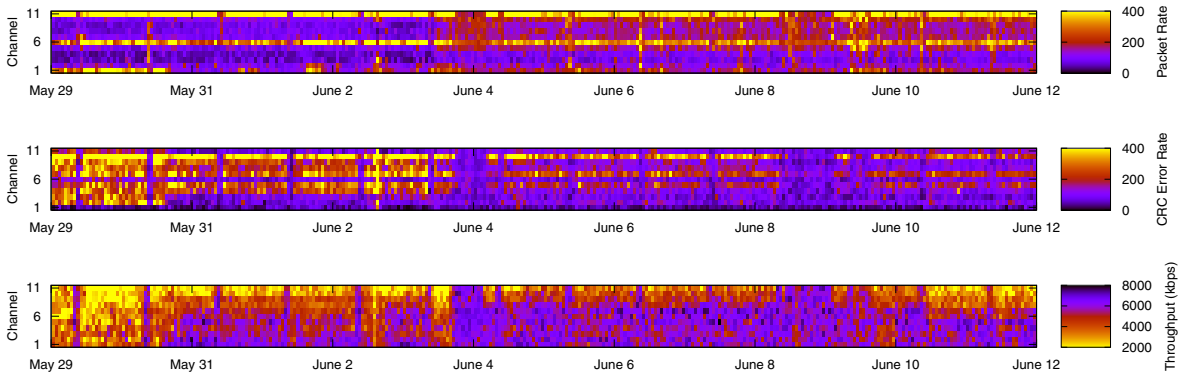

Fig. 1. Subset of the training data for node006 over a two week period from May 29 to June 12, 2008

queried to extract the training and test data used for this work, as well as for generic analysis purposes.

In addition to traffic samples, we gather other measurements which can be used to better characterize the conditions of the wireless medium. In particular, the MadWIFI driver has a tool (*athstats*) which provides additional statistics such as the number of CRC errors (i.e., receptions failed due to bad CRC in the MAC trailer) and PHY errors (receptions failed due to CRC failure on the PLCP header). This information is transferred periodically to the Cognitive Controller and is stored in a dedicated record.

Implementation of the time-based sensing module is done using a combination of shell scripts running *wireless-tools* and MadWiFi tools. These are used to periodically switch the wireless NIC's channel setting in order to gather traffic samples from all 802.11 b/g channels; channel switching is done in parallel to the traffic sensing activity. In order to report additional information about the packet currently being captured, the MadWiFi driver generates the Prism monitoring header, 144 bytes in length, and adds it to the packet. The Prism monitoring header contains, among other information, the Received Signal Strength Indicator (RSSI), channel and data rate of the packet.

We complement the set of Cognitive APs with a number of nodes built using the same hardware, with a single wireless NIC in managed mode, and configured as Programmable Clients (PCs). The Cognitive Controller interacts with the PCs, so that each PC is periodically associated with a different Cognitive AP, and runs a modified version of the *iperf* software to carry out active measurements by performing TCP data transfers both in the uplink and in the downlink direction. In this way application-layer performance metrics, such as throughput, delay, jitter, and packet loss ratio, are collected by the Cognitive Controller.

The MFNN-based Cognitive Controller described earlier in this section was implemented using the Fast Artificial Neural Network library [15].

IV. Performance Results

We used data gathered by the Cognitive Network testbed described in the previous section for the performance evaluation of the proposed Cognitive Controller. This data consists of the measurements described in Section III-B collected by the Cognitive APs of the testbed described in III-C and averaged over one-hour intervals. Data gathered from April 1 to July 15, 2008 was used as the training set, while data gathered from July 16 to September 15 was used as the testing set; the former set was used to train the Cognitive Controller, while the latter was used to test the accuracy of the prediction by calculating the mean square error between the predicted value and the known output value, and also to evaluate the performance achieved by both the MFNN-based channel selection scheme and other comparison schemes. A small subset of the data collected by *node006* from May 29 to June 12 is shown in Figure 1. We note that some correlation is evident between environmental measurements (the Packet Rate and the CRC Error Rate in the first two plots) and performance (the throughput in the third plot). This is exactly the type of correlation that we want the MFNN predictor to learn. The representation of other data is omitted due to space constraints.

Training procedures were performed using the backpropagation algorithm [13], [16] with a learning rate of 0.7 and a number of epochs of 1000. We tested different types of MFNN predictors varying the number of neurons H at the hidden layer and the number of delay taps k. We achieved the best prediction accuracy with $H = 13$ and $k = 2$, and therefore used these MFNN parameter values for the MFNN predictor to be used for channel selection. Minor variations of the H parameter did not result in significant variations of the prediction accuracy; however, too high and too low values performed poorly due to overfitting and insufficient modeling capabilities, respectively. As for the value of the k parameter, values higher than 2 did not provide significant improvements in the accuracy, and were therefore not chosen in order to minimize the complexity of the predictor. The Root Mean Square Error of the chosen predictor, calculated with respect to the test data, is reported in Figure 2.

We now evaluate the usage of the MFNN predictor for the purpose of performing cognitive channel selection. For every Cognitive AP in our testbed, the Cognitive Controller performed channel selection at the beginning of every hour

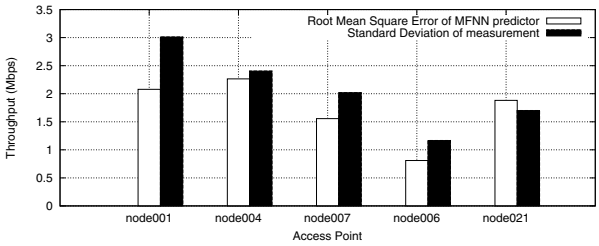

Fig. 2. Accuracy of the performance prediction