Architecture and Performance of an Island Genetic Algorithm-based Cognitive Network

Daniel H. Friend, Mustafa Y. ElNainay, Yongsheng Shi, and Allen B. MacKenzie
Bradley Department of Electrical and Computer Engineering
Virginia Tech
Blacksburg, Virginia 24061-0111
Email: {dhfriend,ymustafa,shiys,mackenab}@vt.edu

Abstract—This paper describes an architecture for a node in a cognitive network that employs distributed learning and reasoning. We present the architecture and describe a method of distributed reasoning using an island genetic algorithm. We then formulate a channel allocation problem that is unique to the use of cognitive radio networks for dynamic spectrum access. We provide simulation results for the island genetic algorithm as applied to our channel allocation problem.

I. INTRODUCTION

The rapid growth of wireless communication and networking has resulted in a number of conveniences for users, but it has simultaneously increased the complexity of networks, making configuration and operation a major challenge. For example, modern cellular networks require teams of skilled professionals to constantly tune network performance and to maintain network components. As network infrastructure and user devices achieve greater levels of reconfigurability, the complexity and cost of network configuration and operation create a pressing need for more intelligent networks. Such a network would perform autonomously (or at least require little human intervention) and be equipped to solve difficult problems and to learn in the face of uncertainty. A potential solution to these challenges is the emerging field of cognitive networks. Cognitive networks are formed when a collection of communication nodes organize to achieve network-level goals with the aid of some form of cognitive processing.

The core of the cognitive network is the cognitive cycle [1], which consists of six processes: observe, orient, plan, decide, act, and learn. A cognitive network achieves network-level cognition by integrating the cognitive cycle across layers in the protocol stack and throughout the nodes of the network. Such cognitive nodes require an architecture that supports observation of the state of the network, collective reasoning to achieve end-to-end network goals, learning from past actions, and reconfiguration of cognitive nodes based on collective decisions.

Our paper has three main contributions which constitute the subject matter of Sections III, IV, and V:

1) In Section III, we present a cognitive node architecture and describe its major elements. We also describe our choice of the island genetic algorithm (GA) for distributed reasoning. This choice is in part motivated by the flexibility of genetic algorithms in solving a wide variety of computationally challenging problems.

2) In Section IV, we describe channel allocation in a form that is unique to dynamic spectrum access (DSA) using cognitive radio networks.

3) In Section V, we present simulation results from applying the island genetic algorithm to the channel allocation problem described in Section IV.

II. RELATED WORK

Other cognitive node and network architectures, in varying levels of detail and with different emphases, have been proposed elsewhere. Sutton et al. [2] focus on the reconfigurable platform that underlies the cognitive node. Nolan et al. [3] detail further work in integrating the reconfigurable platform of [2] with a cognitive processing. Mahonen et al. [4] propose a node-centric architecture called a cognitive resource manager, which selects appropriate optimization methods from a toolbox and communicates with the outside world via a Unified Link-Layer API. Thomas et al. [5] provide a more abstract framework that illustrates the relationships between the functional units in a cognitive network. A broad review of research in cognitive radio, cognitive networks, and DSA is provided in [6].

Channel allocation has been widely studied, particularly for cellular networks [7]. In the cellular case, the goal of channel allocation is to maximize frequency reuse by minimizing the number of channels required to cover all cells in the network, subject to interference constraints. More recently, channel allocation has been applied to multi-hop wireless networks, such as 802.11 ad hoc networks [8]. However, neither of these cases accurately reflect the constraints experienced in dynamic spectrum access. In cellular and 802.11 channel allocation, it is assumed that a channel can be used in any cell, as long as interference constraints are met. This is not the case in DSA because primary user spectrum occupation is what drives channel availability. Also, there is no upper limit on the number of channels that may be used in cellular channel allocation, while 802.11 ad hoc networks have a fixed number of channels that are available at each node. Again, neither of these apply to DSA because different nodes may detect different sets of available channels. More recent work [9], [10] has formulated DSA channel allocation for cognitive networks.
radio networks, but these formulations essentially still use cellular models in which channels are assigned to base station nodes. For an ad hoc multi-hop cognitive network, a unique formulation of the DSA channel allocation problem is needed. The genetic algorithm has been successfully applied to channel allocation problems in the past [11]–[13]. Chakraborty and Chakraborty [11] use a centralized GA to compute a fixed channel allocation. Matsui et al. [12] apply a distributed GA to a fixed channel allocation problem. Fu et al. [13] combine a greedy algorithm with a centralized GA to perform dynamic channel allocation. All of these applications are for cellular networks.

III. COGNITIVE NODE ARCHITECTURE

The architecture presented herein has its roots in the work presented in [14] and [2] and is therefore derived from research in cognitive radio. The work in [2] provides a platform that allows reconfiguration of the whole protocol stack. We assume the presence of the type of platform described in [2] and focus on adding the ability to perform distributed learning and reasoning.

The cognitive node architecture is shown in Fig. 1. There are six major components of the architecture: the reconfigurable platform, stack manager, configuration and observation database (COD), exchange controller, distributed optimization process, and cognitive controller.

A. Reconfigurable Platform, Stack Manager, and COD

The reconfigurable platform and stack manager are inspired by the like-named components presented in [2]. Thus, each cognitive node has a flexible platform that allows the cognitive controller to choose from a variety of stack configurations in response to network conditions. The stack manager constructs the stack and reconfigures protocol layers.

The COD, which is a relational database, is the main repository for observations and configuration information. By keeping this information in a database, rather than internal to the cognitive controller, the information can be accessed by the exchange controller without interrupting the cognitive controller.

B. Exchange Controller

The exchange controller offloads communication and management overhead from the cognitive controller. Policies for exchanging observations between cognitive nodes are set by the cognitive controller and stored and enforced within the exchange controller. Thus, when another cognitive node requests data from the COD or knowledge base, the exchange controller responds according to established policy and without intervention from the cognitive controller. Requests from internal sensors for external data pass through the exchange controller so that multiple requests may be packaged in an efficient way. It is likely for observations to be on the order of a few bytes; therefore, the protocol overhead required to transport a single observation could be an order of magnitude larger than the observation itself if we assume a network protocol such as UDP/IP in conjunction with MAC frame headers. By packaging observations appropriately, the exchange controller alleviates some of the protocol overhead in obtaining observations from other nodes.

The exchange controller acts as an application in communicating with the exchange controllers of other cognitive nodes, which explains its connection to the reconfigurable platform in Fig. 1. Therefore, application layer protocol processing occurs within the exchange controller. This creates greater modularity in the architecture and simplifies the design of the cognitive controller. Part of the application layer overhead in the exchange controller is providing secure communications between cognitive nodes so that the integrity of critical data, such as new configurations, is maintained. The exchange controller may also offload some of the coordination tasks associated with distributed reasoning.

C. Cognitive Controller and Distributed Optimization Process

The cognitive controller and distributed optimization process are the heart of the cognitive node. The cognitive controller manages the flow of information to the distributed optimization process, deciding when it is necessary to re-optimize the current network configuration, as well as how the distributed optimization process should be configured. The cognitive controller also learns from past experience and stores the knowledge acquired by learning into the knowledge base.

Observations and experiences inherently contain uncertainty. This uncertainty may be related to sensing limitations (e.g. noise, nonlinear channels), but they may also be security-related, such as the trustworthiness of observations from an unknown source. As such, the cognitive controller should be based on a method that directly incorporates uncertainty into its processing. The two primary approaches to handling uncertainty are possibility-based, with the classic example being fuzzy logic, and probability-based. Our current work is in the direction of using a fuzzy logic controller as the
method behind the cognitive controller, with the island genetic algorithm for distributed optimization [15].

1) Island Genetic Algorithms: Genetic algorithms, in general, are population-based algorithms that incorporate random methods to search spaces that contain many local maxima and are too large for a complete search. Members of the population are commonly referred to as individuals. Each individual consists of multiple genes, one for each dimension of the search space. An initial population is generated randomly so that the search begins in multiple locations within the search space. Crossover and mutation, the basic evolutionary operations, are used to generate new individuals from existing ones. Individuals are compared using a fitness function, which is to be maximized or minimized.

An island genetic algorithm divides the population into sub-populations, or islands, that interact through the migration of individuals to other islands [16]. This migration is performed according to a migration policy, which defines where and when individuals move. A simple migration policy consists of a migration interval, which is the number of generations that occur between migrations, and a migration topology, which determines where individuals migrate.

We have chosen the island GA over other distributed GAs because of the flexibility in defining the migration policy. Other distributed GAs require that a rigid structure exist between computational nodes, and the amount of communication between nodes is significantly greater than what is typical with island GAs. Using an island GA allows cognitive nodes the flexibility of changing the migration topology as the topology of the network changes and also provides the opportunity to adjust the migration interval according to the bandwidth constraints of the network.

IV. Channel Allocation for Dynamic Spectrum Access

The cognitive network consists of a collection of $N$ nodes, $n_i \in \mathcal{N}$, and a set of $L$ directed communication links, $l_{i,j} \in \mathcal{L}_C$, where the subscript $i,j$ indicates a link for which $n_i$ is the transmitting node and $n_j$ is the receiving node. Together, these sets define the communication graph $\mathcal{G}_C = (\mathcal{N}, \mathcal{L}_C)$. Defining communication links as directional allows $l_{i,j}$ and $l_{j,i}$ to be assigned different channels so that communication between $n_i$ and $n_j$ may be full duplex. In addition to the communication graph, we define an interference graph, $\mathcal{G}_I = (\mathcal{N}, \mathcal{L}_I)$, by augmenting $\mathcal{G}_C$ with a set of directed links that indicate the presence of interference between nodes that do not share a communication link. We assume that nodes transmit omnidirectionally and with the same power on all channels so that for each link $l_{i,j} \in \mathcal{L}_I$, a transmission on any of $n_i$’s outgoing links will interfere with any of $n_j$’s incoming links if the links use the same channel. Therefore, any pair of nodes that shares a communication link can interfere with each other’s links, resulting in $\mathcal{L}_C \subseteq \mathcal{L}_I$.

Under this model, interference is a binary condition; either a pair of links interfere, or they do not. In this sense, our interference model is akin to a protocol model [17]; however, our model does allow interference to be determined based on signal-to-noise ratio, as in a physical model, or even based on random fading or shadowing. The difference between our model and the physical model of [17] is that we do not account for additivity of interference.

We assume that the cognitive nodes are capable of operating on multiple channels simultaneously, both transmit and receive, and that this multi-channel capability extends across the entire band of spectrum being sensed. This assumption is justifiable when the cognitive nodes are sensing on the order of tens of MHz. It is less easily justified when sensing several hundred MHz, although multi-band OFDM ultrawideband technology may make such a scenario feasible as well [18].

The sensed spectrum is broken into channels. Each node senses the spectrum and determines a set of channels available for local use, resulting in $N$ sets of channels, $\mathcal{C}_i$. Based on these sets, each link has a set of available channels, $\mathcal{H}_{i,j} = \mathcal{C}_i \cap \mathcal{C}_j$. We then define the length-$L$ channel assignment vector $\mathbf{h} \in \times \mathcal{H}_{i,j}$ (the Cartesian product of the available channel sets), which is the assignment of channel indices to communication links. In a deviation from standard vector indexing, we use $h_{i,j}$ to represent the element of $\mathbf{h}$ that is the channel assignment for link $l_{i,j}$.

Based on the interference graph, the DSA channel assignment problem is

$$\max_{\mathbf{h} \in \mathcal{H}_{i,j}} \left[ f(\mathbf{h}) = \sum_{l_{i,j} \in \mathcal{L}_C} \frac{w_{i,j}}{1 + |L^h_{i,j}|} \right], \tag{1}$$

where $w_{i,j}$ is the (fixed) capacity of channel $h_{i,j}$ and $|L^h_{i,j}|$ is the cardinality of the set of links that cannot be active at the same time as $l_{i,j}$ under channel assignment $\mathbf{h}$. $L^h_{i,j}$ is determined from the interference and communication graphs by

$$\mathcal{L}^h_{T,R} = \{l_{T,k} \in (\mathcal{L}_C \setminus \{l_{T,R}\}) : h_{T,R} = h_{T,k}\} \cup \{l_{k,T} \in \mathcal{L}_C : h_{T,R} = h_{k,T}\} \cup \{l_{k,R} \in (\mathcal{L}_C \setminus \{l_{T,R}\}) : h_{T,R} = h_{k,R}\} \cup \{l_{R,k} \in \mathcal{L}_C : h_{T,R} = h_{R,k}\} \cup \{l_{k,m} \in \mathcal{L}_C : \exists l_{k,R} \in \mathcal{L}_T, h_{T,R} = h_{k,m}\},$$

where the indices $T$ and $R$ are fixed and the indices $k$ and $m$ are variable.

The denominator of (1) reflects the fact that links which interfere cannot be active simultaneously and discounts the capacity of the link by dividing by the number of links that conflict. The numerator of (1) allows the channels to have non-uniform capacities, though channel capacities are not allowed to be link-dependent. Under ideal conditions, in which the optimal channel assignment, $\mathbf{h}^* = \arg \max_\mathbf{h} f(\mathbf{h})$, results in no link conflicts, $f(\mathbf{h}^*)$ is the maximum sum-capacity that can be achieved. This occurs because $|L^h_{i,j}| = 0 \forall l_{i,j}$, so that (1) reduces to $\max_\mathbf{h} \sum_{l_{i,j}} w_{i,j}$ over the set of interference-free channel assignments.

It is well known that channel assignment problems are generally difficult, with some particular formulations having
been proven to be NP-hard (e.g. [8]). Due to the facts that the sets \( \mathcal{H}_{i,j} \) are not known a priori and that the number of possible combinations for \( h \) grows exponentially in \( L \), we believe that the DSA channel allocation problem is not easily solved, though we make no claim as to membership in NP.

While we may develop a heuristic algorithm for (1), this algorithm will only apply to this specific problem. We are interested in solving (1) with a method that is suitable for a wide range of problems so that the cognitive nodes are able to tackle a variety of circumstances. Therefore, we have selected the island GA because of its demonstrated success with a large number of difficult problems.

V. DETAILS OF THE ISLAND GA AND NUMERICAL RESULTS

The first step in applying the island GA to the DSA channel allocation problem is to define the structure of individuals and the fitness function that is used to evaluate the fitness of individuals. In our case, this is simple, as \( h \) is our individual and (1) is our fitness function. The next step is to define the random crossover and mutation functions. Given two valid channel assignments, \( h_1 \) and \( h_2 \), we perform crossover by selecting a uniform random value from the integer set \( \{2, \ldots, L - 1\} \) and using this as the crossover point for standard 1-point crossover. The parents that are used in crossover are chosen by tournament selection. In our tournament selection, we choose three individuals at random from the parent population and then perform crossover on the two individuals with the highest fitness. Mutation is performed by selection of a uniform random value from the integer set \( \{1, \ldots, L\} \). Supposing that the selected value corresponds to link \( l_{i,j}, h_{i,j} \) is then replaced with a random selection from the set \( \mathcal{H}_{i,j} \setminus \{h_{i,j}\} \).

Populations are initialized randomly, with each \( h_{i,j} \) selected uniformly at random from the set \( \mathcal{H}_{i,j} \). The process for generating the next population from the current population is:

- Find the fitness for each of the \( M \) individuals in the population.
- Eliminate the worst (i.e. lowest fitness) \( \left\lceil M/2 \right\rceil \) individuals from the population. This new population is called the parent population.
- Generate \( M - \left\lceil M/2 \right\rceil \) new individuals from the parent population by tournament selection and crossover.
- Perform mutation on each individual, except the one with highest fitness, with probability \( \rho \).

If we rely solely on the above procedure without a cognitive controller, the island GA is prone to become trapped in local maxima. Therefore, we have implemented an adaptive island GA by keeping track of the number of iterations for which the maximum fitness has not changed and re-initializing the population if the maximum fitness has not changed for a certain number of iterations. The initial limit on the number of iterations before re-initializing the population, \( \tau \), is doubled after every re-initialization. The algorithm finally terminates after a fixed number of iterations.

In addition to specifying how each population is generated, we must determine a migration policy for the island GA. Our migration policy consists of a fixed number of iterations between the sharing of individuals (denoted by \( t \)), a migration topology that loops through all nodes, and the determination that nodes will share the individual with highest fitness. After the specified number of iterations has been reached, nodes share their final solution along the migration topology, with each node replacing the solution it receives with its own solution if it has higher fitness.

In our simulations, the network is generated by uniform random placement of a fixed number of nodes, \( N \), on a square. The area of the square is adjusted to maintain constant density for all values of \( N \) simulated, with the network being 1700 meters on a side for \( N = 100 \). Nodes have a communication range of 250 meters and an interference range of 500 meters. These settings result in an average number of links per node of about 3.5. The sensed spectrum contains 20 channels, with each channel’s capacity being 1 unit (the actual value doesn’t affect (1) if all channels have the same capacity). We randomly selected a set of channels, \( C_i \), for each node from the 20 possible, placing an upper limit of 8 channels and a lower limit of 2 channels on the size of each \( C_i \) to reflect primary user occupation of much of the spectrum.

For the island GA parameters, we set the island population size to 20, \( \tau = 10 \), a migration interval of 5, and a circular loop migration topology. The mutation rate, \( \rho \), was set to 0.4. Because the GA is a random search, the results from running the GA multiple times on the same network may differ. Therefore, results must be viewed statistically. Table I shows the results of running the island GA repeatedly on a fixed network, which ranged in size from 5 to 100 nodes. The optimal \( f(\mathbf{h}) \) values for \( N \geq 50 \) were determined by running the island GA for an extended period of time, whereas the optimal values for \( N < 50 \) were found by exhaustive search.

For smaller networks, with fewer than several million possible solutions (indicated by the Combinations row), the island GA always found the optimal solution. Also, the island GA converged to the optimum solution in about 30 iterations or less on average (indicated by the Avg. Iterations row). Larger networks experienced some small variations in the final solution, but even with a search space on the order of \( 10^{74} \), the island GA produced excellent solutions. Notice that we chose to allocate more simulation time to larger networks because the smaller networks had so little variation in the results. Interestingly, varying the migration interval, \( t \), from 5 to 1000 had little effect on the 100-node network performance.

<table>
<thead>
<tr>
<th>Nodes</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Avg. % Optimal</td>
<td>99.97</td>
</tr>
<tr>
<td>Std. Dev. (%)</td>
<td>0.05</td>
</tr>
<tr>
<td>Avg. Combinations</td>
<td>1.9E+10</td>
</tr>
<tr>
<td>Avg. Iterations</td>
<td>47.5</td>
</tr>
<tr>
<td>Std. Dev. Iterations</td>
<td>36.4</td>
</tr>
</tbody>
</table>
The performance results in Table I establish that our island GA is able to achieve excellent performance when run repeatedly on the same set of networks. This indicates the consistency of the random outcome under static network conditions. In order to make sure that the performance was not coincident with the particular generated networks, we also simulated the island GA for a series of different randomly generated networks with 25 nodes. This particular size was selected so that fitness evaluation of all possible channel assignments was feasible. The results are shown in Table II. As was the case for repeated simulations on a 25-node static network, the island GA was very near the optimal solution every time. The statistics for the number of iterations required to find the optimal solution were greater because the size of the search space for some of the networks (a function of the number of links and the size of the available channel sets) was one to two orders of magnitude larger than the 25-node network in Table I.

VI. CONCLUSIONS

In this paper, we have presented an architecture for a cognitive node and described the functionality of the major components of the architecture. We have proposed the island genetic algorithm for the distributed optimization component of the architecture. To evaluate the performance of the island GA, we presented a formulation of the channel allocation problem that is tailored to the specifics of DSA. The simulated results of the island GA on the DSA channel allocation problem reveal excellent performance.

The current binary interference model is limiting in that it does not allow for the realistic case of additive interference. Therefore, we are working on re-formulating (1) using SINR constraints. We are also working on combining fuzzy control with the island GA to dynamically adapt the island GA parameters so that the cognitive node can learn appropriate settings as both the problem and the environment change.

ACKNOWLEDGMENT

The authors wish to acknowledge the insightful suggestions of the reviewers. This material is based upon work supported by the National Science Foundation under Grant No. 0519959, by Award No. 2005-II-CX-K017 awarded by the National Institute of Justice, Office of Justice Programs, US Department of Justice, and by a Bradley Graduate Fellowship from the Bradley Department of Electrical and Computer Engineering at Virginia Tech.

REFERENCES


<table>
<thead>
<tr>
<th>Nodes</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>50</th>
<th>75</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal f (h)</td>
<td>9.0</td>
<td>10.85</td>
<td>15.12</td>
<td>18.09</td>
<td>21.63</td>
<td>50.44</td>
<td>67.09</td>
<td>97.55</td>
</tr>
<tr>
<td>Average f (h)</td>
<td>9.0</td>
<td>10.85</td>
<td>15.12</td>
<td>18.09</td>
<td>21.62</td>
<td>50.33</td>
<td>66.66</td>
<td>96.71</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.12</td>
<td>0.30</td>
<td>0.39</td>
</tr>
<tr>
<td>Combinations</td>
<td>3.2E+04</td>
<td>3.3E+05</td>
<td>1.3E+06</td>
<td>1.2E+07</td>
<td>1.5E+08</td>
<td>1.8E+09</td>
<td>1.8E+56</td>
<td>1.7E+74</td>
</tr>
<tr>
<td>Avg. Iterations</td>
<td>1</td>
<td>20.1</td>
<td>16.1</td>
<td>29.6</td>
<td>29.2</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Repetitions</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>193</td>
<td>43</td>
<td>30</td>
</tr>
</tbody>
</table>

TABLE I

PERFORMANCE RESULTS OF ISLAND GA ON DSA CHANNEL ALLOCATION