SIRCo - Um Protocolo de Sensoriamento Inteligente para Rádios Cognitivos

Ariel F. F. Marques[1], **Gilson M. Junior**[1], **Libério M. Silva**[1], **Luiz Henrique A. Correia**[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Lavras (UFLA)
Caixa Postal 3037, CEP 37200-000 Lavras-MG Brasil

{arielmarques,gilsonmj}@posgrad.ufla.br,
liba@computacao.ufla.br,lcorreia@dcc.ufla.br

Abstract. *The increased demand for using ISM frequencies (Industrial, Scientific and Medical) has caused interference problems and lack of availability of resources for wireless networks. Cognitive Radios (CR) emerged as an alternative to reduce interferences and intelligently use the spectrum. Several protocols have been developed to solve these problems, however, they focus only on spectrum sensing or spectrum decision. This work presents the ISCRa (Intelligent Sensing for Cognitive Radios), a MAC protocol based on neural networks and data obtained from regulatory agencies. Results demonstrate that SIRCo obtained higher delivery rate in high traffic, when compared to CogMAC and AHP protocols, while maintaining the latency.*

Resumo. *A grande demanda pela utilização do espectro de frequência livre ISM (Industrial, Scientific and Medical) tem causado problemas de interferência e falta de disponibilidade de recursos para redes sem fio. Os rádios cognitivos (RC) surgiram como alternativa para reduzir interferências e para o aproveitamento inteligente do espectro. Diversos protocolos surgiram com a finalidade de reduzir esses problemas, mas a maioria mas a maioria trata do problema de sensoriamento ou decisão do espectro. Este trabalho apresenta o protocolo MAC SIRCo (Sensoriamento Inteligente para Rádios Cognitivos), baseado em redes neurais e dados obtidos de órgãos de regulação. Resultados demonstram que o SIRCo obteve maior taxa de entrega no padrão de tráfego alto, quando comparado aos protcolos CogMAC e AHP, mantendo a mesma latência.*

1. Introdução

Nos últimos anos, o uso os dispositivos de comunicação sem fio, que utilizam a faixa ISM (*Industrial, Scientific and Medical*), tem crescido exponencialmente. A coexistência de diferentes dispositivos e redes que operam na mesma frequência (ou adjacentes) pode causar interferências prejudiciais, limitando a operação dos dispositivos ou sua desconexão da rede [Ferrari et al. 2008]. Atualmente, as frequências ISM, nas faixas de $2,4GHz$ e $5,8GHz$, são as mais utilizadas para o desenvolvimento de dispositivos ubíquos. Apesar disso, estudos mostraram que em algumas regiões o uso desse espectro de frequência é crítico, chegando a ter uma ocupação de até 90% [McHenry et al. 2006]. Em 2006, os autores [Zhou et al. 2006] já previam que no horizonte de 5-10 anos o crescimento das redes de comunicação sem fio, usando as bandas ISM, sofreria problemas de sobreposição, o que poderia afetar amplamente o seu funcionamento.

O IEEE propôs alguns mecanismos para a coexistência entre dispositivos para as redes IEEE 802.15.1 WPAN e IEEE 802.11 WLAN como: alterar o acesso ao meio sem fio, tráfego de pacotes aleatórios, supressão de interferência determinística, seleção de pacote adaptativa, entre outros [IEEE Std. 802.15.2 2003]. No entanto, a maioria dos dispositivos atuais ainda não suporta parte dessas recomendações. Nas últimas décadas, as redes de telefonia celular e outros sistemas de comunicação sem fio, têm usado técnicas para permitir a coexistência de dispositivos baseadas na alocação dinâmica do espectro ou DSA (*Dynamic Spectrum Access*), que incluem sensoriamento de espectro para escolher o melhor canal/frequência disponível e reconfigurar dinamicamente o rádio. Estes mecanismos são considerados a base de desenvolvimento dos rádios cognitivos ou inteligentes.

Os rádios cognitivos (RC) são capazes de explorar faixas do espectro ocupadas por usuários licenciados (primários) e não licenciados (secundários). Segundo [Akyildiz et al. 2008], redes baseadas em RC podem ser construídas para prover alta banda de transmissão para seus usuários em uma rede heterogênea com tecnologia DSA. Medições realizadas por [Patil et al. 2011] em grandes centros urbanos, mostraram que mesmo com a ocupação das faixas ISM e licenciadas, existe uma quantidade significativa de espectro que as redes baseadas em rádios cognitivos podem atuar de maneira oportunista.

A tecnologia de rádios cognitivos é recente e não existem protocolos eficientes que garantam a coexistência entre redes heterogêneas. Além disso, existe um número reduzido de trabalhos que realizam testes em hardware [Zia et al. 2013], tal lacuna na literatura caracteriza potenciais desafios para novos protocolos MAC (*Media Access Control*) baseados em rádio cognitivos. As USRPs (*Universal Software Radio Peripheral*) são um tipo de SDR (*Software Defined Radio*) desenvolvidas pela Ettus Research [Ettus 2014], capazes de lidar com amplas faixas do espectro.

Este trabalho propõe o protocolo SIRCo (Sensoriamento Inteligente para Rádios Cognitivos), que atua nas redes de rádios cognitivos. A meta deste protocolo é realizar o sensoriamento inteligente na camada física e na camada de enlace (MAC), controlar o acesso ao meio e otimizar o processo de decisão do espectro. O protocolo SIRCo tem a capacidade de considerar dados do histórico dos usuários primários (UP) obtidos do órgão de regulação ANATEL (Agência Nacional de Telecomunicações) para uma determinada região. O módulo de decisão do espectro é baseado em RNA (Redes Neurais Artificias) e no histórico dos UP, que proporciona a escolha otimizada do canal. O protocolo SIRCo foi comparado ao módulo de decisão do protocolo CogMAC [Ansari et al. 2010], baseado na leitura do RSSI (*Received Signal Strength Indication*), e a outro módulo de decisão baseado na técnica AHP (*Analytic Hierarchy Process*) [Correia et al. 2012]. Resultados mostram que o SIRCo tem uma taxa de entrega superior em até 76% e 57% do que os protocolos CogMAC e AHP, respectivamente.

Este trabalho está organizado como descrito a seguir. A Seção 2 apresenta os trabalhos relacionados. O protocolo SIRCo é descrito na Seção 3. A Seção 4 apresenta a metodologia do trabalho assim como os resultados obtidos e a Seção 5 apresenta as conclusões e os trabalhos futuros.

2. Trabalhos Relacionados

Segundo a norma FCC 03-322 da *Federal Communicatios Commission* (FCC), Rádio Cognitivo (RC) tem a capacidade de alterar seus parâmetros de transmissão baseado na interação com o ambiente [Commission 2003]. Para [Mitola and Maguire 1999] RC é um paradigma de comunicação sem fio em que o rádio utiliza todos os recursos disponíveis de forma eficiente, com a capacidade de se auto-organizar e se auto-reconfigurar.

Segundo [Akyildiz et al. 2008] redes de rádios cognitivos (RRC) têm que atender a quatro funcionalidades: Sensoriamento do Espectro (SE), Decisão do Espectro (DE), Compartilhamento do Espectro (CE) e Mobilidade do Espectro (ME). Durante o sensoriamento do espectro, os usuários RC escaneiam o espectro a fim de coletar amostras que serão utilizadas na fase DE para auxiliar na escolha do melhor canal.

Na fase DE os RC decidem qual é a melhor faixa de espectro no qual os dados podem ser transmitidos. Segundo [Lee and Akyldiz 2011] a DE é baseada na classificação do espectro, não somente pela observação local do usuário secundário, mas também pelas informações estatísticas de acesso do UP (nível de ruído base presente no canal, largura de banda, frequências utilizadas pelo UP, entre outras). Na fase de compartilhamento CE os RC comunicam, para seus vizinhos, as decisões tomadas na fase de decisão (DE). Após a rede cognitiva encontrar o melhor espectro, os rádios cognitivos são reconfigurados, na fase ME, para migrarem e atuarem na nova faixa de espectro.

O CogMAC é um protocolo para decisão do espectro que escolhe o melhor canal disponível em função do RSSI coletado pelo rádio [Ansari et al. 2010]. Esse protocolo utiliza o recurso de julgamento de canal livre CCA (*Clear Channel Assessment*) para transmissões. Portanto, o canal escolhido será aquele que esteja livre e possua o menor valor de RSSI. Assim, o rádio notifica seus vizinhos para migrarem para o novo canal. A utilização do RSSI como parâmetro único na decisão do espectro não é eficiente, já que o espectro de frequência é dinâmico e apresenta alta variabilidade.

Outro método de decisão do espectro para RRC é proposto em [Correia et al. 2012]. O método emprega informações estatísticas da presença de usuários primários na região monitorada, que são utilizadas como entrada de um classificador multi-parâmetros baseado em AHP [Saaty 2000]. São considerados, a probabilidade da chegada de usuários primários durante o monitoramento, os requisitos da aplicação e a influência de múltiplos parâmetros de entrada (RSSI, horário de coleta e frequência). Além disso, é proposto um mecanismo, baseado em entropia, que automaticamente determina os pesos dos parâmetros de entrada no método de decisão.

Em [He et al. 2010] são apresentadas várias aplicações de RNA em RC, em que a RNA foi utilizada para otimizar o sensoriamento do espectro em redes IEEE 802.11. Em [Sharma and Bohara 2014] é discutida a utilização de RNA aplicada para detecção, ou previsão, do canal do espectro. A ideia é que os US possam prever o estado do canal com base no histórico de detecção classificado e ordenado pela RNA. Com isso a rede pode prever se o canal estará ocioso ou não. Assim, os US podem consultar o histórico de detecção, eliminando o sensoriamento repetitivo, mas não consideram informações do UP. O SIRCo além de considerar uma base de dados fornecida pela ANATEL, cria uma base de dados onde são armazenadas as cinco melhores frequências considerando os UP, e ainda prevê qual o melhor canal disponível em um determinado instante.

Em [Huk and Mizera-Pietraszko 2015] é apresentado um comparativo entre duas técnicas de RNA utilizadas para a predição de canal, redes baseadas em Perceptron Multicamadas (MLP) e em *Sigma-If*. Estudos realizados, considerando essas duas técnicas, mostraram que a técnica *Sigma-If* é mais eficiente em relação a técnica MLP. Os autores trataram um exemplo de previsão do estado do canal como problema de previsão de séries binárias. Dado que neste trabalho a RRC foi utilizada em ambiente real, optou-se por utilizar o MLP no módulo de decisão por ser uma técnica mais estável e viável.

3. Protocolo SIRCo

O protocolo SIRCo utiliza uma abordagem inteligente para escolher o melhor canal em que a aplicação irá transmitir os seus dados. Essa escolha é realizada por uma rede neural artificial que considera o nível de ruído presente no espectro local e as informações dos UP (frequências de transmissão e de recepção da região de teste) disponibilizadas pela ANATEL. O SIRCo é capaz de gerenciar uma rede de rádios cognitivos (RRC), formada por nós escravos e um nó mestre, responsável por controlar a rede, centralizar as informações e realizar a decisão do espectro. Este trabalho propõe um protocolo para escolha do melhor canal para a transmissão de dados, todavia, o problema de roteamento de pacotes não é abordado.

3.1. Arquitetura do Nó Mestre

O nó mestre é responsável pela maior parte do processamento do protocolo SIRCo, assim como pelas funções de decisão e compartilhamento de espectro e funções de controle da rede (Descoberta de Vizinhos).

Algoritmo 1 SIRCo - Mestre

```
1:  ReconfiguraRadio(CCC);
2:  enquanto ciclos < 10 faça
3:      DescobertaVizinhos();
4:      list[] = CriaListVizinhos();
5:      se (ciclo == 0) OU (ciclo == 5) então
6:          SolicitaSenseCompleto(list[]);
7:      senão
8:          SolicitaSenseRestrito(list[]);
9:      fim se
10:     t1_ccc = getTime();
11:     enquanto Não recebeu resultados nenhum vizinho faça
12:         Espera resultados do sensoriamento;
13:         t2_ccc = getTime();
14:         se (t2_ccc - t1_ccc) > 200 então
15:             ReconfiguraRadio(CCC);
16:             Vai para linha 3;
17:         fim se
18:     fim enquanto
19:     t1 = getTime();
20:     enquanto (time ≤ 20) ∨ (Todos os vizinhos enviaram
    seus resultados) faça
21:         se Recebeu novo resultado então
22:             t1 = getTime();
23:         senão
24:             t2 = getTime();
25:             time= t2 - t1;
26:         fim se
27:     fim enquanto
28:     result_total = PreProcessaResults();
29:     melhor_freq = RNA(result_total);
30:     ComunicaVizinhos(melhor_freq);
31:     se (Se todos os nós vizinhos alteraram as suas frequências)
    OU (Tempo esgotado) então
32:         ReconfiguraRadio(melhor_freq);
33:     fim se
34:     t1 = getTime();
35:     enquanto time_trans ≤ 60 faça
36:         TransmiteDados();
37:         t2=getTime();
38:         time_trans = t2 -t1;
39:     fim enquanto
40:     ciclo++;
41: fim enquanto
```

No Algoritmo 1 é descrita a implementação em pseudocódigo do nó mestre. Nas linhas 1 a 4 é realizada a descoberta de vizinhos. Na linha 5 o nó mestre decide se solicita o sensoriamento completo do espectro (faixas de $0,8GHz$ a $5,8GHz$) ou o sensoriamento de um intervalo restrito do espectro. O sensoriamento do espectro é a fase mais demorada do protocolo e se todas as requisições de sensoriamento do nó mestre fossem das faixas

de $0,8GHz$ a $5,8GHz$, o protocolo não teria um desempenho satisfatório. Então optou-se por fazer esse sensoriamento completo a cada 5 ciclos de execução. Cinco ciclos com sensoriamento restrito corresponde aproximadamente a seis minutos, considerando a dinamicamente do espectro, apos seis minutos é desejável que seja feita uma leitura completa do espectro.

Após o nó mestre enviar as solicitações de sensoriamento do espectro (linhas 6 ou 8) ele entra em modo *idle* (linha 12). Entre as linhas 13 e 16, se após 200 segundos o nó mestre não receber informações de sensoriamento ele reconfigura seu rádio para o CCC. Foi definido 200 segundos pois, o tempo máximo de sensoriamento de espectro completo é aproximadamente 160 segundos. Na linha 16, o nó mestre retorna para a fase de descoberta de vizinhos.

Entre as linhas 13 e 28, o nó mestre recebe as informações oriundas do sensoriamento do espectro, essas informações são transmitidas no CCC ($6GHz$). Na linha 29 ocorre o pré-processamento das informações recebidas para que a RNA possa ser executada. Este módulo é capaz de classificar todas as frequências presentes na variável `result_total`, que retorna a melhor frequência na variável `melhor_freq`.

O nó mestre comunica todos os seus vizinhos para que alterem a frequência para a melhor frequência (linha 30), fase CE. Após todos os nós alterarem a sua frequência, o nó mestre, altera a frequência do seu rádio chamando a função `ReconfiguraRadio(melhor_freq)` (linha 32). A transmissão dos dados ocorre entre as linhas 34 e 39. As linhas 32 a 39 representam a fase ME. Na linha 40 a variável ciclo é incrementada. O algoritmo executa até completar os dez ciclos previamente definidos (tempo limite de execução).

A Figura 1 apresenta a implementação do nó mestre em diagrama de fluxo do GNU-Radio para os três protocolos (`SIRCo`(1), `AHP`(2) e `cogmac`(3)). O bloco `GENERIC MAC` (4) foi proposto por [Bloessl et al. 2013] que utiliza os mesmos métodos de acesso ao meio do padrão IEEE 802.15.4. Todos os outros blocos foram implementados em linguagem de programação C++/Python. Nesta figura as USRPs são conectadas à plataforma GNU-Radio pelos blocos `UHD:USRP SINK` (5) e `UHD:USRP SOURCE` (6).

O protocolo SIRCo inicia seu processo de comunicação na fase descoberta de vizinhos (bloco `message_generation` (7)), com o nó mestre solicitando o identificador (ID) de todos os seus vizinhos imediatos. Essa mensagem é enviada tanto no canal de controle comum (CCC), configurado em $6GHz$, quanto no canal corrente. Logo após, o nó mestre constrói uma lista de vizinhos com todos os IDs recebidos. O bloco `Message Parser Master` (8) é responsável por receber todas as mensagens no nó mestre. Com a lista dos vizinhos o nó mestre solicita sensoriamento para todos os vizinhos. Se ele não receber confirmação de sensoriamento do nó escravo, reenvia a solicitação até um tempo limite. Após o envio da solicitação de sensoriamento, o nó mestre entra em modo *idle*.

Existem dois tipos de sensoriamento, o completo e o sensoriamento restrito. Os dois tipos de sensoriamento serão abordados na Seção 3.2. Quando o nó mestre recebe dados do sensoriamento (`File Recorder Master` (9)) ele inicia um temporizador (`Timer` (10)), com isso, se um nó escravo perder a conexão, o nó mestre não irá esperar os resultados para sempre. Com todos os resultados recebidos ou com o temporizador esgotado (`Temporize ACK`(11)), ocorre o pré-processamento dos dados (`Preprocessor`

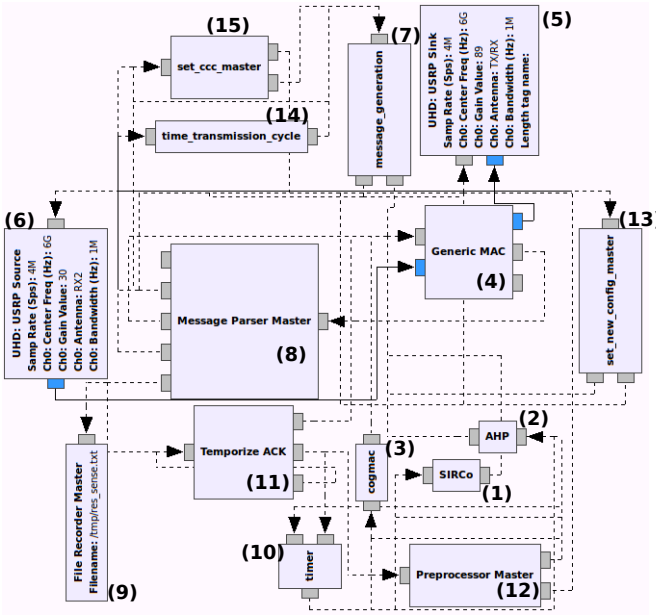

Figura 1. Implementação do nó mestre no GNU-Radio.

`Master` (12)) e então a fase de DE é inicializada. O pré-processamento é realizado para organizar todos os dados de forma que a RNA reconheça esses dados como entradas válidas. Na fase DE o módulo de RNA (`SIRCo` (1)) é inicializado.

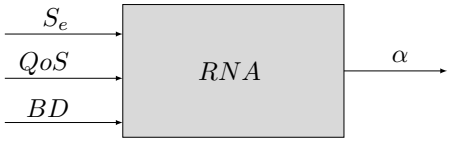

Figura 2. Detalhamento do módulo RNA.

Na Figura 2 observa-se as entradas S_e, QoS e BD para o módulo de classificação dos buracos do espectro. A entrada S_e corresponde às informações oriundas do sensoriamento do espectro, classificando uma chave (frequência, potência) por vez. A entrada *QoS* representa os requisitos da aplicação. A entrada BD representa a base de dados (ANATEL) dos UPs. Na saída α tem-se a classificação em ordem decrescente das frequências de acordo com a nota recebida pela RNA. O módulo DE cria uma lista com as 5 melhores frequências que serão utilizadas nas próximas execuções. Com isso, o tempo da fase de decisão do espectro é reduzido, já que a RNA não é executada, pois uma frequência é escolhida dentro da lista do intervalo sensoriado. Caso as frequências estejam fora do intervalo sensoriado ou a lista estiver vazia, a RNA é executada.

A RNA foi treinada com dados obtidos durante 7 dias ininterruptos, coletados do espectro local, permitindo prever em qual horário o espectro irá ficar livre, utilizando também os dados dos UPs disponibilizados pela ANATEL. Assim, a RNA pode ajustar os seus pesos considerando a existência de um usuário primário. Para o treinamento da RNA foram normalizados todos os dados (frequência, hora de coleta e potência) coletados, sendo utilizados 4 dias de treinamento e 3 de validação. A Tabela 1 apresenta as configurações utilizadas pela RNA. Após o módulo RNA executar, o nó mestre escolhe o melhor canal e inicia a fase de CE, comunicando para os seus vizinhos a nova frequência. A fase ME inicia quando todos os nós vizinhos enviarem mensagem de confirmação de recebimento da nova frequência, ou quando o temporizador esgotar.

Tabela 1. Configuração da RNA

Variáveis	Tipo/Valores
Arquitetura	Perceptrons de Múltiplas camadas (MPL)
Função de Ativação	Sigmóide
Treinamento	Supervisionado
Processo de Aprendizagem	Época
Neurônios intermediários	34
Neurônios de saída	1
Camadas escondidas	2
Momentum	0,0
Taxa de aprendizagem	0,7
Erro Médio Quadrático	0,0004

Durante a fase ME o nó mestre reconfigura o seu rádio para a nova frequência (`set_new_config_master` (13)), com isso tem início o ciclo de transmissão de dados (`time_transmission_cycle` (14)). O ciclo de Tx tem duração de 60 segundos, após esse intervalo o processo se reinicia. Quando o nó mestre fica ocioso ou não recebe informações de nenhum nó vizinho o bloco `set_ccc_master` (15) reconfigura o seu rádio para o canal de controle.

3.2. Arquitetura do Nó Escravo

O nó escravo é responsável por coletar as informações do ambiente por meio do sensoriamento do espectro, empacotá-las e transmiti-las para o nó mestre. No Algoritmo 2 observa-se a implementação em pseudocódigo do nó escravo. O nó escravo inicia em modo *idle* (linha 2). Na linha 3 o nó escravo recebe uma mensagem do nó mestre, essa mensagem deve ser decodificada para que o nó escravo realize suas ações (linha 5).

Algoritmo 2 SIRCo - Escravo

```
1:  enquanto ciclos < 10 faça
2:      Espera alguma mensagem do nó mestre;
3:      {Mensagem destinada ao nó escravo foi recebida}
4:      Inicio
5:          msg = DecodificaMSG();
6:          se msg.tipo == '0' então
7:              Responde com o seu próprio ID;
8:          fim se
9:          se msg.tipo == '2' então
10:             result = SensoriamentoEspectro();
11:             EnviaMsgOrigem(result);
12:         fim se
13:         se msg.tipo == '3' então
14:             EnviaMsgOrigem(ack);
15:             ReconfiguraRadio(msg.dado);
16:             t1 = getTime();
17:             enquanto time_trans ≤ 60 faça
18:                 rssi = EscutaEspectro();
19:                 se rssi < −85 então
20:                     TransmiteDados();
21:                     t2 = getTime();
22:                     time_trans = t2 - t1;
23:                 fim se
24:             fim enquanto
25:         fim se
26:         se msg.tipo == '4' então
27:             msg.tipo = '5'
28:             EnviaMsgOrigem(msg);
29:         fim se
30:         se msg.tipo == '5' então
31:             CalculaRTT();
32:         fim se
33:     Fim
34:     ciclo++;
35: fim enquanto
```

Entre as linhas 6 e 33, o nó escravo, realiza uma ação distinta para cada tipo de mensagem recebida. Se a mensagem for uma requisição de descoberta de vizinhos, o nó escravo, responde a mensagem com o seu ID (linha 7). Se a mensagem for uma requisição para o sensoriamento do espectro (linha 9), o nó escravo realiza o sensoriamento de acordo com o intervalo de frequências presente na mensagem recebida e preprocessa os dados para reduzir o *overhead*. Com a mensagem formatada, o nó escravo, a envia para o nó mestre por meio da função `EnviaMsgOrigem(result)` (linha 11).

Se a mensagem recebida pelo nó escravo for do tipo 3 (mudança de canal), o nó envia um ACK para o nó mestre informando que recebeu a mensagem (linha 14). Na

linha 15, o nó escravo, reconfigura o seu rádio para que opere na nova frequência, função `ReconfiguraRadio(msg.dado)`. Entre as linhas 16 e 24 ocorre a transmissão dos dados do usuário secundário da RRC. Antes de transmitir os dados é feita a leitura de RSSI da frequência em que o nó está operando para garantir que não há nenhum usuário primário presente na frequência atual, caso essa leitura resultar em um RSSI menor que -85 dBm a transmissão dos dados do US é iniciada (linhas 18 e 19). Se a mensagem for do tipo 4, então é uma mensagem de transmissão de dados de algum nó escravo da rede. O nó que recebeu essa mensagem troca o tipo da mensagem para o tipo 5 (ACK) para calcular o RTT da mensagem (linha 31).

A implementação do nó escravo, em diagrama de fluxo do GNU-Radio, é mostrada na Figura 3. As ações realizadas pelo nó escravo são divididas em duas fases: a primeira fase compreende o processo de sensoriar o espectro e comunicar ao nó mestre, e na segunda fase o nó escravo realiza as ações pertinentes à ME.

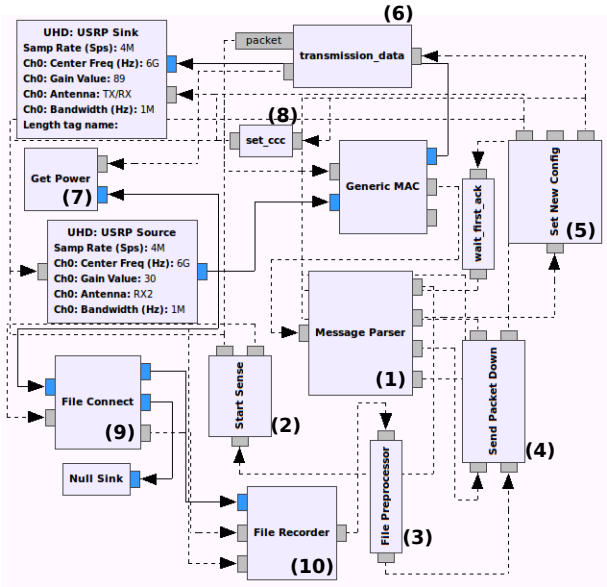

Figura 3. Implementação do nó escravo no GNU-Radio.

O processo se inicia com o nó escravo recebendo uma solicitação de descoberta de vizinhos, esta solicitação é enviada pelo nó mestre, com isso, o nó escravo envia uma mensagem (`Message Parser` (1)) contendo o ID do nó. A seguir, o nó mestre envia uma solicitação de sensoriamento do espectro. A mensagem de sensoriamento é recebida pelo nó destino para que o sensoriamento do espectro tenha início (`Start Sense` (2)).

O sensoriamento do espectro pode ser de dois tipos: o sensoriamento completo que compreende as faixas do espectro de frequências de $0,8GHz$ a $5,8GHz$ e o segundo tipo, o sensoriamento restrito que cobre o intervalo de $Freq-0,3GHz < Freq < Freq+0,3GHz$, sendo que $Freq$ é a frequência atual do dispositivo. O segundo sensoriamento foi idealizado para aumentar a performance do SIRCo já que a fase de sensoriamento do espectro é a mais demorada. Com o sensoriamento do espectro realizado, o nó escravo, pré-processa os dados (`File Preprocessor` (3)). Como a USRP amostra em média 16 vezes a mesma faixa de frequência, o nó escravo seleciona a amostra que contém o maior ruído base, para então enviar essa informação para o nó mestre. Com isso, o nó escravo diminui o *overhead* da rede enviando menos pacotes ao nó mestre. Após o nó escravo enviar as suas informações (`Send Packet Down` (4)) do sensoriamento, o nó

mestre tem uma lista com todas as informações necessárias para realizar a decisão do espectro. Depois que todas as mensagens são enviadas o nó escravo aguarda o início da fase 2.

A fase 2 inicia com o nó mestre enviando uma solicitação para migração de canal. Após receber a solicitação de migração de canal (`Message Parser` (1)), o nó escravo, envia uma confirmação de migração de canal para migrar para o novo canal (`Set New Config` (5)). Com isso, o nó escravo está apto para começar a transmitir os dados da aplicação (`transmission_data` (6)). A transmissão dos dados ocorre dentro de um intervalo de 60 segundos, que pode ser configurado de acordo com a aplicação. Antes de transmitir os dados, o nó escravo faz uma leitura de RSSI no novo canal (`Get Power` (7)) para garantir que nenhum UP esteja este canal. A transmissão ocorre se a leitura retornar um valor inferior a $-85dBm$.

Quando um nó escravo perde a comunicação com o nó mestre, o bloco `set_ccc` (8) age como mecanismo de resiliência da rede, reconfigurando o canal do nó escravo para o CCC ($6GHz$). O bloco `File Connect (9)` e `File Recorder (10)` é responsável por armazenar as informações obtidas durante o sensoriamento do espectro.

4. Resultados e Discussão

Neste trabalho foram utilizadas as USRPs modelos B200 e B210, e o framework GNU-Radio, recomentado pela [Ettus 2014]. Também foi utilizada a máquina volk (*Vector-Optimized Library of Kernels*) avx_64_mmx para converter os valores gerados pelas USRPs do universo dos números complexos para números reais (ponto flutuante) [West 2015]. Os nós considerados neste trabalho são compostos pelas placas USRPs e por computadores, cujas configurações são apresentadas na Tabela 2.

Tabela 2. Configuração de hardware

Software/Dispositivo	Versão/Modelo
Sistema Operacional	Ubuntu 14.04 LTS Kernel 3.16
GNU-Radio	v3.7
VOLK Machine	avx_64_mmx
Processador PC	Intel Core i7-4770 3.40GHz
Memória PC	8 GB
USRP	B200/B210

Na Tabela 3 são apresentados os padrões de tráfego para os testes realizados: *Low*-L, *Medium*-M e *High*-H. Os intervalos entre as mensagens para esses padrões de tráfegos foram escolhidos para medir a entrega de pacotes para as taxas de transmissão de aproximadamente de $1Mbps, 4Mbps$ e $9,5Mbps$.

Tabela 3. Padrão de Tráfego

Frequência de Mensagem	Intervalo de Mensagem
Low - L	50 ms
Medium - M	12 ms
High - H	8 ms

Para avaliar a performance do SIRCo foram implementado os protocolos CogMAC e AHP, considerando as métricas de taxa de entrega, latência e a variação da frequência por classe e por número de ciclos em três cenários distintos: dois, quatro e seis nós. O protocolo CogMAC foi escolhido por sua simplicidade, já que considera somente um parâmetro físico do canal (RSSI). Como alternativa mais sofisticada foi escolhido um protocolo baseado na técnica AHP, cujo modelo de decisão do espectro considera mais de um parâmetro. Os cenários de testes foram realizados em ambientes fechados e reais,

portanto sujeitos a interferências. A aplicação desenvolvida para os testes geram *streans* de bytes, assim a aplicação simula um usuário RC transmitindo dados genéricos. Esses experimentos foram realizados em um ambiente fechado.

Complementando a avaliação de performance, foi O primeiro cenário da Figura 4-(a) consiste de dois nós, um nó mestre e um nó escravo. Este cenário foi desenvolvido para verificar a conectividade entre os nós e testar as heurísticas propostas para as fases cognitivas.

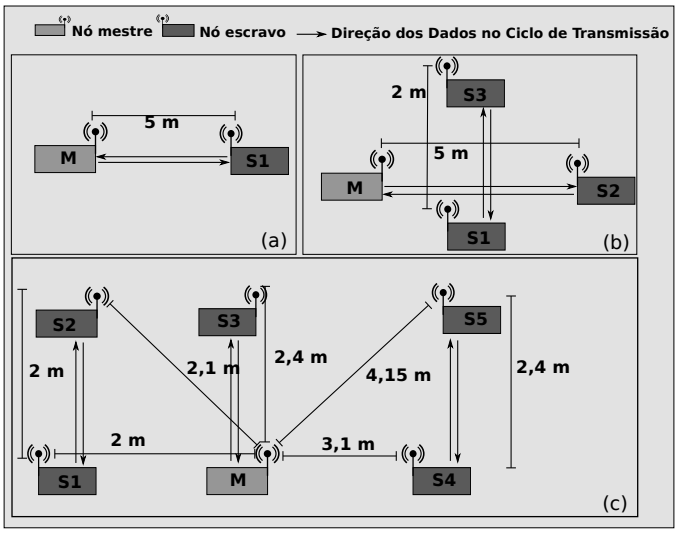

Figura 4. Cenários para testes.

Para o cenário 2, apresentado na Figura 4-(b), a rede foi composta por quatro nós. As trocas de mensagens no ciclo de transmissão de dados ocorreram de forma ortogonal entre os pares de nós (S2,M) e (S3,S1). Já o cenário 3, a rede foi formada por 6 nós, Figura 4-(c). As trocas de mensagens no ciclo de transmissão de dados ocorreram entre os pares de nós (S3,M), (S2,S1) e (S4,S5).

Para todos os cenários foram executadas 3 repetições de 10 ciclos completos do nó mestre para cada um dos protocolos avaliados (SIRCo, AHP e CogMac). Cada repetição foi executada seguindo o padrão de tráfego da Tabela 3. Os dados para avaliação foram coletados durante cada execução, obtendo o tempo de uma mensagem ser enviada e recebida pelo seu emissor (RTT) e o número de pacotes enviados e recebidos por cada par de nó. Juntamente com essas informações, foram coletadas as frequências que cada protocolo escolheu em seu método de decisão. Todos os resultados obtidos consideram um intervalo de confiança de 95% e largura de banda de $1MHz$ (Canal).

Tabela 4. Classes de Frequências

Frequências (MHz)	Classes
800 - 1499	1
1599 - 2199	2
2200 - 2999	3
3000 - 3699	4
3700 - 4399	5
4400 - 5199	6
5200 - 5800	7
6000	CCC

Considerando os padrões de tráfego foi avaliada a variação da frequência em função de suas classes e do número de ciclos. As classes de frequências foram divididas em intervalos de $700MHz$, como mostrado na Tabela 4.

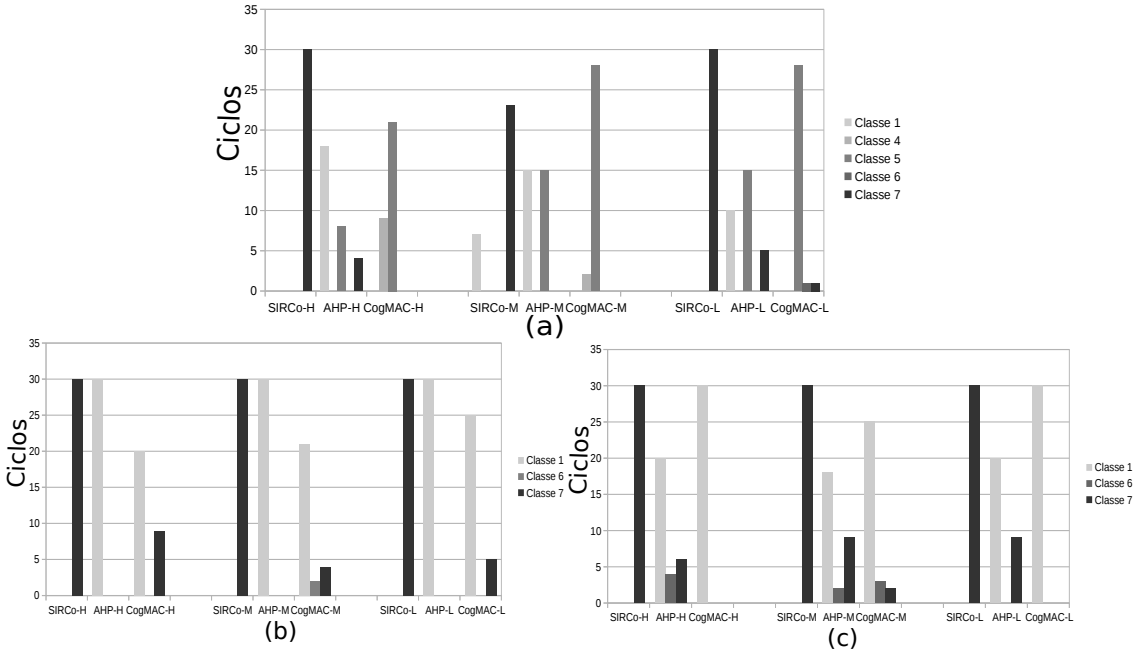

Figura 5. Variação da frequência por classe e por número de ciclos.

Na Figura 5 são apresentadas as variações das frequências para os três casos de teste. Na Figura 5-(a) tem-se os dados referentes ao primeiro cenário. Para os três padrões de tráfego o SIRCo permaneceu por mais ciclos na classe 7, sendo que os protocolos CogMAC e AHP variaram entre as demais classes, exceto nas classes 2 e 3. Da mesma forma, para os gráficos das Figuras 5-(b) e 5-(c) o SIRCo se manteve na classe 7 por mais ciclos que os protocolos CogMAC e AHP. O sensoriamento parcial do SIRCo garante que a variação de frequência seja realizada em passos de $-0,3GHz$ a $+0,3GHz$ a partir da frequência atual. Esse tipo de sensoriamento possibilita que o SIRCo consiga manter-se na classe 7 por mais ciclos, garantindo a maior largura de banda. No gráfico da Figura

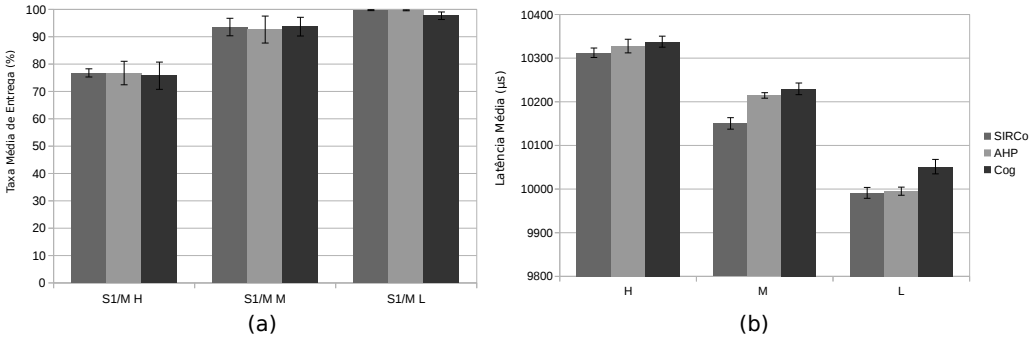

Figura 6. Resultados do Cenário 1.

6-(a) é apresentada a taxa média de entrega de pacotes para o primeiro cenário. Observa-se que os três protocolos avaliados obtiveram as taxas médias de entregas similares. A justificativa para esses resultados é que o primeiro cenário é composto de apenas 2 nós e que o tráfego gerado de mensagens não satura o canal e não causa perdas acentuadas de pacotes. A não saturação do canal implica que a escolha das classes de frequências não causa variações acentuadas nas taxas de entrega dos três protocolos.

O gráfico da Figura 6-(b) mostra que o SIRCo obteve o melhor resultado no padrão de tráfego baixo. Isso se justifica pois, o SIRCo escolhe as frequências do espec-

tro próximo a $6GHz$, essas frequências apresentam melhor qualidade de canal, tanto de largura de banda quanto de ruído base. É importante observar que a latência apresentada representa o RTT entre os nós. Assim, a latência efetiva é metade do valor apresentado, assumindo que os enlaces são simétricos.

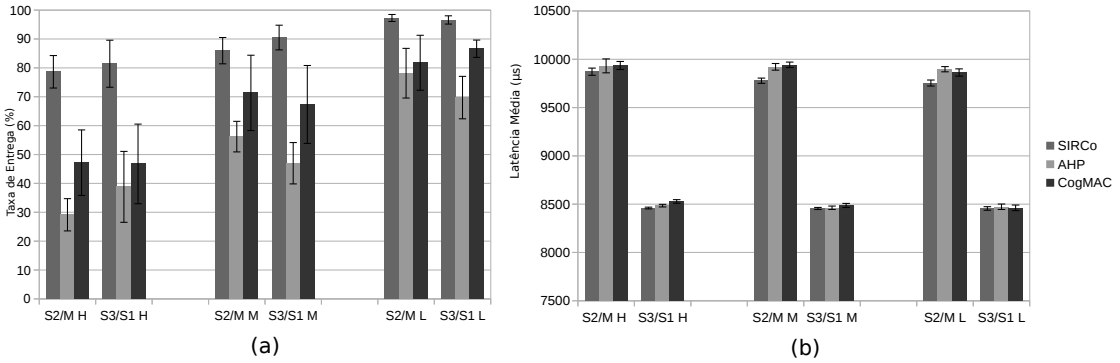

Figura 7. Resultados do Cenário 2.

Na Figura 7 são apresentados os resultados para o cenário 2, no qual a troca de mensagens ocorreu entre os pares de nós: mestre (M) e escravo 2 (S2); escravo 3 (S3) e escravo 1 (S1). A Figura 7-(a) apresenta a média da taxa de entrega. O protocolo SIRCo obteve uma taxa de entrega superior aos protocolos CogMAC e AHP. Para o tráfego alto o SIRCo foi superior 57% e 41% aos protocolos AHP e CogMAC, respectivamente. No padrão de tráfego médio M, o SIRCo foi superior 41% e 21% que os protocolos AHP e CogMAC, respectivamente. No padrão de tráfego baixo L o SIRCo foi superior 23% e 13% que os protocolos AHP e CogMAC, respectivamente. A taxa de entrega do SIRCo seguiu um padrão constante entre os modelos de tráfego, à medida em que o tráfego de mensagens diminuiu, a taxa de entrega aumentou. Os outros protocolos, especialmente o AHP, obtiveram os piores resultados. Esse resultado se deve ao fato do SIRCo possuir um módulo de RNA, cujo o treinamento classifica as melhores frequências, já nos outros protocolos não existe esse treinamento. Essa variação de frequências, dos protocolos AHP e CogMAC, pode ser observada no gráfico da Figura 5-(b).

Na Figura 7-(b) é apresentada a latência média (RTT). Observa-se que não houve variações significativas entre os protocolos devido a proximidade dos nós. No geral, o SIRCo leva pequena vantagem sobre os demais protocolos analisados. A Figura 8 apre-

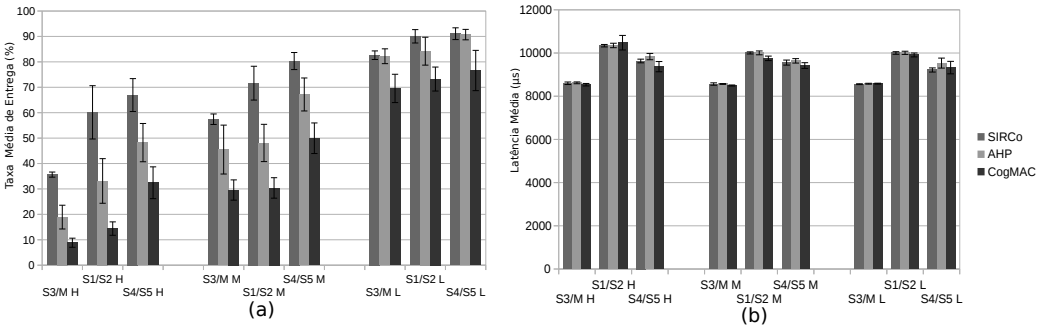

Figura 8. Resultados do Cenário 3.

senta os resultados para o experimento do cenário 3. O objetivo desse experimento é mostrar a escalabilidade do protocolo proposto. Na Figura 8-(a) são apresentados os gráficos de taxa média de entrega de pacotes e na Figura 8-(b) é mostrada a latência média. Igualmente aos outros cenários a latência foi calculada como RTT.

Na Figura 8-(a) observa-se que o SIRCo foi melhor 39% e 76% que os protocolos AHP e CogMAC para o padrão tráfego alto H, respectivamente. No padrão de tráfego médio M o SIRCo foi melhor 23% e 47% que os protocolos AHP e CogMAC, respectivamente. No padrão de tráfego L, o SIRCo obteve um desempenho semelhante ao AHP, SIRCo 2% melhor. Em relação ao CogMAC, o SIRCo foi melhor 16% no padrão de tráfego baixo L. A redução na taxa de entrega para o padrão de tráfego alto H é explicada pela saturação do canal, já que existem três pares de nós enviando mensagens com intervalo de $8ms$ gerando um grande fluxo de mensagens próximo à capacidade do canal. Observa-se que o SIRCo e o AHP tiveram desempenho melhor em relação ao CogMAC, pois permaneceram por mais ciclos na classe 7.

No gráfico de latência da Figura 8-(b) não ocorreram variações significativas. No geral os três modelos testados obtiveram desempenho similar.

5. Conclusões

Neste trabalho foi apresentado um protocolo MAC inteligente para redes de rádios cognitivos, o SIRCo (Sensoriamento Inteligente para Rádios Cognitivos). O módulo de decisão do SIRCo é baseado na técnica RNA que considera uma base de dados da ANATEL juntamente com medições do espectro local para classificar as faixas de frequências. Com isso, o módulo foi capaz de escolher os melhores canais. O protocolo SIRCo foi desenvolvido para USRPs e implementado no *framework* GNU-Radio.

Os resultados mostraram que o SIRCo obteve o melhor desempenho na maioria dos cenários. Para os três padrões de tráfego o SIRCo permaneceu por mais ciclos na classe de maior frequência. Os protocolos CogMAC e AHP variaram a escolha das frequências mais vezes, incluindo classes de frequências baixas. Dessa forma, o SIRCo obteve maior taxa de entrega com latência ligeiramente menor que os outros protocolos.

Como trabalhos futuros a proposta é avaliar novos cenários com UP que troquem de frequência, distâncias maiores entre os nós, analisar a questão de mobilidade dos nós e a escalabilidade da rede.

6. Agradecimentos

Os autores agradecem o apoio financeiro dos órgãos de fomento CNPq e FAPEMIG.

Referências

Akyildiz, I., Lee, W.-Y., Vuran, M. C., and Mohanty, S. (2008). A survey on spectrum management in cognitive radio networks. *Communications Magazine, IEEE*, 46(4):40–48.

Ansari, J., Zhang, X., and Mähönen, P. (2010). A decentralized mac for opportunistic spectrum access in cognitive wireless networks. In *Proceedings of the 2010 ACM Workshop on Cognitive Radio Networks*, CoRoNet '10, pages 13–18, New York, NY, USA. ACM.

Bloessl, B., Leitner, C., Dressler, F., and Sommer, C. (2013). A GNU Radio-based IEEE 802.15.4 Testbed. In *12. GI/ITG KuVS Fachgespräch Drahtlose Sensornetze (FGSN 2013)*, pages 37–40, Cottbus, Germany.

Commission, F. C. (2003). Fcc 03-322. Digital.

Correia, L., Oliveira, E., Macedo, D., Moura, P., Loureiro, A., and Silva, J. (2012). A framework for cognitive radio wireless sensor networks. In *Computers and Communications (ISCC), 2012 IEEE Symposium on*, pages 000611–000616.

Ettus, M. (2014). Ettus research. Technical report.

Ferrari, P., Flammini, A., Marioli, D., Sisinni, E., and Taroni, A. (2008). Coexistence of wireless sensor networks in factory automation scenarios. *Sensors Transducers Journal*, 90(4):48–60.

He, A., Bae, K. K., Newman, T., Gaeddert, J., Kim, K., Menon, R., Morales-Tirado, L., Neel, J., Zhao, Y., Reed, J., and Tranter, W. (2010). A survey of artificial intelligence for cognitive radios. *Vehicular Technology, IEEE Transactions on*, 59(4):1578–1592.

Huk, M. and Mizera-Pietraszko, J. (2015). Contextual neural-network based spectrum prediction for cognitive radio. In *Future Generation Communication Technology (FGCT), 2015 Fourth International Conference on*, pages 1–5.

IEEE Std. 802.15.2 (2003). Recommended Practice for Telecommunications and Information exchange between systems. Local and metropolitan area networks Specific Requirements – Part 15.2: Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Band.

Lee, W.-Y. and Akyldiz, I. (2011). A spectrum decision framework for cognitive radio networks. *Mobile Computing, IEEE Transactions on*, 10(2):161–174.

McHenry, M. A., Tenhula, P. A., McCloskey, D., Roberson, D. A., and Hood, C. S. (2006). Chicago spectrum occupancy measurements & analysis and a long-term studies proposal. In *Proceedings of the first international Workshop on Technology and policy for accessing spectrum (TAPAS), 2006*, pages 1–12.

Mitola, J. and Maguire, G.Q., J. (1999). Cognitive radio: making software radios more personal. *Personal Communications, IEEE*, 6(4):13–18.

Patil, K., Prasad, R., and Skouby, K. (2011). A survey of worldwide spectrum occupancy measurement campaigns for cognitive radio. In *Devices and Communications (ICDeCom), 2011 International Conference on*, pages 1–5.

Saaty, T. (2000). *Fundamentals of the Analytic Hierarchy Process*. RWS Publications, 4922 Ellsworth Avenue, Pittsburgh, PA 15413.

Sharma, V. and Bohara, V. (2014). Exploiting machine learning algorithms for cognitive radio. In *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on*, pages 1554–1558.

West, N. (2015). Vector-optimized library of kernels. Technical report.

Zhou, G., Stankovic, J. A., and Son, S. H. (2006). Crowded spectrum in wireless sensor networks. Workshop on Embedded Networked Sensors.

Zia, M., Qureshi, F., and Shah, S. (2013). Energy efficient cognitive radio mac protocols for adhoc network: A survey. In *Computer Modelling and Simulation (UKSim), 2013 UKSim 15th International Conference on*, pages 140–143.