
Ismail AlQerm and Basem Shihada
CEMSE Division, King Abdullah University of Science and Technology (KAUST), Saudi Arabia,
{ismail.qerm, basem.shihada}@kaust.edu.sa

Abstract—Cognitive radio attracts researchers’ attention recently in radio resource management due to its ability to exploit environment awareness in configuring radio system parameters. Cognitive engine (CE) is the structure known for deciding system parameters’ adaptation using optimization and machine learning techniques. However, these techniques have strengths and weaknesses depending on the experienced network scenario that make one is more appropriate than others. In this paper, we propose a novel design for the cognitive system called supervised cognitive system (SCS), which aims to perform radio parameters adaptation with the most appropriate CE learning technique for the encountered network scenario. To realize SCS, it is required to evaluate the performance of different CEs in different network scenarios and according to certain performance objectives. In addition, the ability to select the most appropriate CE learning technique for adaptation in the current network scenario is also a priority in our design. Therefore, SCS investigates the relationship between learning and performance improvement and it employs online learning to classify scenarios and select the most appropriate CE learning technique for parameters adaptation. The testbed implementation and evaluation results show that the proposed SCS achieves more than 50% in performance gain compared to the best standalone CE.

Index Terms—Cognitive engine, Supervised cognitive system (SCS), system parameters adaptation, Online Learning

I. INTRODUCTION

Cognitive Radio (CR) is considered as a promising technology to improve not only spectrum utilization as in Dynamic Spectrum Access (DSA), but also system capacity, link performance and network efficiency through efficient radio system adaptation. Mitola’s vision [1] for cognitive radio aims to make the system parameters adaptation fully automated by implementing certain functions that constitute the cognition cycle, which are sensing, orientation, reasoning, and acting. Cognitive Engine (CE) is the intelligent agent that performs decisions driven by certain performance objectives to adapt system parameters, according to its observation and learning from the surrounding environment to achieve reliable communication and efficient resources utilization [2].

The design of CE using artificial intelligence is emphasized in the current research not only in DSA systems but also to build fully automated wireless systems. Case Based Reasoning (CBR) was exploited in the design of CE in IEEE 802.22 networks [3]. Heuristics techniques such as Genetic Algorithm (GA) are widely employed optimization techniques for decision-making in CR. Rondeau et al. applied GA to determine radio parameters of a software defined radio (SDR) to meet the user’s QoS requirements [4]. An adaptive GA based cognitive engine was proposed in [5] to tune radio parameters. Other techniques such as, Ant-Colony algorithm and particle swarm optimization were proposed for CE design by N. Zhao in [6] and [7] respectively. Zhang and Xie proposed an improved model of learning based CE with Artificial Neural Networks (ANN) methodology in [8]. In [9], Huang et al. designed a learning engine framework based on Support Vector Machine (SVM) to configure radio parameters using the estimation of bit error rate (BER) and signal-to-noise-ratio (SNR). The work in [10] proposed a primitive architecture for meta-cognition, which is used to rate the solution achieved by CBR and from this point it can decide if it is necessary to look for alternative adaptation algorithm such as GA. Y. Zhao et al. [11] looked into utility function selection for streaming video with a CE testbed. The work in [12] considers a design of cognitive engine using multiple objective optimization for parameters adaptation with enhanced GA as an optimization tool. The authors in [13] proposed a hybrid cognitive engine to perform system adaptation using CBR and decision-trees to boost the performance and expedite the convergence.

Let us shed the light to the fact that CE with certain learning technique can perform well at certain environment conditions such as high SNR, while another CE is more effective at low SNR. This creates the need to re-design the CE structure from a new perspective, which is the capability to select the most appropriate CE technique for the encountered network scenario. However, this requires establishing solid evaluation methodologies to be able to evaluate various CE designs and algorithms in different network conditions.

In this paper, we develop a supervised cognitive system called SCS that integrates typical CE with two additional components which are: evaluation component and control component. These two components distinguish SCS from typical CE designed for radio parameters adaptation in the literature. The evaluation component performs a comprehensive evaluation of the CE learning techniques in different environment scenarios. Our proposed technique for evaluation monitors the CE’s performance represented by certain metric at each decision step during its operation. Then, it estimates the relation between the learning of the CE and the achieved performance metric. The control component employs online learning to categorize the encountered scenarios and assigns the most appropriate learning technique to perform the system parameters adaptation accordingly. Online learning is chosen as it does not require detailed model or previous knowledge.
about the network and it can perform classification in the real-time. The performance of the proposed SCS cognitive system is evaluated using testbed and simulation and compared against other CEs proposed for system adaptation. The obtained results reflect the advantage of the addition of the evaluation and control components to the typical CE.

The rest of the paper is organized as follows, Section II describes the high level architecture of the SCS cognitive system with all its related components. Section III presents the SCS operation process including evaluations and the selection of the CE learning technique. The evaluation results including testbed implementation and simulation are presented in section IV and the paper concludes in section V.

II. SUPERVISED COGNITIVE SYSTEM (SCS)
ARCHITECTURE

In this section, we present our new approach to re-design the cognitive system, which has the capability to predict the performance of the involved CEs adaptation algorithms in various network scenarios and it is able to automatically select the most appropriate learning algorithm that suits the dynamic environment. This is achieved by introducing the evaluation and the control components to evaluate and control the learning process at a higher level. However, there are two critical challenges that must be tackled to realize the SCS system. The first challenge is that the evaluation of the CE algorithms is not trivial as it is necessary to find a way for monitoring the performance of individual CE techniques. Despite the fact that CEs should be able to sense their environment and autonomously adapt to the changing network conditions, different CE designs provide different levels of situation awareness and cognitive functionality. In order to tackle this challenge, we monitor the relation between learning evolution of the CE and the achieved performance. The second challenge is the ability to decide which CE algorithm is the most appropriate one for parameters configuration according to the network scenario, operating objectives, learning technique, and radio’s capabilities. Thus, we employ model-free online learning to develop the CE selection algorithm that fulfill these requirements.

The SCS system components including adaptation component, evaluation component, and control component. (see Fig. 1) are described in the following sub-sections.

A. Adaptation Component (CE)

The adaptation component incorporates the CE learning algorithm exploited to configure radio parameters. This component is generic and has the capability to accommodate any learning or optimization algorithm. In this implementation, we exploit $\epsilon$-greedy, ANN, and online learning as the learning algorithms of the adaptation component. A dedicated memory is exploited to store learning techniques related parameters. The $\epsilon$-greedy technique based on reinforcement learning [14] randomly explores different radio configuration parameters with probability $\epsilon$ and it exploits the radio configuration that achieve the maximum performance with probability $1 - \epsilon$.

This technique guarantees that all the configuration options are explored before convergence. $\epsilon$ is the factor that determines the rate of exploration. Although high exploration rate helps to reach near optimal configuration quickly but this occurs on a penalty over the outcome because of the exploration cost. The Multilayer Feedforward Neural Networks (MFNNs) [15] is exploited as the second learning technique and it provides a non-linear input-output relation with function approximation capabilities. It can accommodate large number of inputs and outputs and estimates performance metrics based on certain observation. Thus, it is a good candidate for system adaptation design. Q-learning [16] is the third CE technique, which performs decision-making without detailed modeling of the environment and improves its decision quality by interacting with it. Experience is the only factor required by Q-learning to operate in an optimal fashion and it is acquired from online interaction with the environment. The learner benefits from interacting with the environment to select an action that maximizes a cumulative reward. In addition, this learning technique achieves context awareness and intelligence in CR networks without training complexity [17].

Goodput, which is the rate of the successfully received packets at the receiver and Packet Error Rate (PER) are the performance objective considered in the configuration process. The environment conditions are represented by Signal to Noise Ratio (SNR) $\gamma$ and eigen-spread $\kappa$ of the communication channel also known as the Demmel condition number [18]. $\kappa = \lambda_{\text{max}}/\lambda_{\text{min}}$ are the maximum and minimum eigen-values of the $HH^H$ matrix. It is assumed that the channel is constant during the transmission time of the packet. The configuration parameters considered in this paper are transmission power, frequency, frame size, and modulation order. Each of these configuration parameters is represented by a set of discrete values and the CE algorithm selects the most appropriate value of the parameter based on its learning methodology. The details of the learning techniques are omitted due to the space and they are out of the paper scope. Reader can refer to [19] [20] [21] for more information about using these techniques.
in cognitive resource management.

B. Evaluation Component

The evaluation component’s function is to evaluate the performance of the employed CE techniques in the adaptation component in order to make the control component capable to classify the encountered environment scenario and selects the most appropriate CE technique to perform transmission parameters configuration. The output of the evaluation component is the feedback about the performance achieved as a result of using certain adaptation technique. The evaluation metric considered in the system design is goodput, which is calculated based on the Bit Error Rate (BER) and symbol error rate (SER). The coding gain $G_c$ for a convolutional code using soft decision decoder at BER=$10^{-3}$ is calculated as,

$$G_c = \frac{rd}{2}$$ (1)

where $r$ is the rate and $d$ is the free distance of the code. The hard code decision reduces the gain by factor of two. A hard decision decoder is used in the V-BLAST decoder because the received symbols need to be hard estimated as a part of the successive interference cancellation employed. Assuming Gray coding, the BER can be approximated from the SER as,

$$BER = \frac{1}{\log_2(M)} \times SER$$ (2)

where $M$ is the modulation index. Let’s assume that a packet with $N$ bits can tolerate a maximum $n_c$ error, then the goodput $GP$ can be calculated as follows,

$$GP = \sum_{i=0}^{n_c} \left( \frac{N}{N-i} \right) \times \left( 1 - BER \right)^{N-i}$$ (3)

Using (2) and (3), $GP_u$, denoted the upper bound of the goodput can be estimated using the additive white gaussian noise (AWGN) SER curves. Similarly, $GP_l$ (the lower bound goodput) can be estimated using the fading channel SER curves. Both upper and lower bounds of the goodput are utilized for further evaluations and the resulting evaluations are used to initialize the experience database.

C. Control Component

This component is the core of SCS as it coordinates radio parameters configuration and it identifies the encountered network scenario and selects the appropriate technique in the adaptation component for parameters configuration. Identification is achieved by grouping data into sub-populations according to their similarities, on the basis of a training set of data containing instances whose category membership is known. To realize the identification process, we need to define various environment states and their impact on performance objectives. This definition is achieved by extracting some distinct features of the environment, which are defined by SNR, eigen spread and performance objectives such as goodput maximization. The estimated SNR and eigen-spread must be discretized for indexing. The range for eigen-spread is from 0 to 25, with 25 equally spaced value. Similarly, the range for SNR is from 0 to 50 dB, with 51 equally spaced values.

Enhanced online learning is the technique employed to perform identification of network scenario and selection of CE technique to perform system configuration. Online learning is a model-free reinforcement learning (RL) technique. Specifically, it can be used to find an optimal action-selection policy for any given Markov Decision Process (MDP). It works by learning an action-value function that ultimately gives the expected utility of taking a given action in a given state and following the optimal policy thereafter. MDP includes a discrete set of environment states $S$ and a discrete set of actions $A$. At each time step $t$, the agent acquires network state information $s$ and selects certain action $a$ to perform. Consequently, the environment makes a transition to state $s'$ at time step $t+1$ with probability $T(s,a,s')$ and receives certain reward $R(s,a)$. This process is iterative and repeated infinitely to converge to an optimal decision-making policy $\pi$ that maximizes the total received reward. This policy is a mapping from environment states to probability distributions over actions. The value function is defined as a measurement of the quality of the action policy adopted and defined as follows,

$$V^\pi(s) = E[R(s,\pi(s)) + \beta \sum_{s' \in S} T(s,a,s') \max_{b \in A} Q^\pi(s',b)]$$ (4)

where $E$ is the expectation operator and $\beta \in [0,1)$ is the discount factor. As there is no prior knowledge about the state transition probability, the optimal policy Q-value for certain state/action pair is defined as,

$$Q^\pi(s,a) = E[R(s,a)] + \beta \sum_{s' \in S} T(s,a,s') \max_{b \in A} Q^\pi(s',b)$$ (5)

In this model free learning, the agent tries to find $Q^\pi(s,a)$ recursively using the following update rule,

$$Q^{t+1}(s,a) = (1-\alpha^t)Q^t(s,a) + \alpha[R(s,a) + \beta \max_{b \in A} Q^t(s',b)]$$ (6)

where $\alpha \in [0,1)$ is the learning rate. If each action is executed in each state infinitely and the learning rate decays properly, the $Q^t(s,a)$ will converge to $Q^\pi(s,a)$ with probability 1 as $t \rightarrow \infty$. The online learning parameters including state, action, transition and reward functions are defined in our system as follows,

- **State**: the state $s$ is defined as the network scenario denoted as $s = \{s, n\}$.
- **Action**: the action $a$ is the selection of the appropriate adaptation technique that maximizes the goodput. It is represented by a binary sequence to identify the adaptation technique as follows, $a = (00)$ for $\epsilon$-greedy, $a = (01)$ for ANN, and $a = (10)$ for the Q-learning.
- **Transition function**: the transition function $T(s,a,s')$ is the probability to switch from one state to the other and is defined as follows,

$$T(s,a,s') = Pr(s(t+1) = s'|s(t) = s, a(t) = a)$$ (7)
• **Reward:** the reward function $R(s, a)$ is defined as the achieved goodput and PER. It is evaluated to 1 if the goodput is above the minimum threshold and PER is below certain threshold. Otherwise, it is evaluated to 0. The procedure followed to select the most appropriate CE learning algorithm is explained in the next section.

### III. SCS Operation

As the SCS system aims to select the most appropriate adaptation technique to adapt radio parameters, this adds more complexity to the system as it requires evaluation the performance of different adaptation techniques in various scenarios. To overcome this issue, we design the system operation such that the evaluation process of the adaptation technique is activated on demand. This demand is triggered by monitoring the PER, which represent the environment feedback of adaptation. With low PER, the system retains the current adaptation technique. Otherwise, the system activates the evaluation and control components to search for the best adaptation algorithm for the current scenario. In this way, the adaptation time required is reduced and this was demonstrated through the adaptation time recorded in the simulation section. The SCS system operation flow goes as follows and illustrated in Fig. 2. First, the output (performance metric) of each CE learning algorithm is observed for certain time in different scenarios and all the observations are analyzed. Then, the evaluation process engages to check the appropriateness of each CE in the specified scenario and it is compared with other CEs. The observation of the CE is performed and output in Fig. 2. SCS operation flow

is monitored at each decision step in multiple runs in order to gain a good estimation about the performance of each CE technique. As result, we can obtain the relationship between learning and the output achieved. Fig. 3 presents a sample of the average goodput achieved for all CEs learning techniques employed as a function of time consumed for learning. The figure shows that Q-learning achieves minimal performance at the beginning as it relies on trial and error learning without any prior knowledge about the network environment. However, it converges to the maximum performance later compared to the other CE techniques as it has already obtained enough experience about the network environment. The $\epsilon$-greedy based CE does not assume any priori estimates about the performance metrics. Thus, the exploration is performed in a way that all the configuration parameters have the same probability to be selected. ANN relies on offline training using a designated training set consists of network scenarios and their corresponding parameters configurations. Therefore, $\epsilon$-greedy and ANN are not as aggressive as Q-learning and provide steady performance increase. After the evaluation stage, the SCS system has enough insights about the appropriateness of each CE in different network scenarios. Thus, it can proceed to the classification of the current network scenario and selection of the CE technique that match with the experienced conditions. Online learning is exploited to perform identification of network scenario and selection of CE technique as it has the capability to explore different network scenario and it is not limited to certain training data. Initially, we assume that there is no data about evaluation of CE algorithms. Therefore, the SCS selects CE techniques to perform parameters configuration randomly from the available CEs with probability $\tau$. After the SCS has gained experience about network scenarios and their corresponding evaluation with different CEs, it starts the exploitation process with probability $1 - \tau$. The exploitation process uses the obtained experience (old states) to handle the new experienced network scenarios (new state) based on the similarity calculated using the Euclidian distance between the current state and the old states $D_E = \sqrt{s(t) - s(x)}$. The SCS experience up to time step $t$ is given by the $\sigma$-algebra as follows,

$$F(t) = \sigma(s(t), a(x)) \cup \{R(s(x), a(x))\}_{x=1}^{t-1}$$ (8)

where information about the network state $s(x)$, associated action $a(x)$ and the reward $R(s(x), a(x))$ can be extracted from the evaluation component. The system checks the distance between the current state $s(t)$ and $s(x)$ in $F(t)$, and obtains a set $S'(s(t), F(t))$, which includes $F$ different most recent states from $F(t)$ that minimize $\sum_{f=1}^{F} D_E(s(t), s(x_f))$. The reward function $R(s(t), a)$ is set to be 1 if $a = arg\max_{a' \in A} Q^*(s(t), a')$ and 0 otherwise. The notation $A(s(t))$ is used to represent the set of actions that achieve $R(s(t), a) = 1$. Let $z$ be an integer that satisfies $1 \leq z \leq F$. Fig. 3. Performance of different CE learning techniques
SCS picks \( z \) records from the action set \( A'(S'(s(t), F(t))) \) with respect to \( S'(s(t), F(t)) \). The SCS performs selection action \( a(x^*) \) if \( (x^*) = \max_x \{ x | a(x) \in A(S'(s(t), F(t))) \cap A'(s(t)) \} \). Otherwise, SCS selects CEs algorithm randomly from \( A'(s(t)) = \{ a | a = \arg \max_{a \in A} R'(s(t), a) \} \), where

\[
R'(s(t), a) = R(s(t), a) \frac{\omega(s(t), a)}{z} \tag{9}
\]

This is calculated using \( z \) records drawn randomly from the \( F \) most recent performed actions where \( \omega(s(t), a) \) is the number of times this action was selected before for \( s(t) \). The procedure for CE algorithm selection using online learning is explained in Algorithm 1.

**Algorithm 1 Online learning for CE technique selection**

**Require:** current state \( s(t) \), old states \( s(x), F(t) \)

**Ensure:** selection of CE algorithm that output max GP

1. **Learning:** given network state \( s(t) \);
2. Exploration with probability \( \tau \);
3. Select action randomly;
4. Update \( A'(s(t)) = \{ a | R(s(t), a) = 1 \} \) for \( s(t) \);
5. Exploitation with probability \( 1 - \tau \);
6. Select \( z \) records \( A'(S'(s(t), F(t))) \) out of \( F \) actions associated with \( S'(s(t), F(t)) \);
7. Calculate \( R(s(t), a) \) according to (9) and populate \( A'(s(t)) \)
8. if \( (x^*) = \max_x \{ x | a(x) \in A(S'(s(t), F(t))) \cap A'(s(t)) \} \) then
9. Select action \( a(x^*) \);
10. else
11. Select action from \( A'(s(t)) \);
12. end if
13. Update \( R(s(t), a) \);
14. \( s = s^{t+1} \)
15. \( t = t + 1 \)

### IV. SCS Performance Evaluation

In this section, we evaluate the SCS performance through testbed experiments and simulation to demonstrate its capability of adaptation by selecting the most appropriate CE learning technique. The evaluation network structure consists of multiple ad hoc nodes each one has SCS system installed. The performance of SCS is compared with other systems in literature including [4], [22], and [23] that performed testbed implementation to demonstrate the capability of their proposed cognitive engines. The work in [4] proposed a genetic algorithm (GA) based cognitive engine for parameters adaptation. The authors in [22] proposed cognitive engine that exploits ANN to learn dynamically how to select the channel, which is expected to yield the best performance for the mobile users while the work in [23] proposed a hybrid engine design using case based reasoning (CBR) and GA. The hybrid engine relies on cases matching using CBR to expedite its convergence. In addition, we compare our simulation to the results obtained by the meta engine proposed in [10].

#### A. Testbed Evaluation Results

The considered testbed consists of one pair of nodes accessing the available network resources in typical office environment. The experimental setup comprises picture file transfer between the two nodes. The configuration parameters ranges are specified as follows: (-20 dBm to - 10 dBm) for transmission power, (20 to 600 bytes) for the frame size, and (BPSK, QPSK, 8-PSK, 16-QAM, 32-QAM, and 64-QAM) for modulation. Each node in the testbed is equipped with USRP-N210, and RFX2400 daughter-board mounted on top of it. The USRP-N210 is used as the radio front-end to perform physical layer information acquisition and parameters adaptation in combination with liquid-DSP [24] and GNU radio [25] platform. To facilitate the ease of implementation, we used an 802.11 feedback network which is usually supported through an internal wireless network interface controller (NIC). SCS performs data processing for the radio front end, as well as message parsing and takes the appropriate action. The implementation involves two radios, one for control exchange and the other for connection and data transfer between nodes. However, we use only one radio to perform both functions by using the concept of virtual WiFi as in [26]. We evaluate the performance of the SCS through three experiments as follows.

1) **Typical Office Environment Experiment:** The first experiment is conducted in an office environment whose layout is shown in Fig. 4. The sender and receiver are located in the same room. We evaluate the proposed SCS system performance by the following metrics: Goodput, PER, and spectral efficiency (SE). The average of the evaluation metrics during the experiment is recorded in Table I for SCS system and compared with the ANN [22] based cognitive engine, GA [4] based engine and the hybrid engine CBR+GA proposed in [23]. The results show that our proposed SCS system achieved the highest goodput, minimal PER and the highest SE compared to the other cognitive engines. Fig. 5 plots the achieved goodput as function of the offered load. The offered load is the percentage of utilization of the channel capacity. Fig. 5 demonstrates the ability of the SCS system to maximize the goodput compared to superior cognitive systems proposed for radio parameters adaptation.

![Office experiment layout](image)

#### 2) Propagation Impact Experiment: The second experiment evaluates the performance of the SCS under signal propagation impact. Fig. 6 shows the layout of the second experiment, where sender and receiver are located in far away isolated rooms. The goodput, PER, and SE for this experiment are
TABLE I
AVERAGE PERFORMANCE METRICS IN THE OFFICE EXPERIMENT

<table>
<thead>
<tr>
<th>Scheme Name</th>
<th>Goodput (Kbps)</th>
<th>PER (bits/s/Hz)</th>
<th>SE (bits/s/Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>420</td>
<td>0.0162</td>
<td>4</td>
</tr>
<tr>
<td>GA</td>
<td>480</td>
<td>0.0153</td>
<td>4.8</td>
</tr>
<tr>
<td>CBR+GA</td>
<td>600</td>
<td>0.0126</td>
<td>5.9</td>
</tr>
<tr>
<td>SCS</td>
<td>850</td>
<td>0.0088</td>
<td>7.1</td>
</tr>
</tbody>
</table>

Fig. 5. Average goodput with various network load

recorded in Table II. We notice that our SCS system achieved 30% improvement in the goodput, 65% reduction in the PER, and 38% raise in the SE compared to the best engine performance which is the hybrid engine. Fig. 7 compares the PER in two different locations: same office as in the first experiment and far away isolated rooms for all the cognitive adaptation engines. Fig. 7 confirms the ability of the SCS system to maintain the PER at very low level compared to the other cognitive systems. In addition, we notice that there is an increase in the PER and drop in the goodput and SE for all the systems in the second experiment compared to the first. The reason for that is the increase in the distance between the sender and the receiver and environment signal propagation impact.

Fig. 6. Propagation experiment layout

TABLE II
AVERAGE PERFORMANCE METRICS IN THE PROPAGATION EXPERIMENT

<table>
<thead>
<tr>
<th>Scheme Name</th>
<th>Goodput (Kbps)</th>
<th>PER (bits/s/Hz)</th>
<th>SE (bits/s/Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>380</td>
<td>0.073</td>
<td>2.8</td>
</tr>
<tr>
<td>GA</td>
<td>350</td>
<td>0.08</td>
<td>2.3</td>
</tr>
<tr>
<td>CBR+GA</td>
<td>490</td>
<td>0.043</td>
<td>3.9</td>
</tr>
<tr>
<td>SCS</td>
<td>700</td>
<td>0.0193</td>
<td>6.3</td>
</tr>
</tbody>
</table>

Fig. 7. Average PER at different locations

transmits signals by varying center frequency as in the layout presented in Fig. 8. The performance metrics measured in this experiment are recorded in Table III. The table shows that our SCS system achieved the highest goodput, minimum PER and maximal SE compared to the ANN, GA and CBR+GA cognitive engines. It is also evident that the performance of all systems has dropped because of the interference effect compared to the previous experiments in terms of all performance metrics. Fig. 9 plots the achieved SE against the SNR of the communication channel. The figure demonstrates the capability of the SCS system compared to other engines. It also shows that the SE remains stable after certain SNR value as the there is a limit for transmission power that can enhance the SE.

Fig. 8. Interference experiment layout

TABLE III
AVERAGE PERFORMANCE METRICS IN THE PROPAGATION EXPERIMENT

<table>
<thead>
<tr>
<th>Scheme Name</th>
<th>Goodput (Kbps)</th>
<th>PER (bits/s/Hz)</th>
<th>SE (bits/s/Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>340</td>
<td>0.083</td>
<td>2.2</td>
</tr>
<tr>
<td>GA</td>
<td>335</td>
<td>0.09</td>
<td>2.0</td>
</tr>
<tr>
<td>CBR+GA</td>
<td>460</td>
<td>0.055</td>
<td>3.2</td>
</tr>
<tr>
<td>SCS</td>
<td>670</td>
<td>0.0283</td>
<td>5.2</td>
</tr>
</tbody>
</table>

The results in Fig 5, 7, and 9 demonstrates the capability of the proposed SCS scheme in system adaptation. This highlights the advantage of making the cognitive engine supervised and able to select the most appropriate adaptation technique to perform system parameters adaptation. In addition, the figures points to the fact that each one of the engines exploited in the comparison has various performance at different network conditions. For example, GA achieves better spectral efficiency than ANN at low SNR while ANN outperforms GA at high
B. SCS Simulation Results

We simulated a multi-carrier system with 64 sub-carriers. Each sub-carrier was assigned a random attenuation value to simulate a dynamic channel. Hence, the SNR varied for each channel, inducing a need for the adaptation for each individual channel. The performance of the SCS system is compared to the performance achieved by the ANN based scheme [22], the hybrid engine (CBR+GA) [23] and the meta engine [10]. The SCS system performance is demonstrated using three evaluation metrics which are goodput, PER, and SE. Table IV presents the ranges of parameters for a system deployed under multiple users environment.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Iterations</td>
<td>1000</td>
</tr>
<tr>
<td>Exploration rate $\epsilon$</td>
<td>0.3</td>
</tr>
<tr>
<td>Number of users</td>
<td>Variable</td>
</tr>
<tr>
<td>Frame size ($N_b$)</td>
<td>24 bytes to 1024 bytes</td>
</tr>
<tr>
<td>Bandwidth ($B$)</td>
<td>312.5 to 812.5 KHz</td>
</tr>
<tr>
<td>Channel coding rate ($R_c$)</td>
<td>1/2, 1/3, 2/3, 3/4</td>
</tr>
<tr>
<td>Modulation type and order</td>
<td>M-PSK and M-QAM</td>
</tr>
<tr>
<td>Transmission power ($P_t$)</td>
<td>-30 dBm to -10 dBm</td>
</tr>
</tbody>
</table>

Fig. 10 presents the goodput achieved by each regular CE ($\epsilon$-greedy, ANN, Q-learning) and the one achieved by SCS. SCS selects one of the three learning techniques that is the most appropriate according to network scenario variation. The scenario variation is conducted by altering SNR to have the range of 0 to 50 dB and eigen spread varies between 0 and 12. The initial parameters configuration for power, modulation, coding rate, frame size and bandwidth follows a naïve style where power is configured at the minimum while modulation, frame size and bandwidth are at the maximum level. The configuration changes as the SCS system become aware of the environment and the involved adaptation techniques. The performance metric considered for this evaluation is the goodput achieved in a session with network scenario that varies every 100 epoch. Fig. 10 clearly shows how the SCS switches between learning techniques to keep goodput at the maximum level. It also shows that the aggressive learning algorithms like Q-learning perform better than ANN and $\epsilon$-greedy at high SNR levels (i.e., between 100 to 200 and 400 to 500 epoch). The rest of the simulations compare the performance achieved by SCS with the other cognitive systems. We evaluate the convergence speed for the competing systems for parameters adaptation. The normalized goodput is measured as a function of the time epoch in Fig. 11 (a). We observe that our proposed SCS achieved the highest goodput with the fastest convergence as shown in Fig. 11 (a) after 200 epoch. GA based engine is the slowest in convergence due to large computation required to perform adaptation. In addition, we evaluate the reliability and spectral efficiency of transmission by plotting the average PER and SE versus SNR in Fig. 11 (b) and Fig. 11 (c) respectively.

We notice that SCS outperforms all other engines. The reason is that the hybrid engine assumes that CBR is the main technique for adaptation and GA (slow in convergence) can be used to enrich the experience of CBR. The meta engine does not consider functionality of fully implemented system for resource management and does not account for convergence time. However, the SCS system performs extensive evaluations to rank the CE algorithms in different scenarios and employs enhanced online learning to determine the most appropriate adaptation technique.

Finally, we present the average time consumed to perform parameters adaptation in all the systems in Table V. The table shows that our SCS system achieved a comparable result which is little higher than the hybrid CBR+GA system thanks to the on-demand approach for activating the SCS system evaluation component and altering the adaptation technique. However, the time consumption pitfall is acceptable as the SCS manages to boost the cognitive engine performance according to the performance metrics investigated in the testbed implementation and simulation.

V. Conclusion

In this paper, we exploited machine learning to re-design the regular cognitive system for radio parameters adaptation. A new supervised system called SCS was proposed to select the most appropriate learning technique for adaptation that
achieves the ultimate performance. To realize this new cognitive system design, we employ three learning techniques including ε-greedy, ANN, and online learning to perform adaptation functionality. The proposed system introduces new components including the evaluation component which ranks the CE algorithms according to the achieved system performance and the control component that exploits online learning selection methodology to select the most appropriate adaptation technique. The SCS evaluation results including testbed implementation and simulation demonstrated its capability to achieve better performance in comparable time in different evaluation scenarios compared to other cognitive systems such as GA, ANN, and hybrid systems proposed in the literature.

REFERENCES


(a) System throughput as function of time step

(b) System PER with variable SNR

(c) System SE with different SNR

Fig. 11. Simulation Results that compares SCS with other cognitive systems