Redes de Rádios Cognitivos Utilizando Sequências de Saltos de Canais Baseadas em Papéis*

Raphael Melo Guedes[1], Marcel William Rocha da Silva[2], Pedro Smith Coutinho[1], José Ferreira de Rezende[1]

[1]COPPE – Universidade Federal do Rio de Janeiro (UFRJ)

[2]DTL - IM – Universidade Federal Rural do Rio de Janeiro (UFRRJ)

{raphael,marcel,coutinho,rezende}@land.ufrj.br

Abstract. *In a multi-channel scenario, where nodes hop periodically among channels without coordination, the rendezvous is the first step in establishing a communication. Several studies that deal with this problem do not deal with broadcast transmissions, which are very useful in sending control messages. An intuitive solution consists of all nodes converge to a unique and synchronized sequence. However, this unique sequence is inefficient for sending data messages in unicast, since it requires negotiation packet-by-packet for selecting the transmission channel of the data message thus the capacity of multiple channels would be used. In this work we propose a role-based policy that defines the on-demand use of different sequences, alleviating the media access problem and enabling the occurrence of parallel transmissions on different channels. This proposal has been implemented in the ns-2 in a cognitive radios mesh network scenario, where the presence of primary users favors the use of channel hopping.*

Resumo. *Em um cenário multi-canal, onde os nós saltam periodicamente entre canais sem coordenação, o encontro entre os nós (rendezvous) é a primeira etapa no estabelecimento de uma comunicação. A maioria dos trabalhos que lidam com este problema não tratam de transmissões em broadcast, as quais são bastante úteis no envio de mensagens de controle. Uma solução intuitiva consiste em todos os nós convergirem para uma sequência única e sincronizada. No entanto, esta sequência única é ineficiente para o envio de mensagens de dados em unicast, pois requer a negociação pacote-a-pacote para a escolha do canal de transmissão da mensagem de dados a fim de que a capacidade dos múltiplos canais seja utilizada. Assim, neste trabalho propomos uma política de papéis que define o uso de diferentes sequências sob demanda, aliviando o problema de acesso ao meio e possibilitando a ocorrência de transmissões em paralelo em diferentes canais. Esta proposta foi implementada no ns-2 em um cenário de redes em malha de rádios cognitivos, onde a presença de usuários primários favorece ainda mais o uso do mecanismo de saltos de canais.*

1. Introdução

Devido à capacidade de acesso dinâmico ao espectro (*Dynamic Spectrum Access* - DSA) dos rádios cognitivos, eles representam uma solução promissora ao problema de escassez

*Este trabalho recebeu recursos do CNPq, CAPES, FAPERJ, FINEP e RNP.

e subutilização do espectro. Esses rádios, ou usuários secundários (USs), possuem permissão para acessar faixas do espectro de frequências licenciadas (*i.e.*, canais) durante os períodos em que os usuários licenciados, ou usuários primários (UPs), daquela faixa não a utilizam [Akyildiz et al. 2006]. Portanto, os USs devem suspender sua operação e migrar para outro canal disponível sempre que um UP estiver presente. Neste cenário multicanal, o encontro entre pares de nós num determinado canal (*rendezvous*) é o primeiro passo para o início de uma comunicação [Theis et al. 2011].

O *rendezvous* pode ser demorado quando não se possui nenhuma informação prévia de quais ou qual canal está livre para o uso. Assim, muitos trabalhos em DSA recorrem ao uso de um canal de controle exclusivo para troca de mensagens de coordenação entre os USs cuja finalidade é facilitar a ocorrência de *rendezvous*. Porém, devido a disponibilidade dinâmica dos canais, qualquer canal selecionado como canal de controle pode estar sendo compartilhado com usuários UPs, podendo permanecer inacessível durante longos períodos. Ou seja, um único canal de controle não é o mais aconselhável em redes DSA. Logo, diversas propostas adotam a ideia de um canal de controle que usa saltos de canais (*channel hopping*), na intenção de tornar-se imune à seleção de um único canal e garantir um encontro mais rápido entre os nós, visto que estes saltam por um conjunto de sequências de canais (*rendezvous channels*) aumentando as chances de um encontro.

Diversos algoritmos foram propostos na literatura para a geração de sequências aleatórias de canais que permitem garantir um tempo máximo para o encontro (*Maximum-Time-To-Rendezvous* - MTTR) entre dois nós. Esses algoritmos são comumente referenciados como algoritmos de *rendezvous* (RV). Formalmente, a sequência de saltos de canais de cada nó é estabelecida na forma (*slot*, canal) e, quando ocorre um casamento entre *slot* e canal de ambos os nós, eles podem então estabelecer uma comunicação e trocar informações, sejam dados ou controle.

Porém, um problema encontrado em grande parte dos algoritmos de RV é o fato deles não lidarem diretamente com transmissões *broadcast*. Assim, mensagens de controle que devem ser enviadas por um nó a todos os seu vizinhos, tais como mensagens de *hello*, RTS/CTS, informações de estado de enlace, entre outras, exigem a ocorrência de múltiplos RVs nó-a-nó. Estes múltiplos RVs *unicast* atrasam a entrega da mensagem a todos os nós e podem prejudicar o estabelecimento de alguma comunicação futura.

Para este problema, uma solução intuitiva seria todos os nós adotarem uma sequência única como canal de controle. Neste caso, uma transmissão *broadcast* ocorreria de forma natural. No entanto, ao se adotar esta solução, os nós passariam a negociar, neste canal de controle, o canal (ou a sequência de canais) a ser utilizado(a) nas futuras transmissões *unicast*. No caso do nó ser um elemento intermediário que encaminha pacotes para diferentes próximos saltos, essa negociação teria que ser feita praticamente pacote-a-pacote, elevando o número de requisições de acesso ao canal de controle e provocando um problema conhecido como gargalo do canal de controle [So and Vaidya 2004, Mo et al. 2005, Luo et al. 2006].

A solução proposta neste trabalho utiliza a ideia dos nós assumirem diferentes papéis no decorrer do tempo, os quais são associados ao uso de diferentes sequências de saltos de canais. Conforme o estado do nó e a necessidade de uso dos canais para transmissão, ele chaveia entre diferentes sequências de saltos de canais. Assim, quando o nó

deseja enviar uma mensagem de controle em *broadcast*, ele chaveia de forma assíncrona para uma sequência de saltos computada localmente e que permite a ele atingir o maior número de vizinhos, evitando os canais com maior probabilidade de uso por UPs. Quando o nó estiver em repouso, ele permanece na sequência definida pelo algoritmo de RV utilizado. Finalmente, para o envio de mensagens em *unicast*, o nó transmissor deve chavear para a sequência de repouso do nó receptor.

O problema da transmissão em *broadcast* pode ser resolvido conforme proposto em [Guedes et al. 2012], onde seleciona-se de maneira gulosa os canais utilizados de forma a relacionar o conhecimento das sequências dos nós vizinhos e a estimativa de uso do canal pelos nós UPs. Adotando esta prática, de mudança de sequências conforme a necessidade, diminui-se a competição pelo acesso ao meio além de favorecer possíveis transmissões em paralelo, inclusive entre nós vizinhos. Além disso, não gera a necessidade de negociação prévia para uma transmissão *unicast*, pois o nó TX deve simplesmente chavear para a sequência de repouso do nó RX.

Neste artigo, a solução proposta e a solução intuitiva foram implementadas no simulador *ns-2*, onde foi necessário incluir: um modelo de interferência físico baseado na SINR (*Signal-to-Interference-plus-Noise Ratio*) [Brar et al. 2006], um novo agente gerador de tráfego em *broadcast*, a possibilidade dos nós saltarem de canal segundo diferentes sequências de saltos especificadas, os algoritmos de *rendezvous* para a geração dessas sequências, o algoritmo proposto em [Guedes et al. 2012] para a geração da sequência de *broadcast*, e a coleta das métricas de desempenho para a comparação das duas soluções.

A Seção 2 descreve os principais trabalhos que tratam do problema de *rendezvous*. A Seção 3 apresenta o modelo do sistema sob qual reside a proposta deste artigo. A Seção 4 descreve o mecanismo proposto neste trabalho e a Seção 5 descreve o ambiente de simulação usado na avaliação de desempenho da solução proposta e de uma solução intuitiva para o problema tratado. Em seguida, a Seção 6 traz os resultados das simulações realizadas, assim como uma discussão a respeito. E, por fim, a Seção 7 apresenta as conclusões deste trabalho.

2. Trabalhos Relacionados

Diversos trabalhos em rádios cognitivos (RCs) adotam a técnica de *channel hopping* precisamente para evitar os canais ocupados pelos UPs, permitindo o encontro desses RCs em outros canais. Além disso, essa abordagem permite a comunicação simultânea de múltiplos RCs na mesma área de interferência, através da atribuição de diferentes sequências de saltos de canais para cada nó. Vários trabalhos na literatura tratam o problema de *rendezvous*, cuja diferença entre eles está na utilização de diferentes estratégias na construção de suas sequências de salto de canais [Silvius et al. 2008, Bahl et al. 2004, DaSilva and Guerreiro 2008, Bian et al. 2011, Cormio and Chowdhury 2010, Theis et al. 2011, Lin et al. 2011, Zhang et al. 2011, Guedes et al. 2012]. Esses trabalhos são avaliados e comparados de acordo com as métricas de tempo máximo necessário para que dois nós se encontrem (MTTR - *Maximum-Time-To-Rendezvous*) e a distribuição dos encontros nos diferentes canais da sequência de saltos. Quanto menor é o MTTR e quanto mais distribuídos são os encontros nos diferentes canais, melhor é o algoritmo.

A maior parte dos algoritmos de *rendezvous* (RV) existentes assume um sistema

de divisão do tempo em *slots*, em que a cada *slot* os RCs saltam entre os canais na tentativa de encontrar seus potenciais vizinhos. Em um esquema onde cada RC salta entre canais aleatoriamente [Silvius et al. 2008], quando dois RCs saltam para o mesmo canal no mesmo instante, um encontro (RV) ocorre, caso o canal esteja livre da presença de algum UP. Caso contrário, os nós devem continuar "saltando" até que ocorra o RV.

Algoritmos de RV podem ser classificados como síncronos ou assíncronos, dependendo da simultaneidade no início das suas sequências de saltos. Em sistemas síncronos, uma coordenação inicial entre os usuários é necessária antes do *rendezvous* para estabelecer o sincronismo entre os *clocks* dos RCs [DaSilva and Guerreiro 2008, Bian et al. 2011, Bahl et al. 2004, Guedes et al. 2012]. Desta forma, as trocas de canal e os *slots* ficam consequentemente alinhados no tempo.

Um exemplo de algoritmo de RV é o *modular clock* (*mclock*) [Theis et al. 2011]. Este algoritmo usa aritmética modular e números primos para garantir um MTTR que seja satisfatório. O algoritmo começa pela seleção de um canal de forma aleatória, a partir da lista de canais disponíveis. Após cada intervalo de tempo (*slot*), o CR salta $r_i \ mod \ p_i$ canais percorrendo esta lista, onde p_i é o menor número primo maior que o número de canais. Após $2 \times p_i$ *slots*, um novo valor de r_i é escolhido aleatoriamente dentro do intervalo $[0, p_i)$ [Guedes et al. 2012].

Em [Bian et al. 2011], é apresentado um esquema baseado em sistema de *quoruns* para construir a sequência de salto de canais. O objetivo é aumentar o número de sobreposições entre as múltiplas sequências. Um *quorum* é definido como um elemento de um sistema S que satisfaz a propriedade de interseção: $p \cap q \neq \emptyset$, $\forall p, q \in S$ [Lo 2011]. Por exemplo, a partir de $Z = \{0, 1, 2\}$ pode-se formar o conjunto de *quoruns* $S = \{\{0, 1\}, \{0, 2\}, \{1, 2\}\}$ e cada elemento de S dá origem a uma sequência, conforme a Figura 1. Assim temos u, v e w que são sequências que apresentam sobreposição de canais entre elas e, cada sequência pode ser atribuída a um nó como uma sequência de salto de canais. Na figura, os campos em branco representam o caso em que o canal será atribuído aleatoriamente.

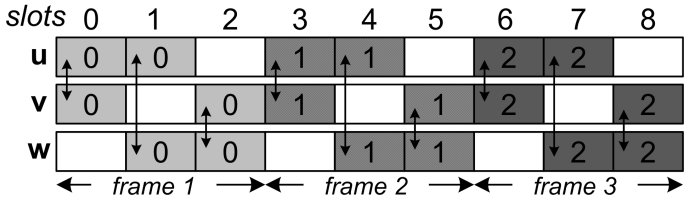


Figura 1. Exemplo de sequências a partir do conjunto de *quoruns*.

Outro algoritmo de RV é o ETCH (*Efficient Channel Hopping for Communication Rendezvous*)[Zhang et al. 2011]. Este algoritmo consiste de três partes: escalonamento, atribuição de canais e execução da sequência. Para conseguir uma melhor média de TTR, o ETCH constrói um conjunto de $2N$ sequências, contendo $2N - 1$ *slots*, onde N é o número de canais, de forma que cada sequência apresente sobreposição de canais com as demais em diferentes *slots*. Para isso, as sequências são construídas em dois passos, no primeiro passo, ele gera $2N - 1$ escalonamentos e, no segundo passo, é feita a atribuição de canais. Ao fim do processo, todos os nós dispõem de um conjunto

de sequências iguais. Na execução, primeiramente, o nó seleciona aleatoriamente uma sequência que irá seguir, depois de passar por todos os *slots* desta sequência, ele realiza uma seleção aleatória entre as sequências restantes, e passa a utilizar esta sequência recém escolhida [Zhang et al. 2011].

Um problema, mencionado em [Theis et al. 2011], é que grande parte dos algoritmos de RV não aborda a questão de transmissões em *broadcast*, ou melhor, o caso do *rendezvous* entre múltiplos usuários. Uma solução pode ser encontrada em [Lin et al. 2011], onde inicialmente cada nó possui uma sequência de canais, gerada com parâmetros próprios do algoritmo. A cada encontro dois-a-dois entre os nós, eles comparam seus parâmetros e um dos nós passa a utilizar a sequência do outro. Para esta comparação, a solução define um conjunto de critérios de desempate dois-a-dois entre os parâmetros. Desta forma, ao fim de alguns encontros, os nós convergem para uma sequência única, e assim, podem enviar mensagens em *broadcast* usando essa sequência. Uma dificuldade é que o tempo de convergência pode ser grande, uma vez que a ordem em que os encontros ocorrem influencia no processo de desempate. Além disso, o uso de uma única sequência cria o problema de múltiplos acessos ao meio compartilhado.

Algoritmo 1: Seleção de sequência de *Broadcast*.

input: $RADIOS$, $nó - broadcaster$, $SEQSIZE$, $CANAIS$, $SEQ(canal, slot)$, $P_{idle}(nó, canal)$
init $sequence = \{\ \}$
init $reachedNodes = \emptyset$
for *(nó in* $RADIOS$*)* **do**
  init $P_{idle}(nó)acumulada = 0.0$
while $(|sequence| < SEQSIZE)$ **do**
  selecione canal C onde $nó - broadcaster$ alcança o maior número esperado de vizinhos $\notin reachedNodes$
  for *(nó in vizinhos alcançados)* **do**
    $P_{idle}(nó)acumulada = P_{idle}(nó)acumulada + P_{idle}(nó, C)$
    if $(P_{idle}(nó)acumulada >= 1.0)$ **then**
      adicione $nó$ a $reachedNodes$
  acrescente C a $sequence$
  if $(reachedNodes == RADIOS)$ **then**
    $reachedNodes = \emptyset$
retorna $sequence$

Para evitar o problema trazido pelo uso da sequência única, em [Guedes et al. 2012], propomos uma solução que consiste em definir uma sequência especial, específica para cada nó, para a transmissão de pacotes em *broadcast*. Esta sequência pode ser utilizada em conjunto com qualquer algoritmo de RV e tem como objetivo permitir o encontro com o maior número de vizinhos em poucos *slots*. A proposta considera ainda estimativas de uso dos canais pelos UPs para evitar canais pouco disponíveis. O Algoritmo 1 proposto para a construção da sequência de *broadcast* utiliza como entradas as sequências dos vizinhos, que podem ser recriadas depois da troca de parâmetros do algoritmo de RV em eventuais encontros, e também a estimativa

de utilização dos canais. Esta estimativa da atividade de cada canal, mais precisamente a probabilidade de um canal estar livre da atividade de usuários primários, é indicada como P_{idle} no algoritmo. No trabalho, comprovou-se através de simulações a eficiência do uso dessa sequência, num cenário de único salto, em relação ao uso de múltiplos *rendezvous* dois-a-dois.

Neste artigo, utilizamos a ideia do nó dispor de mais de uma sequência e, a partir de um esquema de papéis, decidir qual sequência utilizar. Assim, o modelo adotado neste trabalho assume uma topologia de múltiplos saltos onde os nós enviam pacotes de dados em *unicast* e de controle em *broadcast*. Maiores detalhes sobre o modelo e a proposta do uso de papéis serão apresentados respectivamente nas duas próximas seções.

3. Modelo do Sistema

No modelo de rede de rádios cognitivos adotado neste trabalho, consideramos um cenário com N_s usuários secundários (rádios cognitivos) e N_p usuários primários (rádios licenciados), ambos utilizando uma faixa do espectro licenciado composta por C canais. Os usuários secundários acessam os C canais utilizando técnicas de salto de canais, onde o tempo é dividido em *slots* de duração τ. Durante cada *slot*, um único canal c_i ($i = 1, 2, ..., N$) é utilizado pelo usuário secundário, o qual mantém sua única interface de rádio sintonizada neste canal. A duração τ de um *slot* é dada pelo tempo necessário para a transmissão de um pacote de dados seguido pela mensagem de confirmação (*Ack*). As mensagens de dados enviadas pelos usuários secundários possuem tamanho fixo.

Neste modelo, assumimos que os UPs apenas mudam de estado nas transições entre os *slots* e não trocam de canal durante sua operação. Assim, modelamos o padrão de atividade de cada canal $c_i \in C$ de acordo com um modelo de transição de dois estados, como apresentado na Figura 2. Os estados `ON` e `OFF` representam respectivamente os períodos de tempo em que o canal está ocupado (UPs em atividade) ou livre (UPs ociosos). Este modelo `ON-OFF` é amplamente adotado na literatura e é uma boa aproximação para muitos cenários e tecnologias legadas [Wellens et al. 2009].

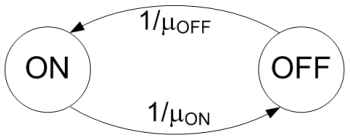


Figura 2. Modelo de atividade nos canais.

Os canais utilizados em cada *slot* de tempo pelo usuário secundário i são representados pela sequência de saltos de canal $S_i = s_i(1), s_i(2), ..., s_i(M)$, onde M é o tamanho da sequência. Assumimos que os USs estão sincronizados, ou seja, as trocas de canal são simultâneas e as sequências de saltos estão alinhadas. Quando o canal c_k usado na posição t da sequência de saltos de dois USs i e j é o mesmo ($s_i(t) = s_j(t) = c_k$), a comunicação entre estes USs se torna possível desde que não haja nenhum primário ativo naquele mesmo canal. De maneira genérica, n USs podem estar sintonizados no canal c_k no *slot* t ($s_1(t) = s_2(t) = ... = s_n(t) = c_k$).

A fim de evitar colisões de pacotes nos *slots*, assumimos que os USs utilizam um mecanismo de controle de acesso ao meio do tipo CSMA/CA (*Carrier Sense Multiple*

Access with Collision Avoidance). Assim, no início de cada *slot*, os USs que desejam transmitir devem escutar o meio livre por um intervalo de tempo escolhido aleatoriamente. Caso o meio permaneça livre durante todo o intervalo, a transmissão é realizada. Caso outra transmissão seja detectada durante este intervalo, o US adia a transmissão do pacote para o próximo *slot*, onde o transmissor novamente disputará o acesso ao meio.

No caso de transmissões de pacotes *unicast*, o US transmissor aguarda por um reconhecimento. Caso o reconhecimento não chegue, o US disputará novamente o meio nos próximos *slots* para realizar novas tentativas de retransmissão do pacote, até que um certo número de tentativas seja atingido. Neste caso, o pacote será descartado. Nenhum controle de erro é usado na transmissão de pacotes em *broadcast*. Em resumo, utilizando este modelo CSMA/CA em *slots*, os USs disputam o acesso aos C canais para a transmissão de pacotes em *unicast* e *broadcast*. Para a transmissão destes dois tipos de mensagem, os USs comutam entre papéis, de acordo com o funcionamento descrito na próxima seção.

4. Proposta

O uso do mecanismo de saltos de canais pelos USs consiste em todos os nós saltarem entre canais, evitando aqueles canais ocupados pelos UPs, seguindo uma sequência de saltos definida por um algoritmo de RV. Toda vez que ocorre um encontro entre dois nós, mensagens de controle ou de dados podem ser trocadas entre eles. Como explicado anteriormente, a maioria dos algoritmos de RV propostos na literatura não tratam do problema da transmissão simultânea para múltiplos usuários, denominada neste trabalho de transmissão em *broadcast*. Assim, quando um nó deseja enviar uma mensagem a todos os seus vizinhos, esta transmissão é feita nó-a-nó, a partir de sucessivos *rendezvous*. Este procedimento leva a grandes atrasos na disseminação da informação, prejudicando o funcionamento da rede.

Como mencionado na Seção 2, um solução para esse problema é apresentada em [Lin et al. 2011]. Nessa proposta, os nós convergem para uma sequência única, facilitando assim o envio de mensagens em *broadcast*. Essa solução, doravante denominada de solução intuitiva, tem como vantagem permitir o envio imediato da mensagem a todos os nós da vizinhança, aumentando a taxa de entrega dessas mensagens. No entanto, o uso dessa sequência única se aplicada a todas as transmissões *unicast* da rede provoca um problema de contenção no acesso ao meio em todos os canais da sequência, o que limita a capacidade da rede. Uma forma de se evitar esse problema seria os nós negociarem, usando o canal de controle fornecido pela sequência única, o canal ou sequência de canais a ser utilizado(a) para as transmissões *unicast* entre dois nós. No entanto, conforme a carga de mensagens *unicast*, a quantidade de negociações no canal de controle pode ser grande, gerando o problema conhecido como gargalo do canal de controle. No caso de um nó com função de roteador, negociações podem ser necessárias pacote-a-pacote.

A proposta neste artigo consiste na definição de uma política de papéis/estado, no qual cada nó mapeia o uso de uma sequência diferente em decorrência do seu objetivo. Desta forma, definimos três possíveis estados para o nó: repouso ou modo de recepção (RX), TX *unicast* e TX *broadcast*. De acordo com o estado do nó, ele decide por uma sequência mais apropriada que evite ou cause menos competição no meio. Assim, o nó utiliza a sequência denominada sequência de repouso quando não possui nada a enviar, ou seja, em modo de repouso. A sequência de repouso é aquela gerada pelo algoritmo de

RV utilizado. Quando o nó deseja enviar uma mensagem de *broadcast*, ele utiliza a sua sequência de *broadcast*, definida conforme [Guedes et al. 2012], e quando deseja enviar uma mensagem *unicast* o nó transmissor chaveia para a sequência de repouso do nó de destino.

A intenção desta estratégia baseada em papéis é aliviar o acesso ao meio e possibilitar transmissões em paralelo, através da atribuição de sequências distintas. Pois diferentemente do caso de uma única sequência, cada nó utilizará uma sequência de *broadcast* própria originada a partir de sua vizinhança. Outra vantagem é de não criar a necessidade de uma negociação prévia pacote-a-pacote para uma transmissão *unicast*, pois na necessidade de envio de uma mensagem o nó transmissor chaveia para a sequência de repouso do nó receptor desejado, sequência esta esperada e válida para o modo de recepção. Esta abordagem reduz a necessidade de troca de informações, mensagens de controle, entre os nós para um agendamento de alguma transmissão futura.

Nas duas próximas seções, apresentaremos o ambiente de simulação utilizado e os resultados numéricos de comparação do desempenho da nossa proposta e da solução intuitiva, respectivamente.

5. Ambiente de Simulação

O simulador utilizado para a avaliação e comparação das soluções foi o *ns-2* [NS-2], sendo necessário, para isto, implementar um conjunto de extensões a esse simulador. Primeiramente, um modelo de interferência físico baseado na SINR (*Signal-to-Interference-plus-Noise Ratio*) [Brar et al. 2006] foi implementado para ser usado com a camada MAC 802.11 padrão do simulador. Este modelo permite computar a interferência acumulada sofrida por uma transmissão, com a qual é possível calcular a SINR de um determinado quadro e, por conseguinte, a probabilidade dele ser recebido com sucesso.

Em seguida, a camada MAC 802.11 foi modificada para permitir que os nós trocassem de canal a cada *slot*, segundo diferentes sequências de saltos, e se comportassem de acordo com o modelo CSMA/CA em *slots*, como detalhado na Seção 3. Para a geração das sequências de saltos, diversos algoritmos de RV (*rendezvous*) foram implementados (*quorum*, ETCH, *mclock*, etc.), incluindo o algoritmo proposto em [Guedes et al. 2012] para a geração da sequência de *broadcast*. Implementou-se também um novo agente gerador de tráfego em *broadcast*, que gera pacotes periódicos com intervalos de tempo $\mu \pm jitter$. Os pacotes em *broadcast* gerados são recebidos pelo próprio agente, o qual coleta e armazena estatísticas por vizinho a respeito dos pacotes recebidos.

O cenário utilizado nos experimentos foi de redes em malha sem fio de múltiplos saltos. Para a geração das topologias foi utilizado um modelo de posicionamento que garante a conectividade entre os nós, evitando que o grau máximo entre todos os nós ultrapasse um determinado valor. Assim, seguindo esse modelo, um nó ao ser disposto aleatoriamente em uma determinada área, somente será mantido se estiver dentro da área de cobertura de pelo menos um outro nó e obedecer o grau máximo de vizinhança.

Os usuários primários presentes nos cenários simulados têm a atividade regida por um modelo `ON-OFF`, assim como descrito na Seção 3. O tempo médio de permanência nos estados `ON` e `OFF` eram dados por variáveis aleatórias exponenciais com médias μ_{ON} e μ_{OFF}, respectivamente. Os UPs simulados possuíam um raio de influência, denominado

R_I. Os USs dentro do raio de influência de um UP ficam impossibilitados de transmitir e receber pacotes naquele canal utilizado pelo UP enquanto o mesmo encontra-se no estado `ON`. Vale destacar que neste modelo, a probabilidade do canal estar livre para um US sobre a influência de um UP é dada por $p_{Idle} = {\mu_{OFF}}/{(\mu_{ON}+\mu_{OFF})}$.

Todos os nós da rede em malha sem fio geram tráfego *broadcast* de acordo com uma determinada carga. Vale ressaltar que esse tráfego é de apenas um salto, ou seja, ele é gerado a partir de um nó para todos os seus vizinhos, não sendo reencaminhado por esses nós. Da mesma forma, o tráfego *unicast* é gerado por todos os nós, tendo como destino um dos vizinhos de cada nó, o qual é escolhido aleatoriamente. Esse tráfego é do tipo CBR cuja carga pode ser controlada pela variação do intervalo entre a geração dos pacotes. Como todo o tráfego gerado é de apenas um único salto, os resultados não são influenciados pelo protocolo de roteamento utilizado, mas apenas pela carga imposta à rede, a qual é variada durante a avaliação de desempenho para ambos os tráfegos.

6. Resultados de Simulação

Em todas as simulações realizadas, 25 nós são dispostos em uma área quadrada de 1000 metros de lado, seguindo o modelo de posicionamento descrito na seção anterior. O grau máximo utilizado para a geração das topologias foi igual a 6. Ainda, um UP para cada canal disponível é disposto aleatoriamente dentro dessa área, o qual tem uma raio de influência R_I igual a 250 metros. O modelo `ON-OFF` de atividade dos UPs é parametrizado pela fixação do parâmetro μ_{OFF} em 20 *slots* e a posterior escolha aleatória da probabilidade de cada primário estar ocioso (p_{Idle}) dentro do intervalo $(0, 1)$. Com estes valores é possível calcular o parâmetro μ_{ON} de cada UP através da fórmula apresentada na seção anterior.

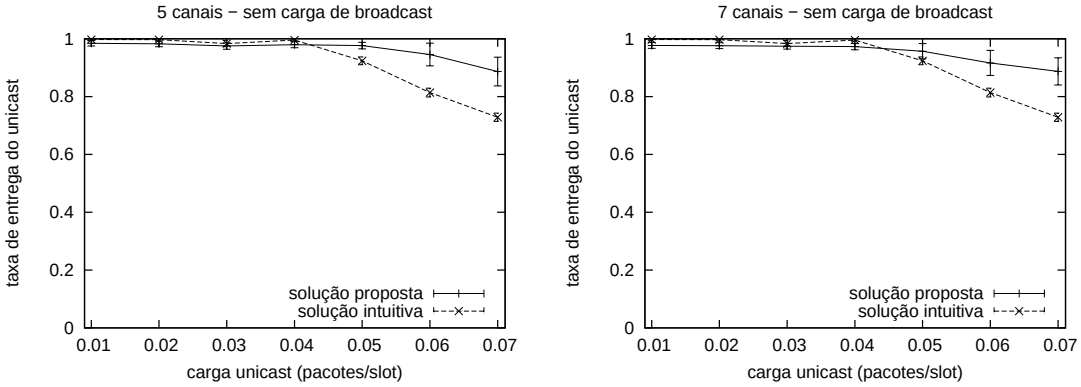
Cada experimento é executado por 50000 *slots* de tempo e as métricas avaliadas foram: a taxa de entrega do tráfego *unicast*, e a taxa de entrega e o atraso, em número de *slots*, do tráfego *broadcast*. Essas métricas de desempenho são mostradas em função da carga *unicast* gerada na rede e do número de canais. Cada valor mostrado nos gráficos equivale a média de 30 rodadas e as barras de erro correspondem ao intervalo de confiança calculado com um nível de confiança de 95%.

As sequências de saltos de canais utilizadas na solução proposta foram atribuídas a partir do algoritmo de RV *mclock* [Theis et al. 2011]. Obtivemos resultados também para o algoritmo ETCH que apresentou melhor taxa de entrega em *broadcast* que o *mclock*, porém a taxa de entrega *unicast* empatou com a solução intuitiva.

A carga de tráfego *unicast* é mostrada na abscissa de todos os gráficos em termos do número de pacotes por *slot*, o qual é variado entre 0.01 e 0.07. A carga do tráfego *broadcast*, μ, assume três valores distintos: 0 (sem carga), 500 (alta carga) e 1000 (média carga). Esses valores correspondem ao intervalo entre geração de pacotes *broadcast*, em número de *slots*. O valor do $jitter$ somado ao intervalo μ é sorteado no intervalo $[-0.05\mu, +0.05\mu]$.

Os gráficos da Figura 3 mostram a taxa de entrega do tráfego *unicast* para sequências de 5 e 7 canais quando nenhum tráfego *broadcast* é gerado. Os resultados demonstram que a solução proposta apresenta um desempenho superior com o aumento da carga, pois consegue aliviar o compartilhamento no meio pelo fato de cada nó utilizar diferentes sequências de canais a cada instante. Essas sequências correspondem às

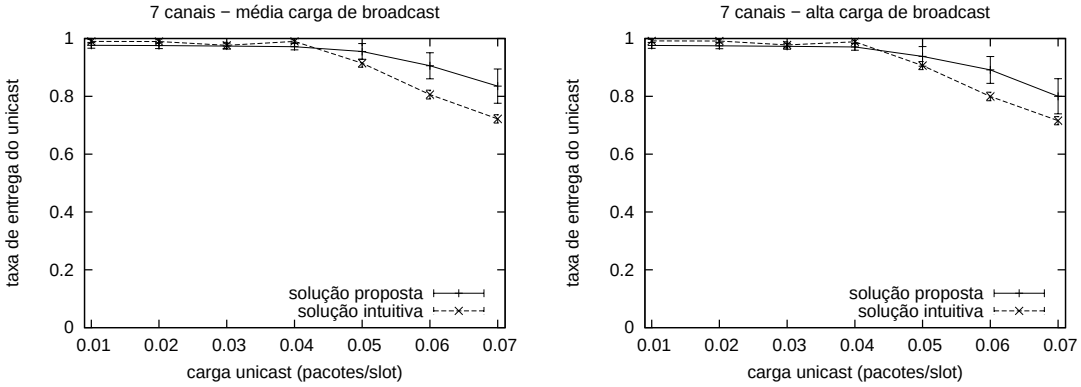
sequências de repouso de seus destinatários. Esse ganho ocorre mesmo quando simultaneamente o receptor chaveia de sequência por ser também um transmissor. O aumento do número de canais de 5 (Fig. 3(a)) para 7 (Fig. 3(b)) tem pouca influência nesse resultado.



(a) 5 canais (b) 7 canais

Figura 3. Taxa de Entrega do Tráfego *Unicast* Sem Carga de *Broadcast*

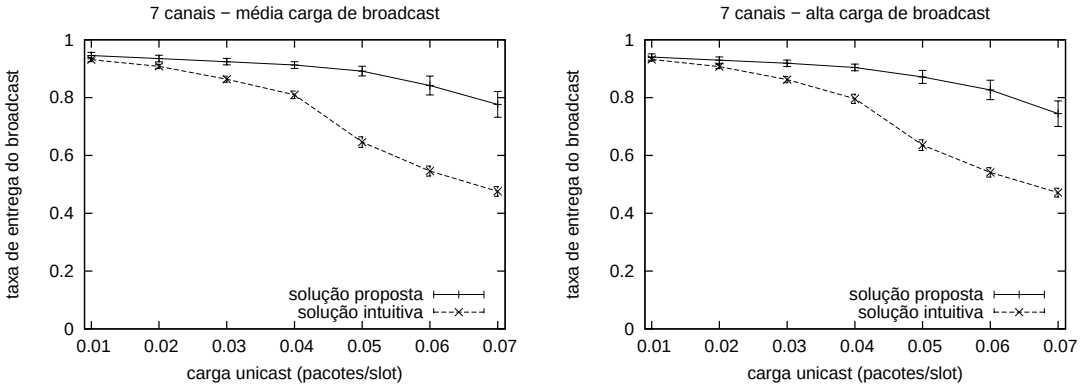
Nos próximos dois conjuntos de resultados (Figuras 4 e 5), a influência do tráfego *broadcast*, em conjunto com o tráfego *unicast*, é analisada. Com relação ao tráfego *unicast* (Figura 4), podemos notar que o acréscimo do tráfego *broadcast* tem um impacto reduzido na taxa de entrega do primeiro. No entanto, podemos ver que a solução proposta é a mais afetada. Isso se deve ao fato de uma única mensagem *broadcast* ter de ser retransmitida em todos os canais da sequência de *broadcast*, o que não acontece no caso da solução intuitiva. Esta característica também afeta a recepção de pacotes *unicast*, já que vizinhos de um US tem maior dificuldade de encontrá-lo em sua sequência de repouso quando ele está utilizando a sequência de *broadcast*. No entanto, de maneira geral a solução proposta continua sendo superior à solução intuitiva nessa métrica pelo fato de diminuir o compartilhamento do meio nos diversos canais, aumentando a capacidade da rede em escoar o tráfego gerado.



(a) 7 canais - média carga (b) 7 canais - alta carga

Figura 4. Taxa de Entrega do Tráfego *Unicast* com Carga de *Broadcast*

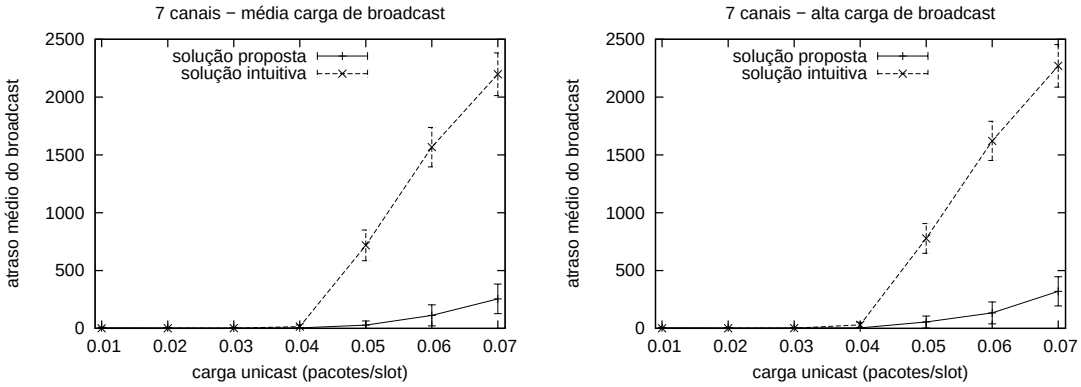
Pelos gráficos das Figuras 6(a) e 6(b), podemos verificar que apesar da solução intuitiva permitir que todos os seus vizinhos sejam alcançados por uma única transmissão,



(a) 7 canais - média carga (b) 7 canais - alta carga

Figura 5. Taxa de Entrega do Tráfego *Broadcast*

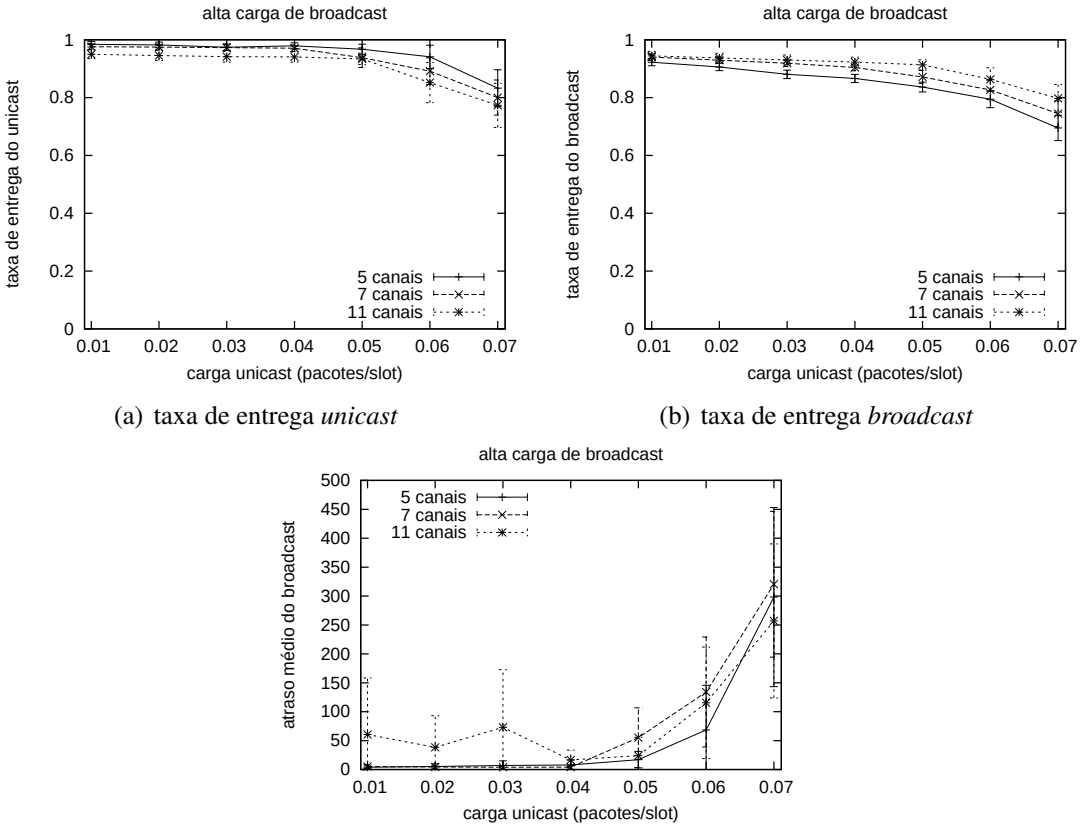
ela apresenta um pior desempenho em termos do atraso médio das mensagens *broadcast*. Quando a carga *unicast* ultrapassa um determinado valor, em torno de 0.04 pacotes/*slot* por nó, o atraso médio dessas mensagens cresce de forma acentuada, assim como a sua taxa de entrega, como evidenciado pelas Figuras 5(a) e 5(b). Portanto, a solução proposta favorece o tráfego *broadcast* em situações de alta carga na rede, aumentando a taxa de entrega e diminuindo o atraso médio na entrega das mensagens.



(a) 7 canais - média carga (b) 7 canais - alta carga

Figura 6. Atraso Médio do Tráfego *Broadcast*

Os gráficos da Figura 7 mostram as três métricas de desempenho para a solução proposta em função da carga *unicast* com um número crescente de canais. Desses gráficos podemos notar que quanto maior o número de canais, maior é a taxa de entrega do tráfego *broadcast*. No entanto, esse aumento no número de canais afeta negativamente a taxa de entrega do tráfego *unicast*, ou seja, a capacidade disponível na rede. A explicação desse fenômeno é a mesma dos resultados das Figuras 4 e 5. Com o aumento do número de canais, maior será o tamanho da sequência de *broadcast*, e mais tempo o US permanecerá fora da sequência de repouso e terá dificuldade em receber pacotes *unicast*. Desta forma, existe um compromisso com relação ao número de canais a serem utilizados nas sequências de saltos da solução proposta.



(a) taxa de entrega *unicast*

(b) taxa de entrega *broadcast*

(c) atraso médio *broadcast*

Figura 7. Influência do Número de Canais

7. Conclusão

O rádio cognitivo é uma tecnologia que permite um tipo de rede definido como DSA. Devido as suas regras de funcionamento, uma abordagem favorável é o uso de sequência de saltos de canais para acesso ao meio. Porém isso dificulta o encontro entre os nós, assim, diversos algoritmos de *rendezvous* foram criados. No entanto nestes algoritmos, as transmissões ou trocas de mensagens para múltiplos destinatários não é bem explorada. Um exemplo disso, são as transmissões de pacotes em *broadcast*, que acabam sendo inadequadas ou ineficientes em face aos procedimentos de saltos de canais.

Deste modo, propomos uma forma de acesso ao meio que se vale de algum algoritmo de *rendezvous* e da criação de uma sequência de *broadcast*. Nesta proposta, a partir de um critério de tipo de mensagem e/ou destinatário cria-se uma abordagem por papéis que define diferentes sequências de saltos de canais a fim de facilitar o compartilhamento do meio. Esta política de acesso foi comparada ao caso mais intuitivo de uso de apenas um canal ou uma mesma sequência de canais, que a princípio é uma boa opção, porém não atende bem a situações onde o acesso ao meio é muito exigido.

A partir dos resultados pode-se constatar que a política de saltos de canais empregada demostrou uma melhora na taxa de entrega de mensagens, seja de *broadcast* quanto de *unicast*, ao mesmo tempo em que gerou uma redução no atraso médio de recepção das mensagens para múltiplos usuários.

Referências

Akyildiz, I. F., Lee, W.-Y., Vuran, M. C., and Mohanty, S. (2006). Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey. *Computer Networks: The Int. J. of Comp. and Telecom. Networking*, 50:2127–2159.

Bahl, P., Chandra, R., and Dunagan, J. (2004). SSCH: Slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *Proceedings of ACM MobiCom*, pages 216–230, New York, NY, USA. ACM.

Bian, K., Park, J.-M., and Chen, R. (2011). Control channel establishment in cognitive radio networks using channel hopping. *IEEE Journal on Selected Areas in Communications*, 29(4):689–703.

Brar, G., Blough, D. M., and Santi, P. (2006). Computationally efficient scheduling with the physical interference model for throughput improvement in wireless mesh networks. In *Proceedings of ACM MobiCom*, pages 2–13, New York, NY, USA. ACM.

Cormio, C. and Chowdhury, K. R. (2010). Common control channel design for cognitive radio wireless ad hoc networks using adaptive frequency hopping. *Ad Hoc Networks*, 8(4):430–438.

DaSilva, L. A. and Guerreiro, I. (2008). Sequence-based rendezvous for dynamic spectrum access. In *Proceedings of IEEE DySPAN*, pages 1 –7.

Guedes, R. M., da Silva, M. W. R., Coutinho, P. S., and de Rezende, J. F. (2012). Agnostic broadcast rendezvous for cognitive radio networks using channel hopping. In *Proceedings of IEEE LCN*, pages 647–654.

Lin, Z., Liu, H., Chu, X., and Leung, Y.-W. (2011). Jump-stay based channel-hopping algorithm with guaranteed rendezvous for cognitive radio networks. In *Proceedings of IEEE INFOCOM*, pages 2444–2452.

Lo, B. F. (2011). A survey of common control channel design in cognitive radio networks. *Elsevier Physical Communication*, 4:26–39.

Luo, T., Motani, M., and Srinivasan, V. (2006). Cam-mac: A cooperative asynchronous multi-channel mac protocol for ad hoc networks. In *Proceedings of IEEE BROADNETS*, pages 1 –10.

Mo, J., So, H.-S. W., and Walrand, J. (2005). Comparison of multi-mhannel mac protocols. In *Proceedings of ACM MSWiM*, pages 209–218, New York, NY, USA. ACM.

NS-2. The network simulator - ns-2. `http://www.isi.edu/nsnam/ns/` - último acesso em 07/12/2012.

Silvius, M. D., Ge, F., Young, A., MacKenzie, A. B., and Bostian, C. W. (2008). Smart radio: Spectrum access for first responders. In *Proceedings of SPIE Defense and Security Conference*.

So, J. and Vaidya, N. H. (2004). Multi-channel mac for ad hoc networks: Handling multi-channel hidden terminals using a single transceiver. In *Proceedings of ACM MobiHoc*, pages 222–233, New York, NY, USA. ACM.

Theis, N. C., Thomas, R. W., and DaSilva, L. A. (2011). Rendezvous for cognitive radios. *IEEE Transactions on Mobile Computing*, 10(2):216–227.

Wellens, M., Riihijärvi, J., and Mähönen, P. (2009). Empirical time and frequency domain models of spectrum use. *Physical Communication*, 2(1-2):10–32.

Zhang, Y., Li, Q., Yu, G., and Wang, B. (2011). ETCH: Efficient channel hopping for communication rendezvous in dynamic spectrum access networks. In *Proceedings of IEEE INFOCOM*, pages 2471 –2479.