Seleção da Ordem de Sensoreamento de Canais em uma Rede Cognitiva Oportunista

André Chaves Mendes[1], **Carlos Henrique Pereira Augusto**[1], **Marcel William Rocha da Silva**[1], **Raphael Melo Guedes**[1], **José Ferreira de Rezende**[1]

[1]GTA - PEE - COPPE – Universidade Federal do Rio de Janeiro (UFRJ)
Caixa Postal 68.504 – 21.945-970 – Rio de Janeiro – RJ – Brasil

{andre,chenrique,marcel,raphael,rezende}@gta.ufrj.br

Abstract. *This work investigates the problem of channel sensing order used by a cognitive multichannel network, where each user is able to perform primary user detection on only one channel at a time. The sensing order indicates the sequence of channels sensed by the secondary users when searching for an available channel. Brute-force algorithms may be used to find the optimal sensing order, but it requires great computational effort. Even in scenarios where the secondary user knows the probability of each channel being available, the sensing order where the most available channels are sensed first is not ideal when using adaptive modulation. Therefore, we propose and evaluate an approach using reinforcement learning to search dynamically for the optimal sensing order, comparing its performance with other mechanisms, and the results obtained are close to the optimal value provided by the brute-force and superior to the other mechanisms in most of the scenarios.*

Resumo. *Este trabalho investiga o problema da escolha da ordem de sensoreamento dos canais utilizados por uma rede cognitiva multicanal, onde cada usuário é capaz de realizar o sensoreamento em apenas um canal por vez. A ordem de sensoreamento indica a sequência de canais sensoreados pelos usuários secundários na busca por um canal disponível para uso. Algoritmos de força-bruta podem ser utilizados para encontrar a melhor ordem de sensoreamento, porém esta abordagem exige grande esforço computacional. Mesmo em cenários onde o usuário secundário tem conhecimento da probabilidade de disponibilidade de cada um dos canais, a ordem de sensoreamento onde os canais são ordenados pela sua disponibilidade decrescente não é a ideal quando se usa modulação adaptativa. Assim, propomos uma abordagem utilizando aprendizado por reforço para busca dinâmica da ordem de sensoreamento ótima e a avaliamos, comparando o seu desempenho com o de outros mecanismos, e os resultados obtidos estão próximos do valor ótimo fornecido pela força-bruta e superior ao dos demais mecanismos, para a maioria dos cenários.*

1. Introdução

A crescente demanda por faixas do espectro de radiofrequências aliada ao uso ineficiente das bandas licenciadas [M. A. McHenry 2005] impulsionaram a ideia de se liberarem essas bandas de frequências subutilizadas para acesso dinâmico oportunístico [FCC 2003].

Nesse modo de acesso, faz-se necessário o uso de dispositivos de rede reconfiguráveis, denominados rádios cognitivos [J. Mitola III and G. Q. Maguire Jr. 1999], capazes de adaptar dinamicamente seus parâmetros e modos de operação às condições do ambiente onde eles se encontram. Basicamente, o rádio cognitivo somente acessa uma determinada faixa de frequências quando os usuários licenciados para uso desta faixa estiverem inativos. Para isso, os usuários dotados do rádio cognitivo, denominados usuários secundários, necessitam determinar quando os usuários licenciados estivarão ativos, evitando causar-lhes interferência de radiofrequência. Portanto, o sensoreamento do espectro é parte importante no funcionamento desses dispositivos.

Num cenário de múltiplos canais e usuários secundários dotados de um único transceptor, apenas um canal pode ser sensoreado por vez para detectar possíveis oportunidades de uso. Nesse cenário, a ordem de sensoreamento dos canais pode ter um grande impacto no desempenho da rede secundária. Assim, a busca pela ordem de sensoreamento ótima é um problema de grande importância e interesse.

Vários trabalhos já estudaram esse problema [S. Guha et al. 2006, H. Kim and K. G. Shin 2008, J. Jia et al. 2008, H. Jiang et al. 2009, Ho Ting Cheng and Weihua Zhuang 2011]. E na sua maioria, esses trabalhos fazem uso da teoria da parada ótima (*optimal stopping*) [Chow et al. 1971] para encontrar a melhor sequência de sensoreamento. Nesses trabalhos, o tempo é dividido em *slots*, e em cada *slot* de tempo, os canais são sensoreados seguindo uma determinada sequência até que um canal livre seja encontrado, não sendo permitido o retorno a um canal sensoreado previamente (*recall*). O restante do tempo do *slot* é então utilizado para a transmissão. Pelo conhecimento a priori das taxas alcançáveis e das probabilidades de disponibilidade de cada canal (indicativa da atividade dos primários), é possível se determinar a recompensa esperada no uso de uma sequência de canais em termos da taxa de transmissão efetiva, ou seja, do produto da taxa alcançável e da efetividade de uso do *slot*. Assim, a sequência ótima pode ser encontrada calculando-se a recompensa esperada de cada uma das sequências possíveis e escolhendo-se a de maior recompensa. No entanto, a complexidade desse algoritmo por força bruta é de $O(N.N!)$, considerando-se que o cálculo da recompensa esperada para cada sequência é de $O(1)$.

Com o intuito de diminuir a complexidade computacional dessa busca pela ordem de sensoreamento ótima, os autores em [H. Jiang et al. 2009, Han Han et al. 2010] fornecem soluções sub-ótimas. A primeira solução utiliza programação dinâmica [H. Jiang et al. 2009] e tem uma complexidade de $O(N.2^{N-1})$, enquanto a segunda faz uso de árvores de decisão [Han Han et al. 2010] com uma complexidade de $O(N^3)$. Essas duas soluções são comparadas nesse último trabalho, além das sequências aleatória e em ordem decrescente de disponibilidade, denominada como "sequência intuitiva" em [H. Jiang et al. 2009].

Em [Ho Ting Cheng and Weihua Zhuang 2011], os autores propõem o uso da sequência de canais em ordem decrescente de suas taxas alcançáveis, não necessitando assim do conhecimento a priori da atividade dos primários. É demonstrado no trabalho que se a regra de parada utilizada for a do primeiro canal livre, a melhor recompensa dessa sequência é alcançada. Na solução apresentada em [J. Jia et al. 2008], todos os canais possuem a mesma probabilidade de disponibilidade. Neste caso, o problema de ordem de sensoreamento se reduz ao uso da sequência de canais em ordem decrescente de suas

taxas alcançáveis, como em [Ho Ting Cheng and Weihua Zhuang 2011]. Todos esses trabalhos têm a deficiência de precisarem do conhecimento a priori das taxas alcançáveis e/ou das probabilidades de disponibilidade de cada canal. Além disso, por apresentarem uma alta complexidade computacional com o aumento do número de canais, não é possível embarcá-los nos rádios cognitivos com facilidade.

Neste trabalho, propomos uma solução de baixa complexidade com um algoritmo baseado em uma máquina de aprendizagem por reforço (*reinforcement learning*), seguindo o modelo de recompensas que se utiliza da teoria da parada ótima [H. Jiang et al. 2009]. Essa solução não requer o conhecimento prévio dos momentos das variáveis de disponibilidade e capacidade dos canais, podendo se adaptar dinamicamente a variações desses momentos. Além disso, essa solução possui complexidade computacional baixa, o que a torna atrativa para ser embarcada em rádios cognitivos. A solução proposta é implementada num simulador próprio e avaliada em comparação à solução ótima e a outros mecanismos mais simples de ordenação. Os resultados obtidos demonstram que a solução proposta tem um desempenho máximo 5% inferior ao obtido pela sequência ótima e superior às demais sequências na maioria dos cenários.

No restante deste artigo, a seção seguinte descreve o modelo do sistema utilizado. A Seção 3 fornece os conceitos básicos em aprendizado por reforço para em seguida apresentar a proposta que utiliza esta técnica para a busca da ordem de sensoreamento ótima. A Seção 4 descreve o ambiente de simulação e mostra os resultados obtidos. Finalmente, a Seção 5 conclui o artigo e enumera trabalhos futuros.

2. Modelo do Sistema

Nesta seção, será descrito o modelo do sistema, similar ao apresentado em [H. Jiang et al. 2009], utilizado para o desenvolvimento e implementação da nossa proposta. Esta modelagem permite determinar a sequência de sensoreamento ótima através da aplicação da teoria da parada ótima (*optimal stopping*). Com isso, o objetivo é fornecer uma decisão sobre o momento para finalizar o sensoreamento de novos canais de tal forma que a recompensa obtida na escolha de um canal seja maximizada. Essa teoria permite então definir a regra de parada que maximiza a recompensa. Além disso, pelo fato do número de canais a serem sensoreados ser finito e igual a N, o método de indução reversa (*backward induction*) pode ser aplicado para encontrar a recompensa esperada de uma sequência de N canais.

Considere um usuário secundário e um número finito de canais, N. Esse usuário possui um modelo de funcionamento baseado em *slots de tempo*, ou seja, o tempo é dividido em *slots* de duração T. Em cada *slot*, cada canal possui a probabilidade p_i de estar livre da atividade de usuários primários. Assume-se que o estado de um canal, livre ou ocupado, em um *slot* é independente do seu estado prévio e do estado de outros canais. Consideramos também que, devido ao efeitos de desvanecimento nos canais, a relação sinal-ruído (SNR) obtida em um canal varia aleatoriamente entre *slots*. Assumimos que esta SNR aleatória é i.i.d entre os *slots* e os diferentes canais, e que é regida por uma distribuição arbitrária. Se o usuário secundário decidir transmitir em um canal considerado livre, c_i, a taxa de transmissão obtida será função da SNR momentânea desse usuário nesse canal. Esta função, $F(SNR_i)$, mapeia de forma monotônica e crescente a SNR do canal c_i na taxa de transmissão que será obtida neste canal.

Antes de decidir utilizar um canal em um determinado *slot*, o usuário secundário deve realizar o sensoreamento do canal com a finalidade de determinar se nele existem usuários primários em atividade. No modelo, assume-se que o processo de sensoreamento é uma tarefa precisa e isenta de erros. Como não existe conhecimento prévio a respeito dos estados dos canais, o usuário secundário realiza o sensoreamento sequencial dos N canais, seguindo uma ordem pré-estabelecida, $\{o_1, o_2, ..., o_N\}$. A eficiência no uso de uma ordem de sensoreamento está relacionada com o tempo gasto no sensoreamento dos canais e com a taxa de transmissão momentânea que pode ser obtida no canal utilizado.

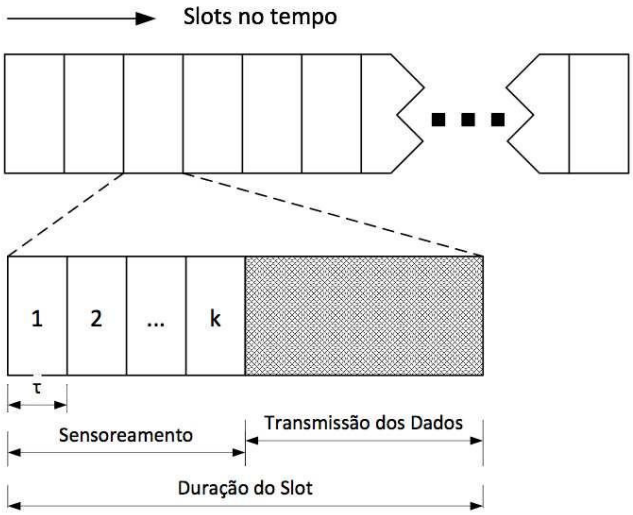


Figura 1. Processo de sensoreamento dos canais em um *slot*.

A Figura 1 exemplifica a atividade de um usuário secundário em um *slot*, o qual possui duas fases: uma fase de sensoreamento, e uma fase de transmissão de dados. O valor τ corresponde ao tempo necessário para o sensoreamento de cada canal. Durante a fase de sensoreamento, se o canal c_i é sensoreado como ocupado, o usuário secundário realiza o sensoreamento no canal c_{i+1}, que é o próximo canal na ordem de sensoreamento que está sendo utilizada. Entretanto, caso o canal c_i seja sensoreado como livre, a taxa de transmissão efetiva obtida pelo usuário secundário no uso deste canal durante o tempo remanescente do *slot* será dada por $e_i \times F(SNR_i)$, onde e_i é a efetividade da transmissão, calculada pela fórmula $e_i = \frac{T - i\tau}{T}$. Com isso, a recompensa no uso de cada canal na sequência pode ser dada por:

$$r_i = \begin{cases} e_i F(SNR_i) & se \;\; e_i F(SNR_i) > R_{i+1} \\ R_{i+1} & \text{nos demais casos} \end{cases} \tag{1}$$

onde R_{i+1} para $i \leq N - 1$ é a recompensa esperada caso o usuário decida prosseguir no sensoreamento. O cálculo da recompensa esperada é dado por:

$$R_{i+1} = \begin{cases} p_{i+1} E[r_{i+1}] + (1 - p_{i+1}) R_{i+2} & se \;\; i < N - 1 \\ p_{i+1} E[r_{i+1}] & se \;\; i = N - 1 \end{cases} \tag{2}$$

Repare que o conjunto de recompensas esperadas $\{R_1, R_2, ..., R_n\}$ pode ser obtida recursivamente a partir de R_N, através das Equações 1 e 2. Logo, R_1 representa a recompensa esperada pelo usuário secundário no uso de uma sequência de N canais. De forma genérica, R_i representa o valor esperado da recompensa no uso de uma sequência parcial de canais $(o_i, o_{i+1}, ..., o_N)$. Desta forma, o uso de um canal sensoreado como livre é vantajoso caso a recompensa no uso do canal r_i seja superior à recompensa esperada do restante da ordem de sensoreamento R_{i+1}. Caso contrário, o usuário deve prosseguir no sensoreamento do próximo canal da sequência, não podendo retornar ao canal anterior (*recall*).

3. Ordem de Sensoreamento Dinâmica Utilizando Aprendizado por Reforço

O mecanismo proposto neste trabalho utiliza aprendizado por reforço para determinar de maneira dinâmica uma ordem de sensoreamento a ser utilizada em cada *slot*. Uma das vantagens do mecanismo baseado em aprendizado por reforço é que não é necessário nenhum conhecimento prévio a respeito da probabilidade de cada canal estar disponível, nem da qualidade estimada de cada canal por meio de suas SNRs médias. Outra vantagem importante desta proposta é quanto a sua adaptabilidade às mudanças de características dos canais garantida pelo aprendizado com as tomadas de ação. Logo, o mecanismo torna-se imune às possíveis mudanças nas probabilidades de disponibilidade dos canais, que podem ocorrer devido a mudanças nos padrões de atividade dos usuários primários, e às possíveis mudanças na qualidade dos canais (SNRs médias), que podem ocorrer devido à mobilidade e aos efeitos de desvanecimento de larga escala.

Nas subseções seguintes, apresentaremos os conceitos básicos sobre a técnica de aprendizado por reforço e a nossa proposta, baseada nessa técnica, para a busca da ordem de sensoreamento ótima.

3.1. Aprendizado por Reforço

Aprendizagem por reforço (*reinforcement learning*) [Sutton and Barto 1998] é um tipo de máquina de aprendizagem que preocupa-se com a maneira com que um agente escolhe ações a serem realizadas, de acordo com informações de causa e efeito obtidas do ambiente. Resumidamente, nesse método temos agentes que ao inspecionarem um estado, em um espaço de estados, realizam alguma ação que se reflete em uma recompensa. A partir do valor da recompensa coletada, o agente aprende a qualidade da ação escolhida. O problema consiste em escolher ações que maximizem o total de recompensas recebidas pelo agente.

O *método de aprendizagem por reforço* adota uma abordagem simples, cuja complexidade envolvida na modelagem do problema pode ser minimizada [Kok-Lim Alvin Yau et al. 2010a], levando também a uma baixa complexidade computacional. Em contrapartida, este método pode apresentar uma convergência lenta. Entre as diversas técnicas de aprendizagem existentes [Sutton and Barto 1998], adotamos neste trabalho a técnica *Q-learning* por sua simplicidade, e por este motivo a apresentaremos em mais detalhes.

O *Q-learning* é um algoritmo em tempo de execução (*online*) que busca determinar, sem conhecimento prévio do ambiente, qual ação ou decisão melhor se aplica naquele instante. O modelo básico deste algoritmo consiste de:

- Instante de decisão, que representa o instante de execução do algoritmo, indicado por $t \in T$, $T = \{1, 2, ...\}$;
- Um conjunto de estados, próprios da modelagem de cada problema, indicados por $s \in S$;
- Um conjunto de ações, que representam as decisões que podem ser tomadas, indicadas por $a \in A$, e que levam a um novo estado;
- Regras que determinam a recompensa de uma ação em um dado estado, indicada por $r_t(s, a)$;
- Regras de transição entre estados.

Cada agente mantém uma *Q-table*, que é uma matriz com $|S| \times |A|$ entradas, onde as linhas representam os estados e as colunas indicam as ações. Os elementos dessa matriz são chamados *Q-value's*, $Q_t(s, a)$, que são valores atualizados através da coleta de recompensas, $r_t(s, a)$, sempre que um agente realiza uma ação a em um estado s.

O *Q-value* estima o nível de recompensa para o par estado-ação. Assim mudanças nos *Q-value's* levam a mudanças nas decisões de que ações devem ser tomadas pelos agentes. A cada instante de decisão t, o agente observa seu estado atual (linha) e escolhe uma ação (coluna) em sua *Q-table*. Posteriormente à execução de uma ação, o agente recebe uma recompensa r_t relativa a esta ação realizada.

A partir da recompensa obtida, o agente atualiza a respectiva entrada na *Q-table* no tempo $t + 1$ conforme:

$$\begin{aligned} &Q_{t+1}(s, a) = \\ &Q_t(s, a) + \alpha[r_t(s, a) + \gamma max_a Q_t(s_{t+1}, a) - Q_t(s, a)] \end{aligned} \tag{3}$$

Na Equação 3, α é chamado de parâmetro de aprendizagem, e γ é chamado de fator de desconto. Maiores valores de α indicam maior importância para a experiência recente em relação ao histórico. Valores maiores de γ indicam que o agente baseia-se mais na recompensa futura que na recompensa imediata [Kok-Lim Alvin Yau et al. 2010b]. O *Q-learning* inicia com a *Q-table* zerada e a cada instante o agente seleciona uma ação baseada em uma estratégia de exploração. Uma estratégia comumente usada é a *ε-greedy* [Sutton and Barto 1998], onde o agente utiliza a probabilidade dada por ε para decidir entre a exploração (*exploitation*) da *Q-table* ou a investigação (*exploration*) de estados aleatoriamente. Dado que ε é normalmente pequeno, na maioria dos casos, o agente seleciona de forma gananciosa a ação que satisfaça $max_a Q(s, a)$, ou seja, aquela que historicamente oferece a maior recompensa. No entanto, ocasionalmente, o agente seleciona uma ação aleatória com uma probabilidade ε. Esta estratégia tenta fazer com que todas as ações, e seus efeitos, sejam experimentados [Jelle R. Kok and Nikos Vlassis 2006].

3.2. Proposta

Um dos maiores desafios encontrados no emprego da ferramenta de aprendizado por reforço no problema da escolha da ordem de sensoreamento foi a modelagem dos estados e ações. Uma modelagem descuidada pode gerar um modelo com muitos estados e/ou muitas ações, o que tornaria lenta a convergência do processo de exploração. No nosso modelo, definimos o estado como o par ordenado formado pela posição na ordem

de sensoreamento, o_k, e o canal que é sensoreado naquela posição, c_i. As ações possíveis de serem tomadas por um usuário secundário a partir de um estado (o_k, c_i) correspondem a escolher o canal que será sensoreado na próxima posição da ordem de sensoreamento, o_{k+1}. Com isso, a *Q-table* será uma matriz de dimensões $N^2 \times N$ ($estados x$ações). Repare que essa modelagem faz com que não exista um estado ótimo a ser alcançado, mas sim uma sequência de ações que maximizam a recompensa imediata a cada estado, criando uma ordem de sensoreamento dinâmica.

Algumas restrições devem ser levadas em consideração no momento das tomadas de ação e atualização da *Q-table*. Uma delas é que uma ação tomada no estado $(o_k, *)$, com $1 \leq k \leq (N-1)$, sempre leva a um estado onde a posição na ordem de sensoreamento é o_{k+1}. No estado onde a posição é $(o_N, *)$, que representa o último canal da ordem de sensoreamento, as ações indicam o primeiro canal a ser sensoreado no próximo *slot*, ou seja, leva a um estado onde a posição na ordem de sensoreamento é a $(o_1, *)$. Quando o usuário secundário decide usar um canal c_i na posição o_k, o sensoreamento nesse *slot* é finalizado. Nesse caso, o primeiro canal a ser sensoreado no próximo *slot* será determinado pela melhor ação no estado (o_N, c_i). Outra restrição importante é com relação a impedir o retorno a um canal sensoreado previamente (*recall*). Para isso, é necessário armazenar os canais já sensoreados no *slot* corrente. Desta forma, antes de tomar uma ação, o usuário secundário deve eliminar, das ações possíveis, os canais já sensoreados.

Outra parte importante do modelo diz respeito à recompensa obtida em cada canal, o qual será utilizado para atualizar a *Q-table*, de acordo com o modelo de recompensa apresentado na Seção 2. Quando o canal c_i é sensoreado como livre na posição o_k da ordem de sensoreamento, a recompensa obtida r_t será dada pela taxa de transmissão efetiva obtida pelo uso daquele canal, $e_k \times F(SNR_i)$. No caso em que o canal é detectado como ocupado, é necessário que exista alguma penalização para reduzir o *Q-value* referente àquela ação. Desta forma, introduzimos o parâmetro δ, que assume valores no intervalo $[0, 1]$ e multiplica o *Q-value* atual referente àquela ação. Assim, garante-se que quando uma ação leva a um canal ocupado, o *Q-value* referente àquela ação será reduzido. Essa estratégia faz com que o *Q-value* represente não apenas a taxa de transmissão efetiva, mas também a disponibilidade dos canais. Logo, os valores da *Q-table* são atualizados da seguinte forma:

$$Q_{t+1}(s,a) = \begin{cases} (1-\alpha) \times Q_t(s,a) + \alpha \times r_t(s,a) & se \text{ canal livre} \\ \delta \times Q_t(s,a) & se \text{ canal ocupado} \end{cases} \tag{4}$$

O funcionamento do mecanismo é descrito em detalhes no Algoritmo 1. No início, todos os pares estado-ação da *Q-table* são completados com zeros. Realizada esta fase de inicialização, começa a fase de aprendizado, que é repetida durante todo o período de funcionamento do mecanismo. Nesta fase, toma-se a decisão entre *investigação*, onde uma ação é escolhida aleatoriamente, e *exploração*, onde a melhor ação é escolhida, baseando-se na *Q-table*. Após a execução da ação, o mecanismo torna-se capaz de calcular a recompensa obtida e atualizar o correspondente *Q-value*.

Uma característica importante da nossa proposta, diz respeito ao uso dos canais

```
1  /* inicializa Q-table */
2  foreach s ∈ S, a ∈ A do
3      Q(s,a) = 0;
4  while (1) do
5      /* aprendizado */
6      sorteia número aleatório x entre 0 e 1;
7      if (x < ε) then
8          /* exploração */
9          seleciona uma ação a aleatoriamente;
10     else
11         escolhe ação a que possua o maior Q-value para o estado atual s;
12     if (canal livre) then
13         /* canal c_a correspondente à ação a */
14         calcula recompensa r_t(s, a);
15         Q_{t+1}(s, a) ← (1 − α) × Q_t(s, a) + α × r_t(s, a);
16         if r_t(s, a) > maxQ(s', a') then
17             /* usa canal c_a */
18             finaliza slot;
19         else
20             /* não usa canal c_a */
21             continua sensoreamento;
22     else
23         /* canal c_a ocupado */
24         Q_{t+1}(s, a) ← δ × Q_t(s, a);
25         continua sensoreamento;
26     s_t = s_{t+1};
```

Algoritmo 1: Mecanismo proposto baseado em aprendizado por reforço.

sensoreados como livres. De acordo com o modelo apresentado na Seção 2, a regra de parada ótima consiste em verificar se a recompensa instantânea é maior do que a recompensa esperada para o restante da sequência. Isso indica que nem sempre será vantajoso utilizar o primeiro canal livre encontrado. De forma similar, a nossa proposta também utiliza um critério de parada que consiste em comparar a recompensa atual, r_t, com o melhor *Q-value* das ações possíveis a partir daquele estado. Assim, é possível estimar se a recompensa do canal livre atual é superior à recompensa esperada da melhor ação existente. Repare que mesmo no caso onde o canal livre não é utilizado, o *Q-value* referente àquela ação também é atualizado.

4. Resultados Numéricos

Para avaliar o comportamento do mecanismo de aprendizado por reforço na solução do problema da ordem de sensoreamento, construímos um simulador utilizando a linguagem Tcl [John Ousterhout 1988]. Nesse simulador, as seguintes ordens de sensoreamento foram avaliadas: a sequência dinâmica dos canais fornecida pela nossa proposta (`RL`), a sequência ótima obtida por força-bruta (`Ótima`), a

sequência dada pela ordem decrescente das capacidades médias de cada canal (`Cap`) [Ho Ting Cheng and Weihua Zhuang 2011], a sequência de canais na ordem decrescente de suas probabilidades de disponibilidade (`Prob`), a sequência dada pela ordem decrescente do produto de suas capacidades médias pela probabilidade de disponibilidade de cada canal (`ProbxCap`), e a sequência na ordem crescente de numeração dos canais (`Aleatória`). Vale ressaltar que todas as sequências acima, com exceção da sequência `RL`, são estáticas, ou seja, não mudam durante toda a simulação. No caso da sequência `RL`, devido ao próprio aprendizado por reforço, a sequência pode variar durante a simulação. Além disso, todas as sequências, exceto na `RL` e `Aleatória`, assumem o conhecimento a priori das capacidades médias de cada canal e/ou de suas probabilidades de disponibilidade.

4.1. Modelo de Simulação

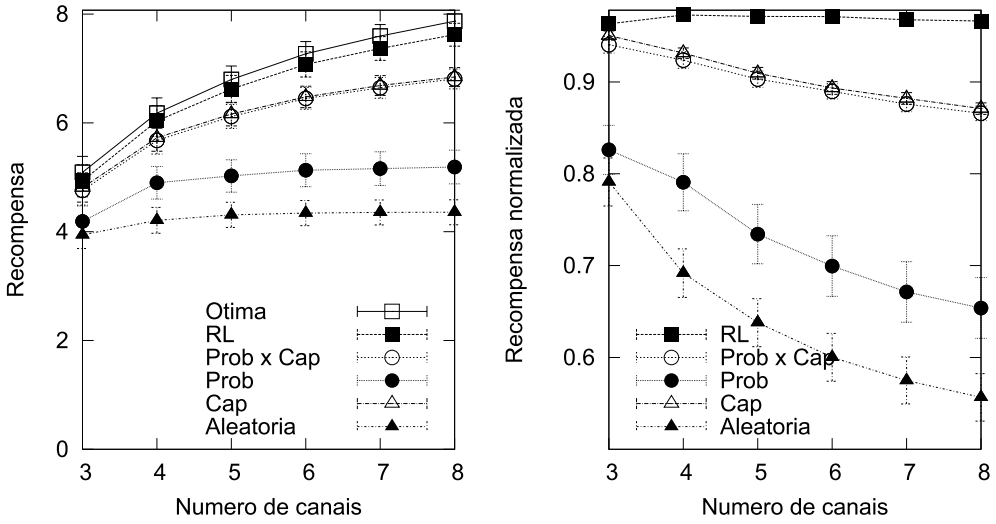
No início de cada rodada de simulação, sorteamos o valor da capacidade média e as probabilidades de disponibilidade de cada canal i, $i \in \{1, ..., N\}$. A capacidade média é sorteada seguindo uma distribuição uniforme dentro do intervalo $[FHC * CAPMAX, CAPMAX]$, onde FHC é o *fator de homogeneidade* dos canais e $CAPMAX$ é a *capacidade média máxima* dos canais. O parâmetro FHC deve assumir valores no intervalo [0,1]. Quanto maior for esse valor, maior é a homogeneidade das capacidades médias dos canais e mais próximas de $CAPMAX$. A probabilidade de disponibilidade de cada canal é sorteada uniformemente dentro do intervalo [0,1].

Numa rodada de simulação, a cada *slot* T, o estado do canal (livre ou ocupado) é sorteado de acordo com a sua probabilidade de disponibilidade utilizando-se uma distribuição uniforme. Além disso, a capacidade instantânea de cada canal é sorteada utilizando-se uma distribuição uniforme dentro do intervalo $[CAPMEDIA * (1 - FVA/2), CAPMEDIA * (1 + FVA/2)]$, onde FVA é o *fator de variabilidade* do ambiente e $CAPMEDIA$ é a *capacidade média* de cada canal. Quanto maior for o valor de FVA, a capacidade instantânea do canal apresenta uma alta variação. O parâmetro FVA deve assumir valores no intervalo [1,2].

A cada *slot* T, o simulador calcula a recompensa obtida por cada uma das sequências implementadas usando-se os mesmos estados e as mesmas capacidades instantâneas dos canais (critério de justiça). A recompensa em cada *slot* corresponde à taxa de transmissão efetiva (Seção 2). Um rodada de simulação consiste na execução de X *slots*. Ao final de cada rodada, o simulador fornece a recompensa média obtida por cada uma das sequências em todos os X *slots*.

4.2. Resultados

Foram realizadas simulações com 50.000 *slots* cada, utilizando-se um número de canais variando de 3 a 8. O fator de exploração ε foi configurado em 0.7 dentro do período de aprendizado, correspondente a 20% do número total de *slots*, passando para 0.0 ao final desse período. O parâmetro α do `RL` foi configurado em 0.1. O tamanho do *slot* T é um múltiplo inteiro do tempo necessário para sensorear um canal (τ). Foram feitas 200 rodadas de simulação para cada conjunto de parâmetros. Em todos os resultados, apresentamos a média das recompensas coletadas a cada rodada, com barras de erro correspondentes ao intervalo de confiança de 95%.



(a) recompensas absolutas (b) recompensas normalizadas

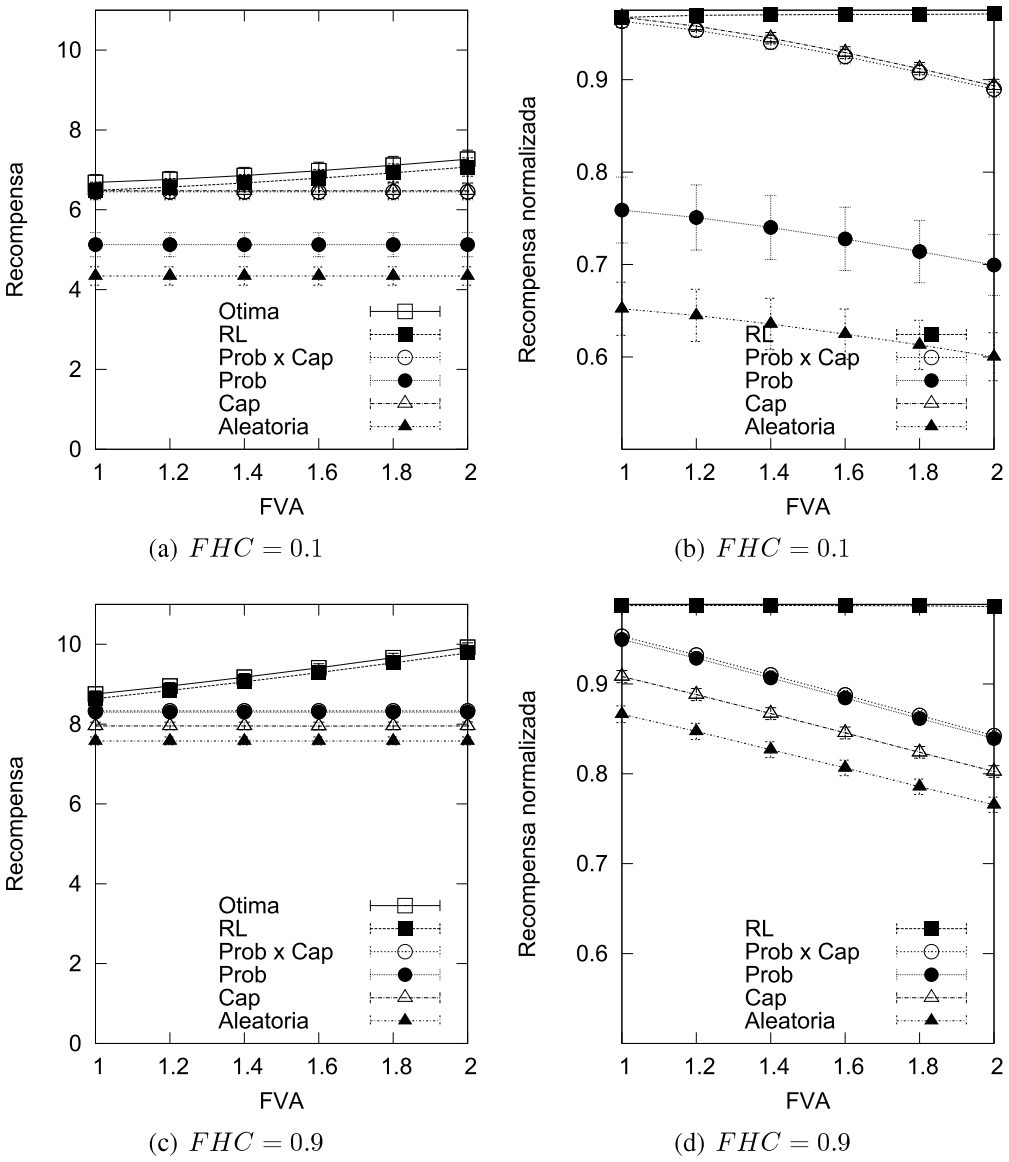
Figura 2. Resultados $FHC = 0.1$, $FVA = 2$ **e** $\delta = 0.95$**.**

Na primeira etapa de simulações, a Figura 2 apresenta resultados da variação do número de canais de 3 até 8. Nestes resultados, $CAPMAX$ e o tamanho do *slot* foram configurados em 10, e os parâmetros FHC e FVA em 0.1 e 2.0, respectivamente.

Analisando os resultados absolutos apresentados na Figura 2(a), percebe-se que o aumento do número de canais causa um aumento da recompensa média em todas as sequências simuladas. Este comportamento ocorre, pois o aumento do número de canais aumenta a possibilidade de existir um canal com alta capacidade média e grande probabilidade de disponibilidade. Também por este motivo, as sequências dadas por `Prob`, `Cap`, `ProbxCap`, `RL` e `Ótima`, que são conscientes das capacidades de canal e das probabilidades de disponibilidade, obtém um desempenho melhor do que o da sequência `Aleatória`.

A Figura 2(b) apresenta os resultados normalizados pelo desempenho obtido pela sequência `Ótima`. A comparação do desempenho das sequências neste gráfico mostra que a nossa proposta, `RL`, é a única que alcança resultados próximos ao valor ótimo. O desempenho das outras sequências é desfavorável, pois nenhuma delas utiliza regras de parada baseadas na predição do desempenho estimado de se continuar sensoreando os próximos canais da sequência, ou seja, nestas outras soluções o primeiro canal sensoreado como livre sempre é utilizado. Desta forma, o `RL`, que utiliza as experiências passadas armazenadas na *Q-table*, consegue determinar de maneira eficiente se é vantajoso utilizar um determinado canal sensoreado como livre. Outra observação interessante a respeito das curvas da Figura 2(b) é que o desempenho da sequência `Prob` é inferior ao desempenho das sequências `Cap` e `ProbxCap`. Isso indica que neste cenário a diferenciação entre as capacidades médias dos canais ($CAPMEDIA$) é mais importante do que a diferenciação entre suas probabilidades de disponibilidade. Com isso, é melhor ordenar os canais pela ordem decrescente de suas capacidades médias, pois aumenta-se a probabilidade de o primeiro canal sensoreado como livre ser um canal de maior capacidade.

Nas segunda etapa de simulações, variamos os parâmetros FHC e FVA, con-



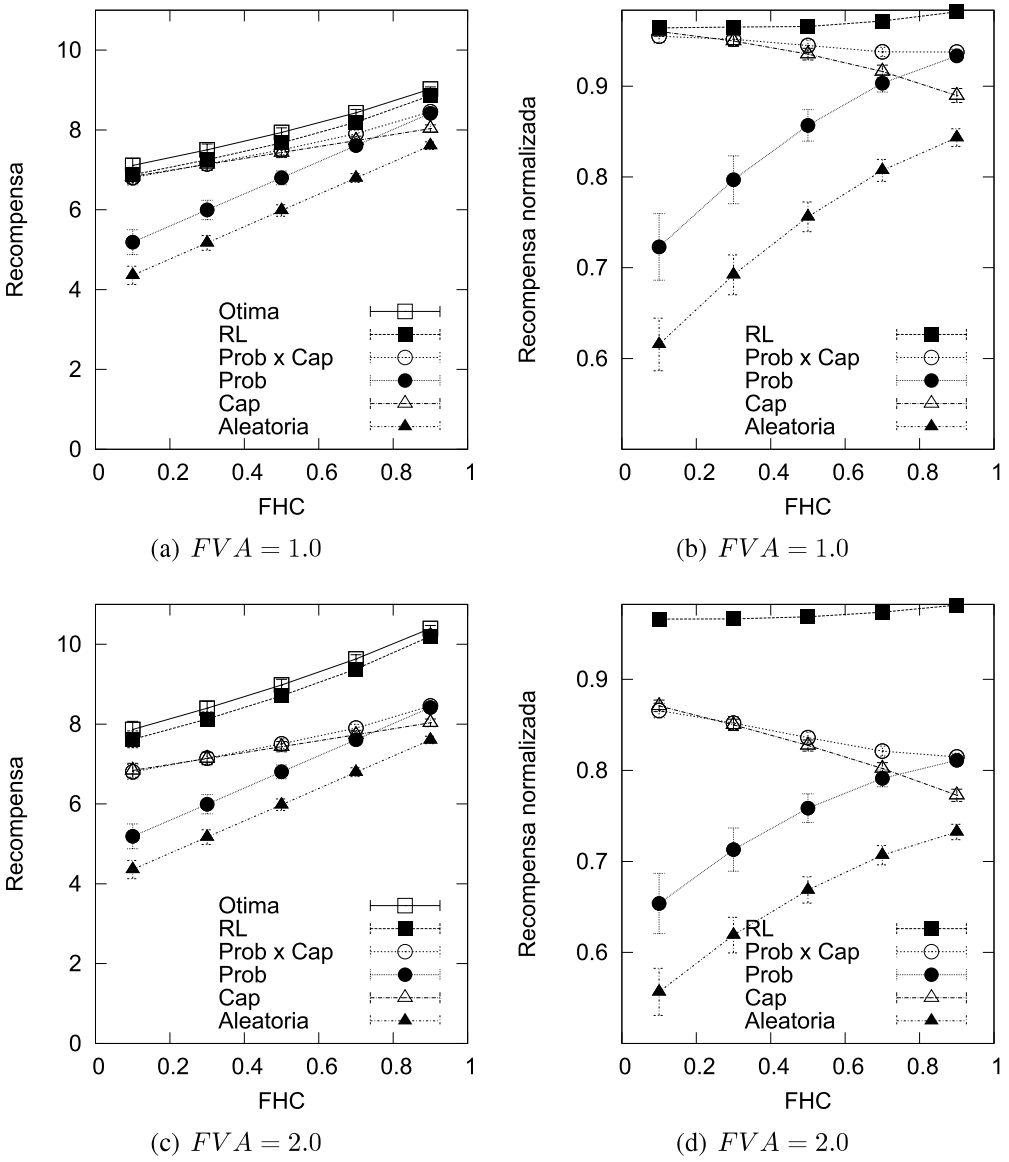
(a) $FHC = 0.1$ (b) $FHC = 0.1$

(c) $FHC = 0.9$ (d) $FHC = 0.9$

Figura 3. Resultados para 6 canais, com tamanho do slot = 10, capacidade = 10.

forme descrito na Seção 4, tamanho do *slot* e capacidade máxima iguais a 10, e comparamos as recompensas absolutas e normalizadas pelo valor da recompensa obtida pela sequência ótima.

O FVA modifica a variabilidade da capacidade instantânea dos canais em torno da capacidade média, que varia a cada *slot*, e é representativa da variação dinâmica da SNR. As estratégias `Prob`, `Cap`, `ProbxCap` e `Aleatória` são invariantes com relação a esse parâmetro, pois elas utilizam as médias das variáveis aleatórias para a geração das suas respectivas sequências. Assim, observando-se as Figuras 3(a) e 3(c), que consideram a recompensa absoluta, e as Figuras 3(b) e 3(b), que consideram a recompensa normalizada, para todos os valores de FVA, as sequências utilizadas serão sempre as mesmas, e as suas recompensas tendem para um valor médio constante. Para as estratégias `FB` e `RL`, o aumento da recompensa absoluta se deve ao fato dessas estratégias apenas utiliza-



(a) $FVA = 1.0$ (b) $FVA = 1.0$

(c) $FVA = 2.0$ (d) $FVA = 2.0$

Figura 4. Resultados para 6 canais, com tamanho do slot = 10, capacidade = 10.

rem um canal livre caso a recompensa instantânea do uso desse canal for maior do que a recompensa esperada do resto da sequência. Assim, elas tendem a utilizar canais livres com grandes capacidades instantâneas, devido a sua maior variabilidade. As demais estratégias sempre utilizam um canal quando ele é encontrado livre, independente da sua capacidade instantânea.

O FHC modifica a homogeneidade dos canais com relação às suas capacidades médias. Com o aumento do valor desse parâmetro, as capacidades médias ficam mais próximas da $CAPMAX$, aumentando a recompensa com o aumento do FHC para todos os mecanismos, conforme a Figura 4. No entanto, os mecanismos baseados em ordenação por capacidade, i.e. `Cap` e `ProbxCap`, não crescem na mesma proporção que os demais. A explicação para isso está no fato de que ao se homogeneizar os canais, o peso da capacidade na escolha da sequência se torna menos importante. Por isso, a estratégia `Prob`

melhora a medida que os canais se tornam mais homogêneos.

5. Conclusões

O sensoreamento do espectro é uma tarefa crítica para o uso oportunista dos canais do espectro de frequências licenciado. Especialmente em cenários onde os usuários secundários possuem apenas um único transceptor, que deve sensorear um canal por vez a fim de detectar oportunidades de uso. Nestes casos, a ordem utilizada para sensorear os canais pode ter grande impacto no desempenho. A teoria da parada ótima (*optimal stopping*) pode ser utilizada para modelar o problema e determinar a sequência de sensoreamento ótima que maximiza o desempenho. Entretanto, esta teoria assume conhecimento prévio a respeito das probabilidades de disponibilidade e da capacidade média esperada de cada canal.

Neste trabalho, propomos uma solução de baixa complexidade que utiliza uma máquina de aprendizado por reforço (*Reinforcement Learning*). Essa solução não requer o conhecimento prévio das probabilidades de disponibilidade e da capacidade média esperada em cada canal, podendo se adaptar dinamicamente às variações dessas características. Além disso, essa solução possui complexidade computacional baixa, o que a torna atrativa para ser embarcada em rádios cognitivos. Para avaliar o desempenho do mecanismo proposto, desenvolvemos um simulador que emula o funcionamento de um usuário secundário utilizando sequências de sensoreamento arbitrárias. Os resultados das simulações mostram que o mecanismo proposto obtém desempenho próximo ao da sequência de sensoreamento ótima e superior aos outros tipos de ordenamento que foram avaliados.

Como trabalhos futuros, pretendemos estender a avaliação para o caso de múltiplos usuários e para cenários onde as probabilidades de disponibilidade e as capacidades médias dos canais variam durante as simulações. Além disso, também seria interessante avaliar como possíveis erros no sensoreamento podem afetar o mecanismo proposto.

Referências

Chow, Y. S., Robbins, H., and Siegmund, D. (1971). *Great Expectations: The Theory of Optimal Stopping*. Houghton Mifflin Company.

FCC (2003). FCC-03-322 - NOTICE OF PROPOSED RULE MAKING AND ORDER. Technical report, Federal Communications Commission.

H. Jiang, L. Lai, R. Fan, and H. V. Poor (2009). Optimal Selection of Channel Sensing Order in Cognitive Radio. *IEEE Transactions in Wireless Communications*.

H. Kim and K. G. Shin (2008). Fast Discovery of Spectrum Opportunities in Cognitive Radio Networks. In *IEEE DySPAN*.

Han Han, Jin-long Wang, Qi-hui Wu, and Yu-zhen Huang (2010). Optimal Wideband Spectrum Sensing Order Based on Decision-making Tree in Cognitive Radio. *International Conference on Wireless Communications and Signal Processing (WCSP)*.

Ho Ting Cheng and Weihua Zhuang (2011). Simple Channel Sensing Order in Cognitive Radio Networks. *IEEE Journal on Selected Areas in Communications*.

J. Jia, Q. Zhang, and X. Shen (2008). HC-MAC: a Hardware Constrained Cognitive MAC for Efficient Spectrum Management. *IEEE Journal on Selected Areas in Communications*.

J. Mitola III and G. Q. Maguire Jr. (1999). Cognitive Radio: Making Software Radio more Personal. *IEEE Personal Communications*, 6(4):13–18.

Jelle R. Kok and Nikos Vlassis (2006). Collaborative Multiagent Reinforcement Learning by Payoff Propagation. *J. Mach. Learn. Res.*, 7:1789–1828.

John Ousterhout (1988). Tcl - Tool Command Language. http://www.stanford.edu/ ouster/cgi-bin/tclHistory.php.

Kok-Lim Alvin Yau, Peter Komisarczuk, and Paul D. Teal (2010a). Applications of Reinforcement Learning to Cognitive Radio Networks. In *IEEE International Conference in Communications (ICC)*.

Kok-Lim Alvin Yau, Peter Komisarczuk, and Paul D. Teal (2010b). Enhancing Network Performance in Distributed Cognitive Radio Networks using Single-agent and Multi-agent Reinforcement Learning. In *IEEE Conference on Local Computer Networks (LCN)*.

M. A. McHenry (2005). NSF Spectrum Occupancy Measurements Project Summary. Technical report, Shared Spectrum Company report.

S. Guha, K. Munagala, and S. Sarkar (2006). Approximation Schemes for Information Acquisition and Exploitation in Multichannel Wireless Networks. In *44th. Allerton Conference*.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: an Introduction*. MIT Press.