Enhancing Network Performance in Distributed Cognitive Radio Networks using Single-Agent and Multi-Agent Reinforcement Learning

A Technical Report submitted to the School of Engineering and Computer Science, Victoria University of Wellington, New Zealand, February 2010
Technical Report: ECSTR10-04

Kok-Lim Alvin Yau, Peter Komisarczuk, Paul D. Teal and Winston K. G. Seah
Communications and Networking Research Group
School of Engineering and Computer Science, Victoria University of Wellington
P.O. Box 600, Wellington 6140, New Zealand
{kok-lim.yau, peter.komisarczuk, paul.teal}@ecs.vuw.ac.nz

***Abstract*-Cognitive Radio (CR) is a next-generation wireless communication system that enables unlicensed users to exploit underutilized licensed spectrum to optimize the utilization of the overall radio spectrum. A Distributed Cognitive Radio Network (DCRN) is a distributed wireless network established by a number of unlicensed users in the absence of fixed network infrastructure. Context awareness and intelligence are the capabilities to enable each unlicensed user to observe and carry out its own action as part of the joint action on its operating environment for network-wide performance enhancement. These capabilities can be applied in various application schemes in CR networks such as Dynamic Channel Selection (DCS), congestion control, and scheduling. In this paper, we apply Reinforcement Learning (RL), including single-agent and multi-agent approaches, to achieve context awareness and intelligence. Firstly, we show that the RL achieves a joint action that provides better network-wide performance in respect to DCS in DCRNs. Secondly, we show that RL achieves high level of fairness. Thirdly, we show the effects of network density and various essential parameters in RL on the network-wide performance.**

I. Introduction

Cognitive Radio (CR) [1] enables unlicensed spectrum users or Secondary Users (SU)s to use, in an opportunistic manner, the unused licensed users' or Primary Users' (PU)s' spectrum (called white space) conditional on the interference to them being below an acceptable level. A Distributed Cognitive Radio Network (DCRN) is a distributed wireless network comprised of a number of SUs that interact with each other in a common operating environment in the absence of fixed network infrastructure such as a base station. Context awareness and intelligence are key characteristics of CR networks to achieve a joint action, which is the actions taken by all the SUs throughout the entire DCRN, that provides network-wide performance enhancement. Through context awareness, an SU is aware of its operating environment; and through intelligence, an SU utilizes the sensed and *high* quality white space in an efficient manner without following a strict and static pre-defined policy. We apply Reinforcement Learning (RL) to achieve context awareness and intelligence with respect to Dynamic Channel Selection (DCS) in this paper, though it can be applied in most application schemes that require context awareness and intelligence such as scheduling and congestion control.

There are two types of RL approaches, namely Single-Agent Reinforcement Learning (SARL) [2] and Multi-Agent Reinforcement Learning (MARL) [3]. Traditionally, SARL has been applied in operating environment with a single agent (or decision maker), such as the base station in a centralized network, so that it learns and takes action that maximizes its own network performance; while MARL has been applied in operating environment with multiple agents, such as all the SUs in a DCRN, so that they learn and take their own respective action, in a cooperative and distributed manner, as part of the joint action that maximizes the overall network performance. The SARL has been called RL in most literatures, however, in this paper, we refer to SARL as the single-agent approach and RL as the general approach comprised of SARL and MARL henceforth to avoid confusion.

In [4], a RL approach, or specifically, multi-armed bandit is investigated with respect to DCS. In [5], RL is applied to enable each SU to detect PU signal that may have deviated from its known signature. The investigation in [4] and [5] use machine learning performance metrics such as regret and fitness value, while this paper uses network

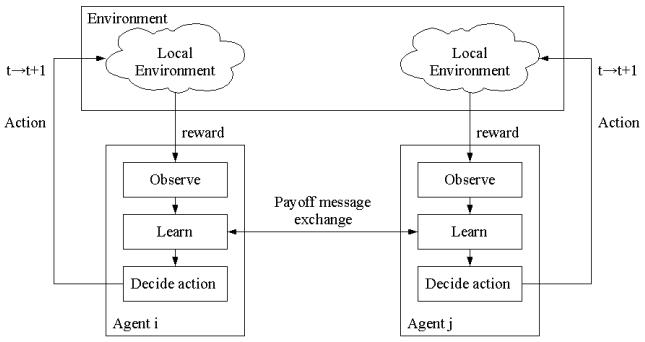

Figure 1. Agents (or SU communication pairs) and their environment.

performance metrics such as throughput and number of channel switchings. In [6], RL is applied in DCS in distributed CR networks in order to reduce call blocking and dropping probability, and the effects of RL parameters on network performance are investigated. In [7], RL is applied in DCS among the base stations in order to enable each of them to cover a minimum percentage of service area with the highest SINR so as to reduce call blocking and dropping probability. In [8], RL is applied to identify channels with the most available white spaces at the base station in centralized networks. In [9], RL is applied in spectrum assignment strategy in OFDMA networks in order to improve the PU's performance metrics including spectral efficiency, users' quality of service satisfaction, and the amount of licensed spectrum to be released to SUs. As a complement to [4]-[9] which investigate into SARL only, this paper investigates both SARL and MARL approaches.

The DCS provides strategy for the SUs to select a channel respectively from the available licensed channels for data packet transmission given that the objective is to increase network-wide throughput and to reduce number of channel switchings in order to decrease data packet transmission delay. We model each SU communication pair as a learning agent, as shown in Figure 1 because the transmitter and receiver share a single set of learned outcomes or knowledge. At a particular time instant, the agent observes only its own local operating environment due to its limited sensing capability. The learning engine provides knowledge on the operating environment comprised of multiple agents through observing the consequences of its prior action [3] in the form of local reward. The agent improves the global reward in the next time instant through carrying out a proper action. The global reward is a linear combination of all the local rewards at each agent. The difference between SARL and MARL is the additional feature in MARL, namely Payoff Message Exchange (PME). The PME mechanism is a payoff, which is computed using the local rewards, message exchange mechanism that helps each agent to communicate and compute its own action as part of the joint action. In other words, PME is a means of communication for the learning engine embedded in each agent. Note that SARL does not implement PME because it is a single-agent approach. As time progresses, the agents learn to carry out the proper action to maximize global reward. As an example, the learning engine is used to learn the channel conditions such as PU Utilization Level (PUL) and channel Packet Error Rate (PER). Higher levels of PUL indicates higher levels of PU activity, and hence smaller amount of white spaces. Higher levels of PER indicates higher levels of packet drop rate due to interference, channel selective fading, path loss, and other factors. SARL maximizes the local rewards; while MARL maximizes the global reward. Based on the application scheme, the reward indicates distinctive performance metrics such as throughput and successful data packet transmission rate. Thus, maximizing the local and global rewards provides network-wide performance enhancement.

We have successfully applied SARL in DCS for centralized CR networks in [9] where SARL is embedded in the BS. Although SARL is a single-agent approach, we have successfully applied it in DCS for DCRNs in [10] where SARL is embedded in each SU. In [11], we applied a PME approach called Payoff Propagation (PP) and it has been shown to converge to a joint action that provides better network-wide performance including DCRNs with cyclic topology; and fast convergence is possible. In this paper, we newly implement MARL, which is a combination of both SARL and PME, to further enhance network-wide performance. The MARL approach also addresses several drawbacks of game theory [11], which is a prominent tool to achieve context awareness and intelligence in CR networks. There are two major contributions in this paper. Firstly, we show that SARL and MARL achieve a joint action that provides better network-wide performance in DCS for DCRNs. Secondly, we show the effects of network density and various essential parameters in SARL and MARL on network-wide performance. The remainder of this paper is organized as follows. Section II discusses the characteristics of DCRNs. Sections III presents SARL, MARL and their RL model for DCS. Section IV presents Medium Access Control (MAC) protocols with DCS implementation for DCRN. Section V presents simulation experiments, results and discussions. Section VI presents our conclusions.

II. CHARACTERISTICS OF DCRNS

We refer to a single node as an SU; and an SU communication pair as an agent henceforth. The single-hop DCRN, as illustrated in Figure 2, is comprised of V SUs. T_i is the transmitter and R_i is the receiver of an agent i. There are $U = V/2$ agents. Each agent maintains a single set of knowledge because the transmitter and receiver must choose a common channel for data transmission. We consider a common assumption of a single collision domain; hence, all the agents can hear each other. There are K PUs, $PU = [PU_1, \ldots, PU_K]$ and each of them uses one of the K distinctive channels of frequency $F = [F_1, \ldots, F_K]$. We consider $K \leqslant U$, so the agents are competing to use the channels. Each channel is characterized by various levels of PUL, $L = [L_1, \ldots, L_K]$. Each agent i experiences different levels of PER, $P_i = [P_{i,1}, \ldots, P_{i,K}]$ for each channel; thus, we consider heterogeneous channels. However, most schemes including our previous work [10] assume the similar levels of PERs are observed for various channels for

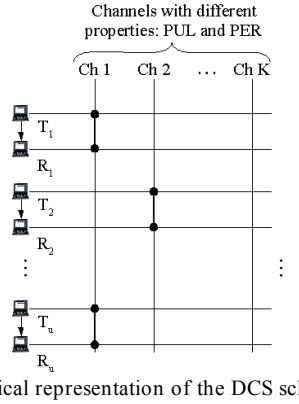

Figure 2. Graphical representation of the DCS scheme. Bold link indicates data transmission over a chosen channel.

all agents such that the PER is $P=[P_1,\ldots,P_K]$. Each agent infers the PUL, PER and contention level in each channel, and selects in a distributed manner a channel for data transmission. A channel with low PUL does not imply a good channel if it has a high level of contention or PER at the agent. Figure 2 illustrates the concept of the DCS. Suppose, agent 1 or T_1-R_1 chooses channel 1; while agent 2 or T_2-R_2 chooses channel 2. Channel K is not chosen because, say, it has high PUL and PER at all agents. Agent u chooses channel 1 because the channel has lower PUL and PER compared to channel 2. This channel selection provides better network-wide performance.

III. SARL-BASED AND MARL-BASED DCS

In this section, we present SARL and MARL based on our application context. Next, we present RL-based DCS that details the RL model for SACC and MACC.

A. SARL

Q-learning [2] is an on-line algorithm that determines an optimal policy without detailed modeling of the operating environment. Denote decision epochs by $t\in T=\{1,2,\ldots\}$, a constant epoch duration by t_D, actions by $a\in A$, and delayed rewards by $r_{t+1}(a_t)$. Each agent i maintains a Q-table with $|A|$ entries to keep track of learnt action value or Q-value, $Q_t(a)$ within an interval of $[0,Q_{max}]$ for all its possible actions. The Q-value estimates the level of local reward for an action a; hence changes in the Q-value will lead to changes in an agent's action. At each decision epoch t, agent i chooses an action a_t and receives a local reward $r_{t+1}(a_t)$ at time $t+1$. The agent i updates the Q-value of action a_t at time $t+1$ as follows:

$$Q^i_{t+1}(a^i_t)\leftarrow(1-\alpha)Q^i_t(a^i_t)+\alpha r^i_{t+1}(a^i_t) \quad (1)$$

where $0\leqslant\alpha\leqslant 1$ is learning rate. Higher value of α indicates greater reliance on the recent local reward compared to the past knowledge. As this procedure evolves through time, agent i receives a sequence of rewards. An optimal policy is being searched for that maximizes value function V^π as shown below:

$$V^\pi=\max_{a\in A}(Q^i_t(a)) \quad (2)$$

As an example, in DCS, the Q-value represents the throughput and it is dependent on the local PUL, PER and the joint action that represents the channel selection made by all the agents. The joint action affects the Q-value due to the dependency of actions among the agents; for example, two neighbour agents that choose a particular channel may increase their contention level, and hence reduces their respective Q-values for the action.

B. MARL

The MARL is a combination of both SARL and PP mechanism. The SARL, which is the learning engine embedded in each agent, provides the local reward, while the PP mechanism provides a means of communication for the learning engine.

Each agent i maintains a Q-table with $|A|$ entries; and a μ-table with size $|\Gamma(i)|\times|A|$ to keep track of the payoff messages. The $\Gamma(i)$ represents all the neighbours of agent i. The agent computes its respective action $a_t^{i,*}\in A$ as part of the optimal joint action $\boldsymbol{a}^*_t$ using its own local Q-value from its Q-table, $Q^i_t(a^i_t)$ and its neighbours' local Q-value from its μ-table, $Q_t^{j\in\Gamma(i)}(a^j_t)$ to achieve the optimal global Q-value $\boldsymbol{Q}_t(a^*_t)$.

Each agent i constantly sends payoff message $\mu_{i*}(a^*_t)$, which is the local Q-value of its own current action, to its neighbour agents $j\in\Gamma(i)$ as follows:

$$\mu_{i*}(a^*_t)=[Q^i_t(a^i_t)] \quad (3)$$

The payoff messages are exchanged among the agents until a fixed optimal point is reached. Before convergence, the messages are an estimation of the fixed optimal point as all incoming messages are yet to converge. As an example, when agent j receives the $\mu_{i*}(a^*_t)$ while it is taking action a^j_t, $\mu_{i*}(a^*_t)=\mu_{ij}(a^j_t)$ indicates the local rewards of agent i while agent j is taking action a^j_t.

Each agent selects its own optimal action to maximize the local payoff as follows:

$$g^i_t(a^i_t)=\max_{a\in A}[Q^i_t(a)+\sum_{j\in\Gamma(i)}\mu_{ji}(a)] \quad (4)$$

Each agent i determines its optimal action as follows:

$$a_t^{i,*}=\arg\max_{a\in A} g^i_t(a) \quad (5)$$

The global payoff $g_t(\boldsymbol{a}_t)$ at time t is a linear combination of all the local payoffs generated by each SU:

$$g_t(\boldsymbol{a}_t)=\sum_{i=1}^{u}[Q^i_t(a^i_t)+\sum_{j\in\Gamma(i)}\mu_{ji}(a^i_t)] \quad (6)$$

The MARL is executed until the agent converges to an optimal local action where the changes of its local Q-values and local payoff values between iterations are insignificant. However, due to dynamic operating environment, learning must be carried out constantly as the optimal joint action changes with time. In [11], the PP mechanism is shown to converge to a joint action that provides better network-wide performance in a distributed manner including a DCRN with cyclic topology, and fast convergence is possible. Additionally, if entries in the Q-table and μ-table at each agent are stable and fixed, PP will converge to a joint action that provides better network-wide performance.

C. Other Mechanisms in SARL and MARL

The update of the Q-value in (1) does not cater for the actions that are never chosen. Exploitation chooses the best known action, or the greedy action, at all times for performance enhancement. Exploration chooses the other non-optimal actions once in a while to improve the estimates of all the Q-values in order to discover better actions. In the ε-greedy approach [2], an agent explores with a small probability ε, and exploits with $1-\varepsilon$.

The global Q-value at time t is a linear combination of all the local Q-values at each agent as follows:

$$Q_t(\boldsymbol{a}_t)=\sum_{i=1}^{u}Q^i_t(a^i_t) \quad (7)$$

Note that equation (7) is not a utility function, which is not defined in the MARL [3]. Equation (7) shows that a global reward can be optimized through maximizing the local rewards, thus simplifying the complexity of maximizing the global reward.

Note the difference between the global Q-value, $\sum_i Q_t^i$ in (7) and the global payoff, $\sum_i Q_t^i + \mu_{ji}$ in (6). The global Q-value is the total local rewards received by all the agents in the network; while the global payoff is the total local rewards received by all the agents in the network and total payoff value exchanged among the agents. Global Q-value is a performance metrics for MARL and SARL; while global payoff has been shown to converge to an optimal joint action [11].

D. RL-based DCS

The RL-based DCS (RL-DCS) enables each SU agent to select an available channel among the licensed channels for data transmission given that the objective is to maximize overall throughput and to minimize number of channel switchings in the presence of channel heterogeneity characteristics including PUL and PER, which is agent dependent, as well as the channel contention level.

The RL model for each agent in the DCS scheme is shown in Table I. The action A is to choose one of the K available licensed channels for data transmission. The reward $r_{t+1}(a_t) = N_D / t_D$ is the amount of throughput obtained within the recent epoch t, where N_D is the number of packets successfully transmitted by the transmitter T_i within the epoch. Data packet transmission is considered successful when a link layer acknowledgment is received for the data packet sent. In addition, a transmission is considered unsuccessful if a chosen channel is reoccupied by the PU immediately prior to transmission.

TABLE I
RL Model for Each Agent in DCS

	DCS Model	
	Description	Representation
Action	Available channels for data transmission.	$A = F = \{a = 1,2,\ldots, K\}$
Reward	Throughput within t_D.	$r_{t+1}(a_t) = N_D / t_D$

The RL approach helps an agent to adapt to its dynamic and uncertain operating environment. In reality, the radio resources, channel heterogeneity characteristics and other factors affect an agent's performance in a complex manner. Rather than addressing a single factor at a time, an agent observes all the factors and optimizes a general goal as a whole, such as throughput. The RL also adopts a simple modeling approach. Thus, the complexity involved in modeling the environment and channel heterogeneity can be minimized. For instance, an agent does not model the channel behavior characterized by channel selective fading, path loss and PU interference.

IV. Cognitive MAC Protocol with DCS Implementation

Each SU is equipped with two transceivers, namely a control transceiver and a data transceiver, thus it is capable of accessing two different channels simultaneously. The control transceiver is tuned to a common channel for control message exchange; while the data transceiver is tuned to one of the available data channels in the licensed bands for data packet transmission. The PU activities exist in the data channels only. We apply a Carrier Sense Multiple Access (CSMA)-based cognitive MAC and the reader is referred to [10] for its details. In the next few subsections, we present for later comparison three types of cognitive MAC based on different methods of DCS, namely Random MAC (RMAC), SARL-based MAC (SMAC), and MARL-based MAC (MMAC). For each subsection, the mechanism of channel switching, DCS, as well as the operation of the control and data transceiver are described.

A. RMAC

In R-MAC, the DCS chooses a data channel randomly. There are two conditions that trigger channel switching at agent i. Firstly, an unsuccessful data packet transmission at the data interface when a T_i fails to receive an ACK after a data packet transmission. Secondly, an agent must change its channel at least once every second. This avoids all the agents choosing a particular channel with low PUL and PER that provides higher occurence of successful data packet transmission at the expense of lower throughput due to a high level of contention. In addition, an agent does not switch channel within a duration of two data transmission cycles right after a channel switching.

B. SMAC

In SMAC, the DCS applies the SARL approach to choose a data channel. Each agent divides the time horizon into epochs and keeps track of the number N_D of successful data packet transmissions in the past epoch. No synchronisation is required among the agents. At the beginning of each epoch, an agent uses N_D to update its Q-value using (1) and chooses its channel in the next epoch using (2) with probability $1-\varepsilon$. During exploitation, in order to improve stability, an agent does not switch its channel if the difference between the Q-value of its previous exploitation channel and the current optimal channel using (2) is less than a small threshold value of β. For exploration, an agent is not allowed to explore for two consecutive epochs. Although an agent has decided to switch its channel at the beginning of an epoch, it is only carried out in the midst of an epoch when a new control transmission cycle starts, which is subject to contention among the agents. Hence, immediately prior to a channel switching, the T_i must update the Q-value of its initial channel which has been learned. Upon channel switching, it sets $N_D = 0$ and continues to operate in the epoch.

C. MMAC

According to [3], using SARL would result in instability or oscillations in the presence of multiple agents because an agent switches channel from time to time. MMAC addresses two drawbacks in SMAC that contribute to the instability. The next two subsections present improvement on stability and the PME mechanism, which is the payoff mechanism.

1) Improving Stability in SMAC: Firstly, when several agents undertake exploration at the same time, the Q-values (or the throughput performance) become unstable and they do not portray the exact level of PUL, PER and contention of the channels. For instance, when two agents explore a particular channel, the Q-value for the channel reduces for all agents and does not portray the exact level of contention. Secondly, an agent that explores a particular channel, and then exploits the other one in the following epoch causes the Q-values of both channels in itself and its neighbour agents to fluctuate.

One of the purpose of MMAC is to provide stability to the existing Q-learning approach. The instability is caused by the exploration. To tackle the first drawback, an agent would only explore if its neighbour agents are not exploring, and it must announce to its neighbour agents in a CTRL packet when it starts and terminates its exploration. This is to ensure that there is only a single agent undergoing exploration within a neighbourhood. To tackle the second drawback, the exploring agent and its neighbour agents must update and store the Q-tables and set $N_D=0$ during channel switching in order to learn a new environment whenever the exploration begins. At the end of the exploration, using (5), the exploring agent chooses to exploit the channel being explored or to exploit the other channel. The agent would have to retrieve its stored Q-table and set $N_D=0$ if it chooses to exploit the other channel, otherwise it would maintain its Q-table. The decision is broadcast to the neighbour agents using CTRL so that the neighbour agents follow suit to retrieve or maintain their Q-tables, and to set $N_D=0$.

2) PME Mechanism: Each agent divides the time horizon into epochs comprised of t_D and t_C. Each agent keeps track of the number N_D of successful data packet transmissions within t_D, and exchanges payoff messages (3) during t_C. The Q-values in the payoff message indicates the performance of each agent during exploitation or the recent exploration if any of the agents is undergoing exploration. No synchronisation is required among the agents although the neighbour agents send at least one payoff message to inform the exploring agent of their respective Q-value if any of the agents is undergoing exploration. At the beginning of each epoch, which is the end of t_C and the beginning of the next t_D, an agent updates its Q-values using N_D and payoff messages received from its neighbours. Equation (1) is used to update the Q-values, and the payoff message is used to update the stored μ-values. During exploitation, the optimal channel is chosen using (5).

V. Simulation Experiments, Results, and Discussions

A. Simulation Model, Assumptions and Parameters

We have implemented a CR-enabled environment in OMNeT++ [12]. The simulation scenario is shown in Section II. Due to the limited sensing capability at each SU node, the number of available data channels is set to K. Simulation parameters are shown in Table II.

TABLE II
Notations and Default Parameter Settings in Simulation

Category	Symbol	Details	Values
Initialization	U	Number of agents	{3,6,12}
	K	Number of available channels	3
	L	PUL of each available channel	[0, 0.9] Default = 0.5
	P_i	PER of each available channel at agent i	[0, 0.3] Default = 0.15
	T	Total simulation time	100s
SU	$t_{DATA.SU}$	Data packet duration	5.44ms
	$t_{CTRL.SU}$	CTRL packet duration	272μs
	T_{CSD}	Channel switching delay and initial channel sensing	2ms
PU	$t_{DATA.PU}$	Data packet duration	5.44ms
		Maximum queue size	5
RL	t_D	Epoch duration	187.14ms
	α	Learning rate	{0.05,0.1,0.2,0.4} Default: 0.2
	ε	Exploration probability	{0.05,0.1,0.2,0.4} Default: 0.2
	β	Q-value threshold value	1
		Initial Q-value	1
	Q_{max}	Maximum Q-value	20

Three levels of network densities are simulated with $d=U/K=\{1,2,4\}$. For SU, the CTRL is a small packet with $t_{CTRL,SU}$ duration and it contains information related channel switching and payoff message. The PU traffic model follows a Possion distribution with the mean arrival rate determined according to PUL, and it is independent and identically distributed (i.i.d.) across the available channels; while the SU T_i is always backlogged. The PUs broadcast data packets throughout the entire simulation area whenever they have packets. The PUs do not use four-way handshaking. An epoch duration is 30 data transmission cycles, or $t_D=30\times(t_{RTS}+t_{CTS}+t_{DATA,SU}+t_{ACK}+(3\times t_{SIFS}))$. In MMAC, the PME duration is $t_C=1.2\times(|\Gamma(i)|\times t_{CTRL,SU})$. Each agent observes different levels of PER across different channels with the default average value of PER across the K channels being 0.15. Upon receiving a packet, an SU discards the packet with the PER probability.

B. Performance Metrics

Our goal is to maximize overall throughput over different heterogeneous channels with different levels of PUL, as well as PER at different agents. The mean amount of

throughput per agent of RMAC, SMAC and MMAC are compared. The number of channel switchings of exploitation channel is measured for SMAC and MMAC to show the level of stability. Note that channel switching for exploration purpose is not counted. Jain's fairness index is applied to evaluate the fairness among the throughput achieved by each agent in the entire network. Denote the throughput achieved by agent i by x_i, the Jain's fairness index [13] is as follows:

$$f(x_1, x_2, \ldots, x_u) = \frac{(\sum_{i=1}^{u} x_i)^2}{(u \sum_{i=1}^{u} x_i^2)} \quad (8)$$

where $0 \leqslant f(x_1, x_2, \ldots, x_u) \leqslant 1$, and $f(x_1, x_2, \ldots, x_u) = 1$ when all agents achieve the same level of throughput. Graphs are presented with PUL and PER as ordinate. For each value of PUL and PER, the corresponding throughput or number of channel switchings is the average value of 50 runs using different levels of PULs and PERs across the $K = 3$ channels. For instance, a PUL level of 0.2 may indicate the PUL of [0.025,0.248,0.327] or [0.163,0.402,0.035] in the channels.

C. Simulation Results

Simulation results are presented in four subsections.

1) Stabilization of Global Q-value and payoff value: Figure 3 shows that the instantaneous global Q-value for the exploitation channel for SMAC and MMAC increase and become stable as time goes by in a medium density network. In other words, the agents attain a better joint action. The PUL is $L = 0.5$ with [0.5,0.5,0.5] across $K = 3$ channels, and mean PER at agent i is $P_i = 0.15$ every channel. The Q-learning parameters are $\alpha = 0.2$ and $\varepsilon = 0.2$. With $U = 6$ and $Q_{max} = 20$, the maximum global Q-value is 120. Although $L = 0.5$ for all data channels, due to the Poisson traffic model, the channels have different levels of PUL at a particular time. Although the MMAC aims to increase global Q-value; while SMAC aims to increase local Q-value, SMAC achieves slightly higher global Q-value compared to MMAC. This is because SMAC can explore the channels at any time to discover a better channel; while in MMAC, an agent can only explore if none of its neighbour agents is doing so.

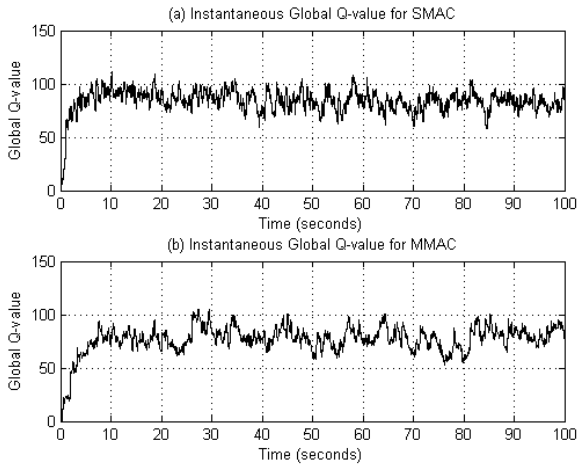

Figure 3. Global Q-value for the exploitation channel for SMAC and MMAC in a medium density network.

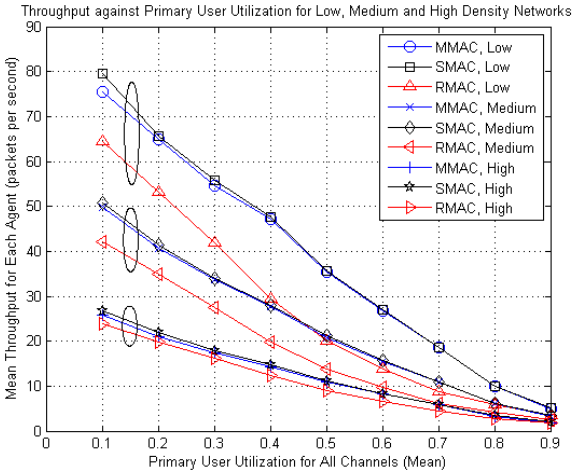

Figure 4. The mean throughput for each agent against mean of PUL for MMAC, SMAC, and RMAC in low, medium and high density networks.

2) Effects of Network Density on Network Performance: Figure 4 shows the mean throughput for each agent against various levels of mean PUL for MMAC, SMAC, and RMAC in low, medium, and high density networks. Mean PER at agent i is $P_i = 0.15$ every channel. The Q-learning parameters are $\alpha = 0.2$ and $\varepsilon = 0.2$. The MMAC and SMAC achieves approximately similar throughput, followed by RMAC in all types of network densities; and the throughput enhancement offered by the MMAC and SMAC reduce as the network density increases. At PUL $L = 0.5$, the MMAC outperforms the RMAC by 1.77 times, 1.5 times, and 1.2 times in low, medium and high density networks respectively. Figure 5 shows the equivalent graph with PER as ordinate and PUL is $L = 0.5$ with [0.5,0.5,0.5], and similar trend is observed. In short, in a high density network or $d \to \infty$, the throughput enhancement achieved by MMAC and SMAC approaches 0. We believe that this happens in most intelligence methods due to the high contention level.

Figure 6 shows the mean number of channel switchings of exploitation channel for each agent against various levels of mean PUL for MMAC and SMAC in low, medium, and high density networks. Mean PER at agent i is $P_i = 0.15$

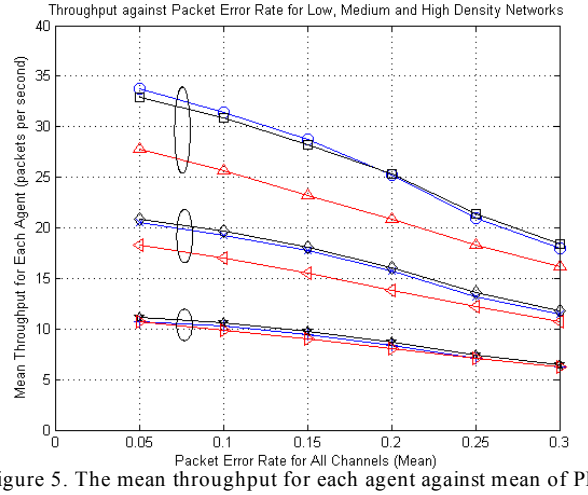

Figure 5. The mean throughput for each agent against mean of PER for MMAC, SMAC, and RMAC in low, medium and high density networks. See legend in Figure 4.

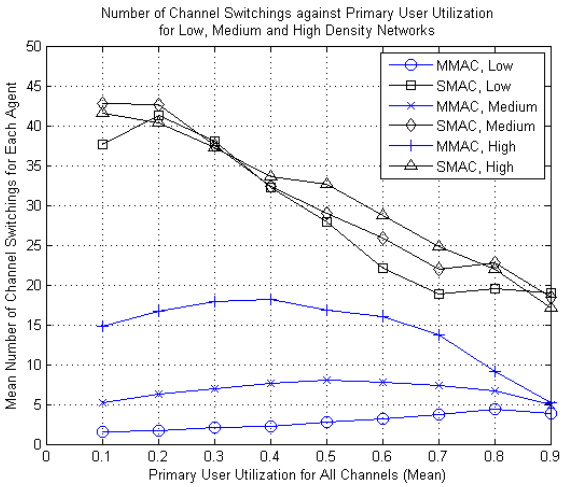

Figure 6. The mean number of channel switchings of exploitation channel for each agent against mean of PUL for MMAC and SMAC in low, medium and high density networks.

every channel. The Q-learning parameters are $\alpha=0.2$ and $\varepsilon=0.2$. The MMAC achieves significantly lower number of channel switchings, hence it provides higher stability. At PUL $L=0.5$, the number of channel switchings in SMAC is 10 times, 3.58 times and 2 times of that in MMAC in low, medium and high density network respectively. Generally speaking, an agent switches its exploitation channel because the difference between the Q-values among the channels is greater than the threshold $\beta=1$, and the agent exploits a better channel. There are two reasons an agent does not switch channel. Firstly, all the channels provide equal level of performance, hence an agent exploits the same channel. Secondly, all the channels provide very good or very poor performance, and hence the Q-values approach the Q-value's limit, specifically, $Q_t(a)\rightarrow 0$ or $Q_t(a)\rightarrow Q_{max}$ for $\forall a\in A$. For instance, MMAC and SMAC have lower number of channel switchings as the PUL increases because $Q_t(a)\rightarrow 0$ for all channels. The MMAC also increases network stability [3] through reducing the number of channel switching. Figure 7 shows the equivalent graph with PER as ordinate and PUL [0.5,0.5,0.5].

3) Fairness Index of MMAC and SMAC: With respect to PUL in Figure 8 and PER in 9, RMAC achieves the highest fairness, while MMAC and SMAC achieves approximately

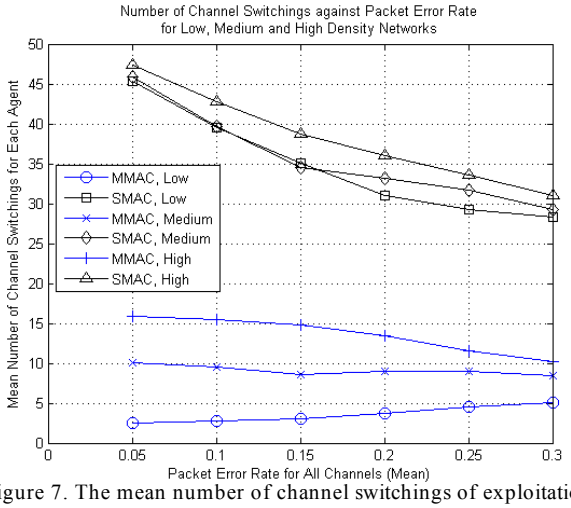

Figure 7. The mean number of channel switchings of exploitation channel for each agent against mean of PER for MMAC and SMAC in low, medium and high density networks.

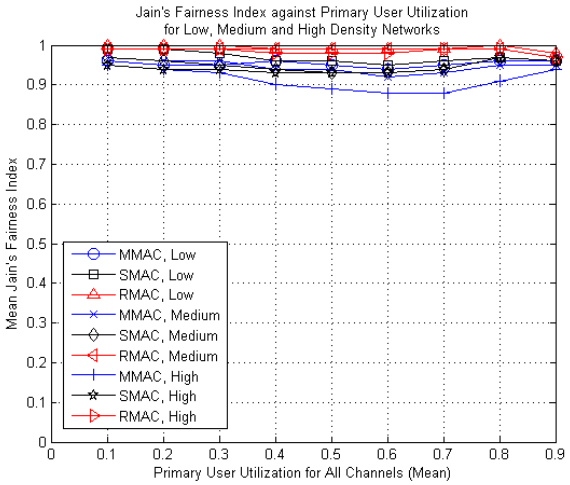

Figure 8. The mean Jain's Fairness Index against mean of PUL for MMAC, SMAC, and RMAC in low, medium and high density networks.

similar high levels of fairness. In RMAC, all agents choose channel randomly, hence the Jain's fairness index approaches to nearly 1. For MMAC and SMAC, some agents may choose better channels compared to others, hence the Jain's fairness index is lower than that in RMAC.

4) Effects of α and ε on Network Performance: The effect of α and ε on throughput is insignificant in most cases, and its graph is not provided. Figure 10 shows the effect of α on the mean number of channel switchings of the exploitation channel for each agent against various levels of mean PUL for MMAC and SMAC in a medium density network. Mean PER at agent i is $P_i=0.15$ every channel. The number of channel switchings increases with α for all cases. In short, lower value of α provides higher stability. Figure 11 shows the equivalent graph with PER as ordinate and PUL $L=0.5$ with [0.5,0.5,0.5]. A similar experiment is performed to investigate the effects of ε on the mean number of channel switchings, with results shown in Figures 12 and 13, which share similar trends to Figure 10 and 11 respectively.

Conclusions

In this paper, we achieve context awareness and intelligence in Distributed Cognitive Radio Networks (DCRNs) using Reinforcement Learning (RL). Both single-agent-based approach (SMAC) and multi-agent-based

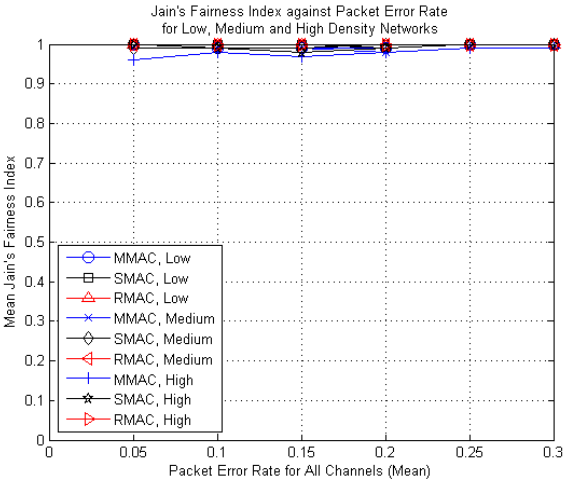

Figure 9. The mean Jain's Fairness Index against mean of PER for MMAC, SMAC, and RMAC in low, medium and high density networks.

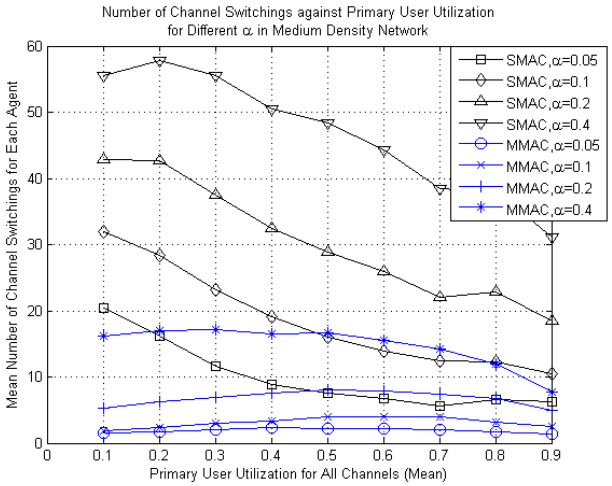

Figure 10. The mean number of channel switchings of exploitation channel for each agent against mean of PUL for MMAC and SMAC with different α in a medium density network.

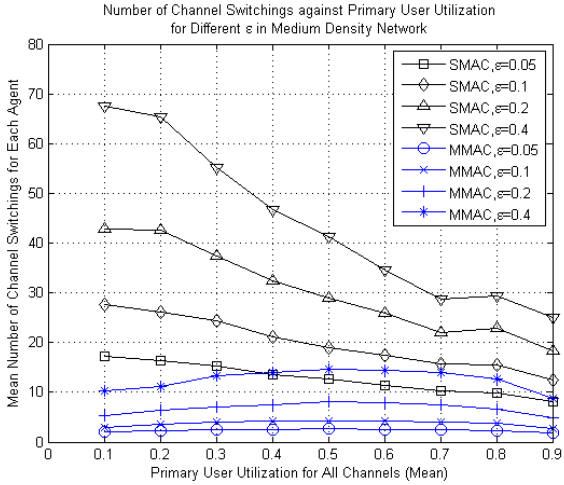

Figure 12. The mean number of channel switchings of exploitation channel for each agent against mean of PUL for MMAC and SMAC with different ε in a medium density network.

approach (MMAC) are investigated. RL is suitable to be applied in most application schemes, and we investigate its performance in respect to Dynamic Channel Selection (DCS). The MMAC and SMAC approaches achieve stable joint actions as the global learned value or Q-value increases and becomes stable. Both MMAC and SMAC provide network-wide performance enhancement: approximately similar level of throughput, and MMAC is more stable with significant reduced number of channel switchings. The performance enhancement reduces as network density increases. Both MMAC and SMAC achieve approximately similar high level of Jain's fairness index. The essential parameters including learning rate α and exploration probability ε in RL are investigated. Lower value of α and ε provides better throughput and stability.

References

[1] J. Mitalo and G. Q. Maguire, "Cognitive radio: making software radios more personal," *IEEE Prsnl. Comm.*, 6, pp. 13-18, Aug. 1999.

[2] R. S. Sutton and A. G. Barto, Reinforcement learning: an introduction. Cambridge MA, MIT Press, 1998.

[3] J. R. Kok, and N. Vlassis "Collaborative Multiagent Reinforcement Learning," *Jnl. Mach. Learn. Rsrch 7*, pp. 1789-1828, Sep. 2006.

[4] A. Alaya-Feki, E. Moulines, and A. LeCornec, "Dynamic spectrum access with non-stationary multi-armed bandit", *9th IEEE Wrkshp on Sgnl. Prcss. Advc. in Wls. Comm. (SPAWC)*, 2008.

[5] Y. B. Reddy, "Detecting primary signals for efficient utilization of spectrum using Q-learning", *5th Intl. Conf. on IT: Nt. Gen. (ITNG)*, 2008.

[6] T. Jiang, D. Grace, and Y. Liu, "Performance of cognitive radio reinforcement spectrum sharing using different weighting factors", *3rd Intl. Conf. on Comm. and Nwk. in China (ChinaCom)*, 2008.

[7] M. Yang, and D. Grace, "Cognitive radio with reinforcement learning applied to heterogeneous multicast terrestrial communication systems," *4th Intl. Conf. on Cog. Rad. Otd. Wls. Nwk. & Comm. (CROWNCOM)*, 2009.

[8] U. Berthold, F. Fu, M. v. d. Schaar, and F. K. Jondral, "Detection of spectral resources in cognitive radios using reinforcement learning", *3rd Sym. on N. Frontiers in Dy. Spec. Acs. Nwk (DySPAN)*, 2008.

[9] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "A context-aware and intelligent dynamic channel selection scheme for cognitive radio networks," *4th IEEE Intl. Conf. on Cog. Rad. Orntd. Wls. Nwk. & Comm. (CROWNCOM)*, June 2009.

[10] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Context-awareness and intelligence in distributed cognitive radio networks: a reinforcement learning approach," *IEEE Aust. Comm. Thry. Wksp. (AUSCTW)*, Feb 2010.

[11] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Achieving efficient and optimal joint action in distributed cognitive radio networks using payoff propagation," *IEEE Intl. Conf. on Comm. (ICC)*, May 2010.

[12] INET Framework for OMNet++/OMNEST release 2006-10-12. http://www.omnetpp.org/doc/INET/.

[13] Jain, R., Chiu, D.M., and Hawe, W. (1984) *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems*. DEC Research Report TR-301.

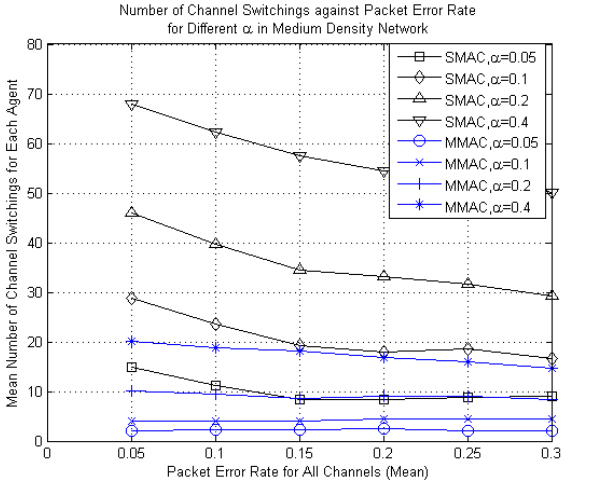

Figure 11. The mean number of channel switchings of exploitation channel for each agent against mean of PER for MMAC and SMAC with different α in a medium density network.

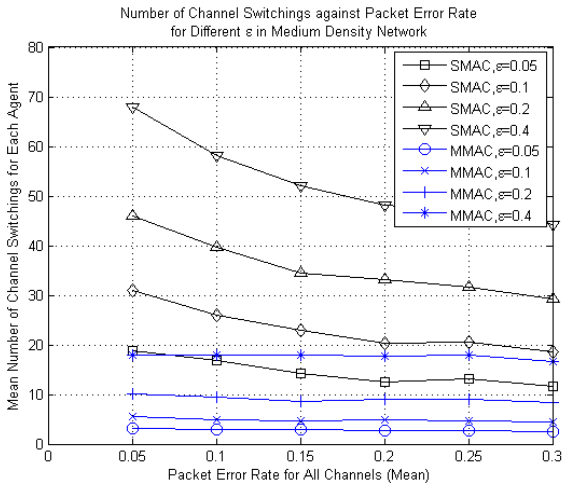

Figure 13. The mean number of channel switchings of exploitation channel for each agent against mean of PER for MMAC and SMAC with different ε in a medium density network.