ESCALONAMENTO DISTRIBUÍDO LIVRE DE COLISÕES PARA REDES DE SENSORES SEM FIO COM MÚLTIPLAS FONTES E MÚLTIPLOS SUMIDOUROS

Alexandre Alves dos Santos

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadores: Felipe Maia Galvão França
Daniel Ratton Figueiredo

Rio de Janeiro
Março de 2012

ESCALONAMENTO DISTRIBUÍDO LIVRE DE COLISÕES PARA REDES DE SENSORES SEM FIO COM MÚLTIPLAS FONTES E MÚLTIPLOS SUMIDOUROS

Alexandre Alves dos Santos

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Examinada por:

Prof. Felipe Maia Galvão França, Ph.D.

Prof. Daniel Ratton Figueiredo, Ph.D.

Prof. Valmir Carneiro Barbosa, Ph.D.

Prof. Rosa Maria Meri Leão, Dr.

Prof. Antonio Alfredo Ferreira Loureiro, Ph.D.

Prof. Juraci Ferreira Galdino, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2012

Santos, Alexandre Alves dos

Escalonamento Distribuído Livre de Colisões para Redes de Sensores Sem Fio com Múltiplas fontes e Múltiplos Sumidouros/Alexandre Alves dos Santos. – Rio de Janeiro: UFRJ/COPPE, 2012.

XIII, 90 p.: il.; 29, 7cm.

Orientadores: Felipe Maia Galvão França

Daniel Ratton Figueiredo

Tese (doutorado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2012.

Referências Bibliográficas: p. 78 – 81.

1. Mecanismo TDMA. 2. Sem Colisões. 3. Escalonamento por Reversão de Arestas. I. França, Felipe Maia Galvão *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Ao meu pai Victor †, a minha mãe Elce e a meu filho e meu anjo bom, Guilherme.

Agradecimentos

Agradeço imensamente ao professor Felipe França por ter sido pessoa tão extraordinária dentro de um professor único, que me orientou como aluno e me apoiou fraternamente nas dificuldades da vida.

Agradeço imensamente ao professor Daniel Ratton, cujas admiráveis capacidades laborativa e intelectual foram fundamentais para que eu conseguisse chegar ao dia da defesa deste trabalho.

Agradeço ao meu pai Victor que partiu desta vida em 2009 durante este curso, mas que sempre me presenteou com o melhor que podia dar: o estudo e o exemplo.

Agradeço a minha mãe Elcenir por representar meu pai e a si mesma no meu suporte, nos momentos da minha vida nos quais precisei de um porto seguro, ou até mesmo, de um rebocador. Que Deus a abençoe sempre e que me ilumine para que eu retribua nesta vida este amor materno.

Agradeço a meu filho Guilherme, que tinha dois anos quando comecei este doutorado e oito quando acabei, pela inconsciente compreensão da ausência do pai em dezenas de ocasiões que não estive presente. Como pai, humildemente espero poder deixar exemplos positivos para sua vida.

Agradeço a minha ex-mulher Mauren pelos 15 anos em que compartilhamos conquistas pessoais, profissionais e a dádiva de um filho.

Gostaria de agradecer a algumas pessoas que foram especiais nos últimos anos e, particularmente, na fase final deste curso de doutorado. André Pinho, Francisco Benjamim, Alexandre de França, Antonio Carlos Castañon, entre outros. Estarei orando, rezando e meditando por estas pessoas e suas famílias! É a forma mais honesta que conheço para agradecê-los. Deus os abençoe todos os dias!

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

ESCALONAMENTO DISTRIBUÍDO LIVRE DE COLISÕES PARA REDES DE SENSORES SEM FIO COM MÚLTIPLAS FONTES E MÚLTIPLOS SUMIDOUROS

Alexandre Alves dos Santos

Março/2012

Orientadores: Felipe Maia Galvão França
Daniel Ratton Figueiredo

Programa: Engenharia de Sistemas e Computação

Em uma rede de sensores sem fio o consumo de energia é crucial para a vida útil da rede e é essencial minimizar seu gasto. Para evitar qualquer desperdício de energia deve-se avaliar as possibilidades do uso de múltiplos sinks e de transmissões livres de colisão.

Este trabalho apresenta duas versões de um mecanismo distribuído de escalonamento livre de colisões para uso em rede de sensores sem fio com múltiplos sinks onde os nós possuem demandas distintas de transmissão. Ambas versões produzem escalonamentos em que os nós lidam com problemas inerentes a comunicações sem fio, tais como a colisão de pacotes, escuta ociosa ou recepção inúteis. A primeira versão, denominada *broadcast*, permite que todos os vizinhos recebam a transmissão de um nó, enquanto a segunda, denominada *unicast*, permite que apenas um vizinho específico receba a transmissão de um nó e que os dados sejam encaminhados pela rede de acordo com rotas previamente definidas.

A análise do mecanismo através de simulações de diferentes cenários apresenta resultados que avaliam a concorrência na operação dos nós da rede em relação a diferentes parâmetros tais como conectividade média dos nós, diferentes demandas de transmissão, existência de múltiplos sinks e algoritmos de orientação acíclica.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

A COLLISION FREE DISTRIBUTED SCHEDULING IN WIRELESS SENSOR NETWORKS WITH MULTIPLE SOURCES AND MULTIPLE SINKS

Alexandre Alves dos Santos

March/2012

Advisors: Felipe Maia Galvão França
Daniel Ratton Figueiredo

Department: Systems Engineering and Computer Science

In a wireless sensor network the energy consumption is crucial for the network live cycle and it is essential to minimize its cost. In order to avoid energy waste, one should evaluate the possibilities of using multiple sinks and collision free communication.

This work presents two versions of a distributed collision free scheduling mechanism for wireless sensor networks with multiple sinks where nodes have different transmission demands. Both versions generate schedulings in which nodes deal with wireless communication problems such as packet collision, idle listening and overhearing. The first version, called *broadcast*, allows that all one hop neighbors receive the transmission from a node, while the second one, called *unicast*, allows that only a specific neighbor receives the transmission and that the data be sent through the network by the previously defined routes.

The analysis of the mechanism using different simulation scenarios shows results that evaluate the concurrency in the network node operation regarding different parameters such as mean connectivity, different transmission demands, multiple sinks and algorithms for acyclic orientation.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Motivação

Nos últimos anos a computação distribuída tem sido largamente empregada em novos domínios de aplicações, gerando demandas por novas técnicas e algoritmos em vários níveis da engenharia de sistemas computacionais. Um destes campos que encontra-se em permanente evolução, particularmente devido a multiplicidade de aplicações, são as redes de sensores sem fio (RSSF). Sensores são dispositivos computacionais com capacidade de comunicação, geralmente miniaturizados e com fonte limitada de energia, sendo empregados em diversas áreas, tais como saúde, indústria, robótica, monitoração de habitats, monitoração do clima e na área militar. Particularmente nesta última, são extremamente complexas e variadas as possibilidades de aplicações, sendo motivo de destaque o seu uso em sistemas de busca, acompanhamento e reconhecimento de alvos, sistemas de vigilância, sistemas de comando e controle, sistemas de guerra eletrônica e cibernética, sistemas de mísseis e foguetes e em sistemas de veículos aéreos, terrestres e marítimos não tripulados [1].

Os sensores são dispositivos que geralmente produzem grandes quantidades de dados que podem ser armazenados, tratados, agregados ou transmitidos. Além da função primária de sensorear, realizada pelo elemento sensor, o avanço das tecnologias em diversas áreas tem definido novas funções a estes dispositivos, tais como processar dados e comunicar-se com outros sensores utilizando o espectro eletromagnético. Entretanto, um dos recursos mais escassos em um sensor é a energia para mantê-lo em funcionamento, pois na maioria das vezes estes operam com baterias.

O agrupamento destes dispositivos através do estabelecimento entre eles de enlaces rádios, óticos ou infra-vermelhos, dá origem a rede de sensores sem fio (RSSF). As RSSFs são tipicamente formadas por uma grande quantidade de nós[1] cujas arquite-

[1]Neste texto, os termos nó, nó sensor, sensor, processo e processador serão utilizados como sinônimos.

turas e topologias são influenciadas por vários fatores, incluindo tolerância a falhas, escalabilidade, custo de produção, ambiente de operação, restrições de hardware, meio de transmissão e consumo de energia [2].

Inicialmente uma RSSF poderia ser considerada apenas uma variação de um outro tipo de rede, conhecido como redes *ad-hoc*. As redes *ad-hoc*, também chamadas de *Mobile Ad hoc Network* (MANET), tem a premissa de criar uma infra-estrutura de comunicação sem fio para suportar o tráfego entre os aplicativos que são executados nos dispositivos computacionais que a formam. Entretanto, diferentemente das redes *ad-hoc*, uma RSSF tem como principal objetivo sensorear e transmitir estes dados através dos sensores que compõem a rede, para um ou mais nós de destino, especialmente chamados de *sorvedouros*, *sumidouros*, *nós sinks* ou simplesmente *sinks* [3].

Desenvolvimentos recentes em RSSF têm apresentado cenários onde os dados sensoriados precisam ou podem ser enviados para múltiplos sinks. Esta arquitetura de rede é requerida quando existem vários pontos de coleta de um monitoramento ou quando a mesma RSSF está servindo a múltiplas aplicações, cada uma executando serviços diferentes. Há ainda aplicações em que nós específicos da rede possuem funções de atuadores dentro de determinado cenário, participando apenas da transmissão dos dados e corroborando a necessidade do envio de dados de múltiplas fontes para múltiplos sinks.

Existem diversas formas de tratar um dado sensoriado ou recebido no caminho entre os nós sensores e os sinks. As aplicações definem como este tratamento ocorre nos sensores, ou seja, o nó pode, por exemplo: coletar e transmitir os dados; somente retransmitir dados recebidos; ou, agregar informações de um ou mais nós a fim de retransmiti-las de modo mais eficiente. A agregação ou fusão de dados é uma estratégia para economizar energia e dar maior escalabilidade à rede e que deve levar em consideração o custo desta agregação. Por exemplo, em uma rede com objetivo de coletar informações da temperatura ambiente, o cálculo da temperatura máxima, mínima ou média, entre valores recebidos pelo nó sensor, tem um custo que pode ser considerado insignificante para a vida útil do nó. Entretanto, se for considerado o custo de um nó decifrar uma transmissão criptografada, realizar uma agregação de imagens e criptografar novamente para retransmissão, certamente este processamento terá um custo computacional elevado. Assim, verifica-se que a avaliação dos custos de fusão, de recepção e transmissão, determina as decisões de envio de pacotes em direção ao sink [4].

Independente da escolha do algoritmo que determina o processamento que ocorre no sensor, cada nó da rede possui uma quantidade de informação que precisa ser transmitida a um ou mais nós vizinhos. Esta quantidade de informação, chamada de *demanda*, vai determinar a quantidade de tempo que o nó sensor necessita utilizar

o meio eletromagnético, para escoar sua demanda de dados pela rede.

O espectro eletromagnético é um meio físico compartilhado e utilizado pelos rádios para enviarem seus sinais entre os nós sensores. É um meio muito vulnerável as atenuações do ambiente e as interferências decorrentes de outros canais rádio ou sinais que coexistem no mesmo espectro eletromagnético. Para que possam se comunicar de forma adequada, os nós da rede precisam saber lidar ou evitar os problemas decorrentes deste meio de transmissão, que geralmente são abordados pelos *protocolos de acesso ao meio* (MAC).

De um modo geral, protocolos MAC são classificados em baseados em contenção e baseados em escalonamento. Os primeiros, tendem a se adaptar mais facilmente a mudanças de topologia e questões de sincronização, mas geralmente possuem alto custo com as colisões de pacotes, escutas ociosas[2] e recepções inúteis[3]. Por outro lado, os protocolos baseados em escalonamento são eficientes em evitar colisões, escutas ociosas e recepções inúteis, entretanto lidam com dificuldades com os requerimentos de diferentes demandas, sincronização e mudanças de topologia. Estas características serão detalhadas no Capítulo 2.

Definir como escoar as demandas dos nós sensores para os nós sinks através do uso do canal rádio é crucial para um funcionamento eficiente e uma prolongada vida útil do dispositivo sensor e da própria rede. Protocolos como o IEEE 802.11 desperdiçam grande quantidade de energia com a escuta ociosa e recepção inútil, e mesmo com o esforço dos pesquisadores em criar dispositivos eletrônicos e rádios de baixíssimo consumo energético, o projeto de um protocolo MAC eficiente no consumo de energia é fundamental para a rede.

Torna-se então necessário, um estudo de viabilidade que deve considerar aquelas que são as principais causas de perdas de energia com relação ao uso do rádio em um nó sensor: a colisão de pacotes, a escuta ociosa, a recepção inútil de pacotes de outros destinatários e a sobrecarga de pacotes de controle nos protocolos MAC [2, 3, 5]. Evitar as colisões de pacotes de dados decorrentes do problema do terminal oculto, mostrado na Figura 1.1, é um dos desafios que o mecanismo proposto nesta dissertação aborda. Na Figura 1.1, o nó n_1 e o nó n_3 não sabem da existência um do outro, e transmitem ao mesmo tempo para o nó n_2, causando colisão e inviabilizando o recebimento de qualquer uma das mensagens pelo nó n_2.

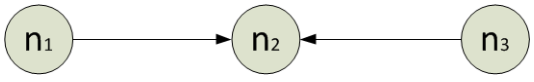


Figura 1.1: Problema do terminal oculto.

[2]Escuta ociosa ocorre quando um nó sensor aguarda pelo recebimento de pacotes mas não há transmissões em andamento.

[3]Recepções inúteis ocorre quando há escutas de pacotes destinados a outros nós.

1.2 Definição do Problema

A monitoração de cenários com dezenas ou centenas de sensores produz grande quantidade de dados que podem sofrer agregações durante a coleta e precisam ser escoados através de transmissões de rádio pela rede, até um ou mais destinos específicos. Tratando-se de uma RSSF, cujo consumo de energia é crucial para a vida útil da rede, o projetista deve considerar a operação dos sensores de forma a minimizar o gasto de energia na operação da RSSF. Deve, então, ser avaliado o uso de múltiplos sinks e de transmissões rádio livre de colisões.

No cenário proposto neste trabalho, considere uma RSSF onde os nós estão aleatoriamente inseridos em uma área, de modo que cada nó possui uma demanda de transmissão d_i, e existam um ou mais nós sinks que devem receber estes dados. Considere a topologia da rede formada pela relação de vizinhança de acordo com o alcance de transmissão dos rádios dos nós sensores. Considere ainda que cada nó pode descobrir sua vizinhança e definir o vizinho para o qual deve enviar sua demanda de forma que os dados cheguem ao nó sink escolhido através de alguma rota pela rede. Neste cenário nós podem produzir quantidades de dados não uniformes gerando demandas heterogêneas. Os nós precisam realizar transmissões de rádio proporcionais as suas demandas para escoar os dados para um dos sinks da rede.

O problema que se deseja estudar é como escoar esta demanda dos nós até os sinks de forma que não ocorram colisões nas transmissões, escutas ociosas ou recepções inúteis.

Deseja-se usar um mecanismo distribuído que utilize apenas informações locais, sem a existência de um agente central na rede e sem a necessidade de identificação global dos nós. Outro problema relacionado é a definição e encorporação no funcionamento do mecanismo da escolha do caminho dos dados até o sink desejado. A Figura 1.2 ilustra uma rede com dois sinks e vários nós sensores com demandas de pacotes a serem transmitidas. Como escoar esta demanda sem colisões e apenas com informações locais?

O escopo deste problema, integralmente ou parcialmente, tem sido estudado em trabalhos relacionados a mecanismos e protocolos MAC baseados em escalonamento livre de colisões, como nas referências [6–12], discutidas em mais detalhes no Capítulo 2. Entretanto, pode-se verificar nos trabalhos científicos um número reduzido de estudos que se referem a redes de sensores com múltiplos sinks com demandas variadas e pré-estabelecidas, o que é uma das abordagens desta dissertação.

Assim, este trabalho propõe a apresentação de duas versões de um mecanismo que tem como entrada as informações do cenário e problema supracitados, e como saída, o conhecimento local em cada nó de uma quantidade de slots que forma um quadro periódico de escalonamento de transmissões, onde o nó conhece os slots em

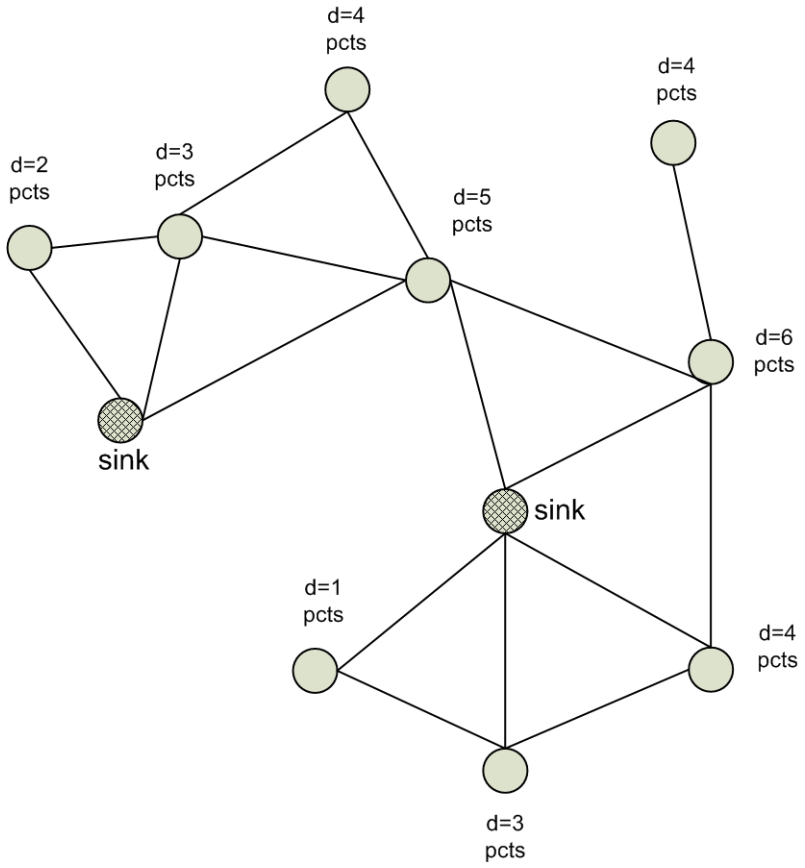


Figura 1.2: Representação em grafo de uma rede de sensores com múltiplos sinks e nós fontes com demandas a serem transmitidas, livre de colisões.

que deve transmitir de modo que possa escoar sua demanda da rede livre de colisões nas transmissões e escutar as transmissões apenas quando necessário.

1.3 Contribuições deste Trabalho

Esse trabalho possui as seguintes contribuições:

1. A primeira contribuição desta tese é a apresentação de duas versões de um algoritmo distribuído baseado em reversão de arestas para escoar qualquer demanda de dados em uma rede de sensores sem fio, livre de colisões, com múltiplos sinks e sem a necessidade de identificação global dos nós. Algoritmos de reversão de arestas em grafos, entre outras aplicações, são utilizados para atribuir escalonamentos de acesso entre processos computacionais a recursos diretamente compartilhados entre eles. A proposta deste trabalho faz uso destes fundamentos em ambas as versões do algoritmo apresentado. Uma versão escalona com o objetivo de que todos os vizinhos de um nó escutem sua transmissão, enquanto a outra versão realiza um escalonamento de enlace onde apenas um vizinho específico escuta e recebe a transmissão de um nó.

2. A avaliação experimental através de simulações dos mecanismos propostos sobre diversos parâmetros.

3. O trabalho apresentado em [13], como será visto no Capítulo 3, trata de um algoritmo de escalonamento baseado em reversão de arestas que utiliza conhecimento global para escalonar o acesso a recursos compartilhados e em frequências não uniformes. Uma segunda contribuição desta tese é a apresentação de uma extensão desta proposta através de um algoritmo distribuído que determine o escalonamento utilizando apenas conhecimento local. Além disso, é apresentado um algoritmo distribuído para a inicialização da dinâmica do algoritmo de reversão de arestas que evita condições de *deadlocks* previstas em um importante teorema apresentado em [13].

1.4 Estrutura da Tese

A apresentação do texto desta tese está feita da seguinte forma: no próximo capítulo há uma breve apresentação de trabalhos relacionados à agregação de dados, escoamento de dados e a protocolos MAC baseados em escalonamento livre de colisões. Também é explicada a inserção deste trabalho no contexto dos trabalhos relacionados. Em seguida, o Capítulo 3 apresenta dois algoritmos que utilizam a reversão de arestas para o compartilhamento de recursos, assim como a apresentação de dois outros algoritmos que realizam orientação acíclica em sistemas distribuídos, etapa obrigatória no funcionamento dos algoritmos de escalonamento por reversão de arestas. Prosseguindo, o Capítulo 4 descreve o funcionamento das duas versões do mecanismo de escalonamento proposto e apresenta passo a passo o funcionamento dos algoritmos. Também descreve o algoritmo distribuído de inicialização que evita *deadlocks* no escalonamento por reversão de arestas e ainda o algoritmo distribuído que utiliza informações locais para obtenção do escalonamento. O Capítulo 5 apresenta o ambiente construído no simulador de redes OMNET++ versão 4.2 para a verificação do funcionamento dos mecanismos propostos e avaliação de seu desempenho. Por fim, no Capítulo 6 estão expostas as conclusões gerais dos resultados obtidos e as considerações sobre possíveis trabalhos futuros.

Capítulo 2

Agregação, Escoamento e Escalonamento em RSSF

2.1 Agregação de Dados

Agregação de dados, fusão de dados, fusão de dados de múltiplos sensores, são alguns dos termos empregados para descrever tópicos relativos a questões dos processos, sistemas, infra-estruturas, algoritmos, ferramentas e métodos que visem produzir informações em quantidade e qualidade, em princípio, superiores às obtidas por uma única fonte de dados [14, 15].

O conceito de agregação de dados é o de combinar informações de origens diferentes para obter uma nova informação ou uma informação mais precisa do que a de uma única fonte. Os seres humanos e os animais desenvolveram a habilidade de usar múltiplos sentidos para auxiliá-los a sobreviver. Por exemplo, utilizar somente o sentido da visão pode não assegurar que uma determinada substância é comestível. A combinação da visão, tato, olfato e paladar fornecerá uma informação mais completa sobre a substância [1].

Em cada aplicação em rede de sensores na qual é desejável a agregação de dados, é necessária uma análise das entidades sensoriadas, das inferências desejadas e de um conjunto de desafios para sua efetivação. Entre estes fatores está a escolha da arquitetura e a definição dos protocolos para o funcionamento da infra-estrutura da rede. Especial atenção deve ser dada a escolha dos algoritmos de roteamento e de acesso ao meio para que a agregação de dados possa, através da correlação entre os dados e da capacidade de processamento, minimizar a redundância das informações originadas nos sensores e, por consequência, reduzir a carga na rede.

Independente do protocolo de roteamento e de acesso ao meio, as estratégias utilizadas no projeto de uma RSSF não podem se abster de avaliar o custo da agregação de dados. Para algumas aplicações este custo pode ser praticamente desconsiderado,

mas para aquelas em que a agregação requer operações complexas e processamentos contínuos, este custo deve ser muito bem avaliado. Estudos realizados mostram que o processo de agregação de imagens, por exemplo, custa dezenas de Joules por bit, o que é da mesma ordem que o custo de comunicação reportado na literatura [16, 17].

Uma das principais características das aplicações de agregação de dados é que seu custo afeta diretamente as decisões de roteamento. Na verdade, o projeto da rede tem que considerar o custo das agregações e o custo das transmissões para minimizar o consumo de energia. Assim, fica claro que a preocupação no consumo de energia nestas aplicações devem estar centradas em dois focos distintos: o custo de transmissão na rede de sensores sem fio e o custo inerente ao processo de agregação. A seguir, dois relevantes trabalhos que tratam do custo da agregação são apresentados.

"Routing Correlated Data with Fusion Cost in Wireless Sensor Networks"[16]

O trabalho de Hong Luo et al. [16] considera um *grid* onde os nós sensores estão distribuídos como na Figura 2.1, e devem encaminhar seus dados para um nó sumidouro. Os autores apresentam o *Adaptive Fusion Steiner Tree (AFST)*, um algoritmo de roteamento que não apenas otimiza os custos de transmissão e de fusão, mas também auxilia os nós sensores nas decisões de agregação. Baseado na avaliação dos benefícios da fusão para a rede com a análise dos custos de transmissão e fusão, e a estrutura da rede, o AFST toma decisões em tempo real de fusão para os nós roteadores durante o processo de construção da rota.

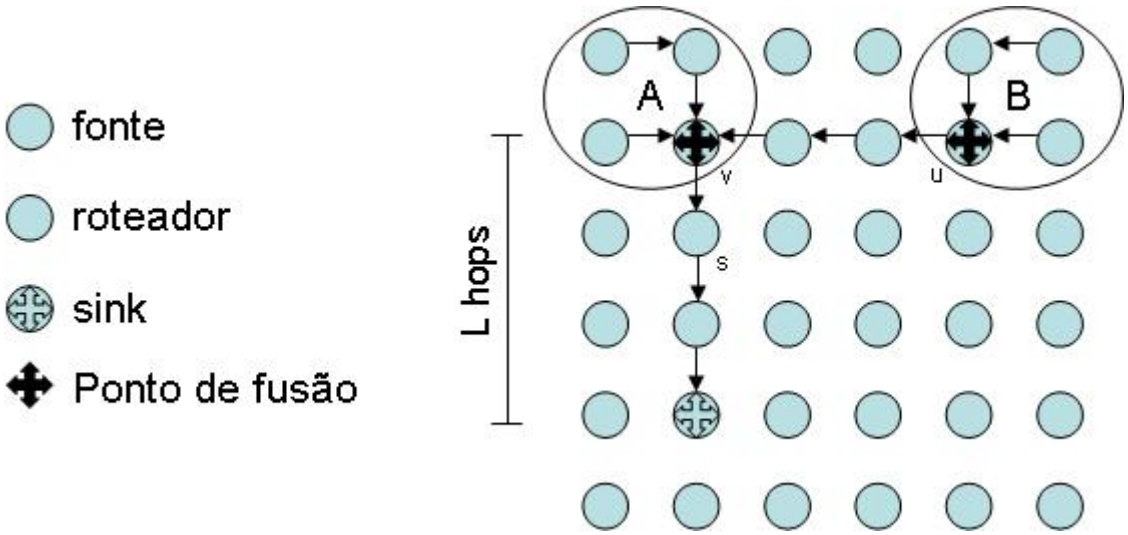


Figura 2.1: Exemplo de grid de agregação de dados

As setas formam a árvore de agregação em que os nós u e v agregam dados das áreas A e B, respectivamente. O nó v ainda agrega os dados oriundos de u antes de enviar seus bits para o sink. Considerando que cada nó tem o mesmo custo de

transmissão c_0, o custo total é linear com a quantidade total de dados recebidos, e o custo da fusão interna q_0. Define-se como $\omega(u)$ e $\omega(v)$ a quantidade de dados, respectivamente, em u e v antes da agregação entre eles. A quantidade de dados agregados resultante em v pode ser expressa pela Equação 2.1, onde σ_{uv} representa a razão de redução de dados resultante da agregação:

$$(\omega(u) + \omega(v))(1 - \sigma_{uv}) \tag{2.1}$$

Neste cenário, se v realiza a fusão de dados, a energia total consumida na rota de v até o sumidouro, considerando L saltos entre os nós, será dada pela Equação 2.2:

$$Lc_0(\omega(u) + \omega(v))(1 - \sigma_{uv}) + q_0(\omega(u) + \omega(v)) \tag{2.2}$$

Por outro lado, se v não executa a fusão de dados, o total de energia consumida na rota é dado apenas a utilizada nos custos de roteamento, dado pela Equação 2.3:

$$Lc_0(\omega(u) + \omega(v)) \tag{2.3}$$

Considerando o consumo total de energia da rede, v não deve executar a fusão de dados se:

$$\sigma_{uv} < q_0/Lc_0 \tag{2.4}$$

Esta é a teoria básica do AFST, proposta em [16].

"The Impact of Data Aggregation in Wireless Sensor Networks"[18]

Krishnamachari et al. [18] apresenta uma análise do impacto da agregação de dados sobre uma RSSF no que se refere ao consumo de energia. A abordagem de roteamento na rede leva em consideração a possibilidade de nós situados no caminho para o sink realizarem a consolidação de dados redundantes enviados por outros nós, ainda mais distantes em relação ao nó sink. O trabalho utiliza uma abordagem centrada em dados e não em endereços, como seria o caso em que os nós fontes enviam seus dados pelo caminho mais curto para o sink. Três esquemas de agregação de dados são comparados dentro da abordagem centrada em dados:

- Centro na Fonte Mais Próxima (CNS - *Center at Nearest Source*) - neste esquema todas as fontes enviam seus dados diretamente para a fonte mais próxima do sink, que então envia a informação agregada para o sink;
- Árvore de Menor Caminho (SPT - *Shortest Paths Tree*) - cada fonte envia sua informação para o sink mais próximo, e os caminhos sobrepostos dos nós para

o sink formam a árvore de agregação;

- Árvore Incremental Gulosa (GIT - *Greedy Incremental Tree*) - o primeiro passo da árvore de agregação consiste de apenas o menor caminho entre o sink e a fonte mais próxima. Após cada passo a próxima fonte mais próxima da árvore de agregação já formada é conectada a ela.

Os resultados obtidos mostram que é considerável o ganho de energia da agregação de dados particularmente quando há vários nós fontes próximos entre si mas que estão distantes do nó sink. Também é relevante neste trabalho a exposição do *trade off* entre latência e agregação de dados, o que ocorre quando um nó fonte reune dados enviados por vários outros aguardando a chegada dos mesmos em diferentes instantes. Somente depois da chegada de todas as mensagens o nó executa a agregação e reenvia a mensagem resultante para o próximo nó na árvore de agregação, causando um atraso proporcional ao número de saltos até o sink.

2.2 Escoamento de Dados

Inicialmente, o emprego de RSSF era comumente baseado no paradigma de comunicações de múltiplas fontes com um nó sumidouro responsável por receber todos os dados sensoriados pelos nós da rede. Recentemente, entretanto, cenários com múltiplos sinks estão sendo propostos com frequência cada vez maior, como no caso de redes que possuem nós com função de atuador em determinado monitoramento. Assim, novos algoritmos para o paradigma *múltiplas fontes - múltiplos sinks* estão sendo propostos, tendo em vista a ineficiência das soluções para cenários com apenas único sink quando aplicadas a cenários com múltiplos sinks. A seguir, dois trabalhos relevantes com soluções para múltiplos sinks são apresentados.

"Data Collection with Multiple Sinks in Wireless Sensor Networks"[19]

Um dos desafios da coleta de dados em RSSF é a ocorrência de interferência rádio que pode impedir que sensores próximos transmitam seus pacotes simultaneamente. Chen et. al [19] propuseram duas soluções para o problema da coleta de dados com múltiplos sinks: um algoritmo de aproximação para minimizar a latência no escalonamento da coleta de dados em um grafo que considera a demanda e a capacidade do canal e um algoritmo heurístico baseado no algoritmo de Busca em Largura, BFS (*Breadth First Search*).

O algoritmo de aproximação é composto de três componentes:

- a primeira componente constrói um *grafo expandido no tempo* que representa a progressão dos estados da rede ao longo de um período de tempo T, durante o qual a coleta dos dados ocorre, conforme visto na Figura 2.2 retirada de [19];

- a segunda componente gera os fluxos possíveis para o envio de dados através do grafo de tempo expandido considerando as interferências que podem ocorrer;
- a terceira componente define para cada slot de tempo o escalonamento para a coleta dos dados, conforme pode ser visto em outro exemplo, na Figura 2.3 (figura retirada de [19]).

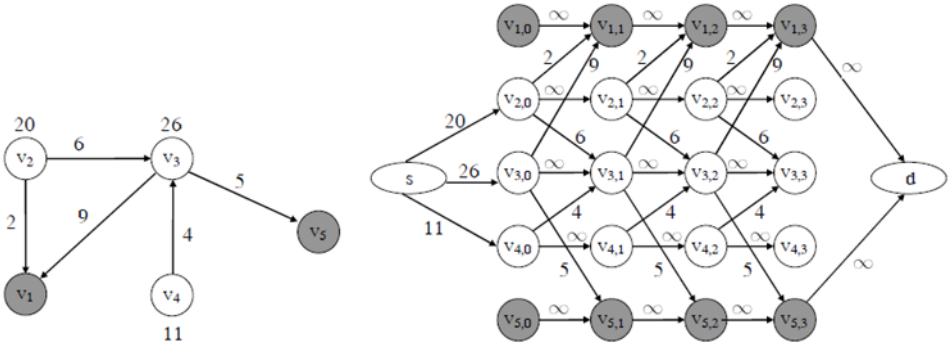
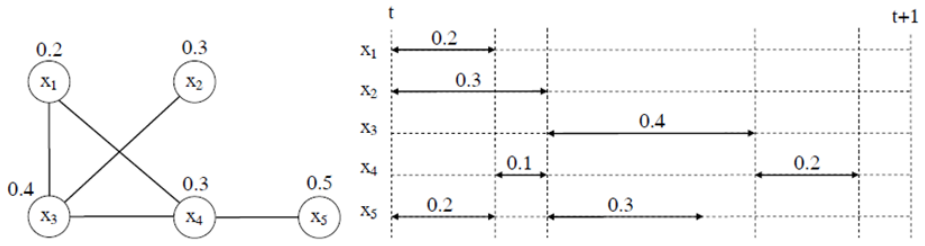


Figura 2.2: O grafo G, representativo da rede, e seu correspondente grafo estendido no tempo, G_T, onde $T = 3$ e os nós escuros representam os sinks.

O algoritmo se baseia na razão *fluxo de dados* e *capacidade do enlace* para definir a porção do slot de tempo que será usado por cada nó e um grafo de interferência para evitar colisões. O nó deve compartilhar o slot com os nós que interferem em sua transmissão, e assim, a mesma razão *fluxo de dados* e *capacidade do enlace* dos nós vizinhos é utilizada na definição do escalonamento. Apesar do algoritmo produzir um escalonamento livre de colisões que atende a demanda dos nós, o mesmo é centralizado e requer conhecimento global da rede das demandas.



(a) $G_{I_t} = (V_{I_t}, E_{I_t})$ (b) Data collection schedule

Figura 2.3: Outro exemplo, ilustrando a análise de interferência e o slot resultante (figura retirada de [19]).

O segundo algoritmo é baseado em uma heurística na qual os nós são alocados em camadas de acordo com sua distância ao nó sink mais próximo. Os nós fontes das camadas superiores, ou seja, com maiores distâncias dos nós sinks, são ordenados e transmitem seus dados para os nós da camada inferior. Desde que sua transmissão

não interfira em outra transmissão de um nó com maior precedência, os nós podem transmitir ao mesmo tempo. O conjunto de interferência é dado pela vizinhança de dois saltos. Ao final, toda a demanda da rede é enviada para os nós sinks.

"Data Acquisition on Multiple-Sink Sensor Networks"[20]

O trabalho realizado por Abhimanyu et al. [20], apresenta um modelo para adaptar os algoritmos utilizados para coletar dados em uma rede com apenas um sink para redes com múltiplos sinks. Adicionalmente, o artigo mostra analiticamente a economia de energia esperada em uma rede cuja coleta de dados é feita por múltiplos sinks. Neste modelo, o grafo de conectividade da RSSF com múltiplos sinks é mapeado para um grafo lógico equivalente, onde todos os sinks originais do grafo G são representados por único um sink virtual do grafo $G^{'}$. Assim, as arestas que conectam um sensor a um sink em G são mapeadas para o sink virtual em $G^{'}$. As arestas possuem um peso que é mantido o mesmo para todos os nós que não se conectam com sinks, e para os nós conectados ao sink virtual esse peso é o mínimo. A partir desta construção pode-se adaptar várias estratégias existentes de coleta de dados, normalmente usadas em redes com somente um sink para ser aplicadas ao grafo lógico $G^{'}$. Do ponto de vista do consumo de energia, os resultados obtidos apresentam uma relação direta entre o aumento da economia de energia com o crescimento do número de sinks, visto que cada vez menos saltos são necessários para uma determinada informação ser coletada.

2.3 Escalonamento em RSSF - Controle de Acesso ao Meio

Protocolos de controle de acesso ao meio (MAC) tem sido extensivamente estudados nas comunicações sem fio. Já nos anos 70, o protocolo ALOHA [21] foi desenvolvido para a transmissão de pacotes em redes rádio UHF com o objetivo de realizar a comunicação entre vários campis e centros de pesquisa espalhados pelas ilhas do arquipélago do Havaí e o campus principal da Universidade do Havaí, localizado próximo a Honolulu. Posteriormente nos anos 90, o surgimento de redes locais sem fio (WLAN) renovou os interesses pelo desenvolvimento de protocolos MAC que pudessem otimizar o funcionamento destas redes. Atualmente, pode-se observar duas direções principais nos projetos de protocolos MAC para redes sem fio: protocolos baseados em *contenção* e baseados em *escalonamento*.

Os **protocolos baseados em contenção**, diferentemente dos protocolos baseados em escalonamento, geralmente permitem aos nós que iniciem suas transmissões em instantes aleatórios e que disputem o canal com seus vizinhos. Sem a necessi-

dade de compartilhar algum tipo de escalonamento, este modelo de acesso consome menos recursos de processamento (ex: sincronização de relógios) e é mais flexível na escalabilidade da rede. O principal desafio para os protocolos de contenção é a redução do consumo de energia causado pelas colisões, escutas desnecessárias e escutas ociosas.

Um dos exemplos de protocolo baseado em contenção é o Protocolo de Acesso Múltiplo com Sensoriamento da Portadora e Prevenção de Colisões (CSMA/CA), do inglês *Carrier-Sense Multiple Access with Collision Avoidance*. O CSMA/CA foi adotado no padrão de redes sem fio IEEE 802.11, entretanto seu uso em redes de sensores é inviável devido ao consumo excessivo de energia pela escuta ociosa.

Os **protocolos baseados em escalonamento** clássicos normalmente possuem uma autoridade central, como um ponto de acesso ou uma estação base, que regula o acesso ao meio através do *broadcast* de uma escala que especifica, quando e por quanto tempo, cada nó poderá transmitir no meio compartilhado. Um exemplo deste protocolo é o Acesso Múltiplo por Divisão do Tempo (TDMA, do inglês *Time Division Multiple Access*). Melhor visto na Figura 2.4, o TDMA divide o canal em slots de tempo individuais, que são agrupados em quadros. Em cada slot de tempo, apenas um nó está autorizado a transmitir. A autoridade central escala qual slot de tempo deve ser usado e qual nó está a ele associado. Esta decisão pode ser avaliada a cada novo quadro ou determinada para múltiplos quadros [22].

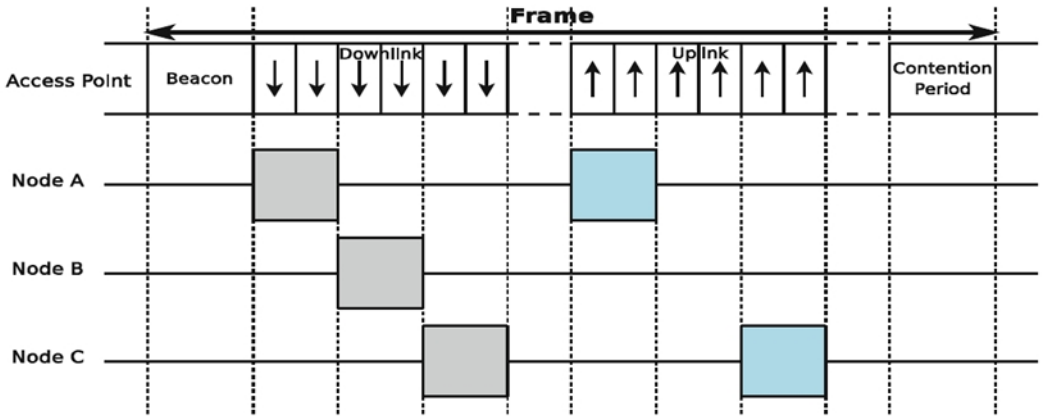


Figura 2.4: Comunicação TDMA: cada nó está associado a um determinado *slot* de tempo dentro do quadro. A comunicação é livre de colisões e possui baixo custo de energia.

O protocolo TDMA foi muito usado na telefonia celular, inclusive no Brasil. Uma das principais vantagens do TDMA é o baixo consumo de energia. Um nó sensor pode comutar o rádio para um modo de menor consumo de energia durante todo o período em que não está engajado em uma comunicação. Devido a necessidade da sincronização precisa entre os nós e a autoridade central, eles devem acordar exatamente no início do slot de tempo em que participam da comunicação, e para

isso existem algoritmos de sincronização. Outra questão para os protocolos TDMA é a alteração do número de nós da rede. Quando o número de nós é alterado o protocolo deve dinamicamente mudar seu tamanho de quadro e a designação dos slots de tempo.

Apesar da característica de evitar colisões e ser eficiente no consumo de energia, um dos desafios dos protocolos TDMA é operar em redes de sensores sem infra-estrutura, onde não há a figura de um nó central. É necessário ainda tratar problemas como a escalabilidade, a comunicação ponto-a-ponto e a interferência. De posse destes desafios e com o objetivo de atender aos requisitos inerentes a rede de sensores, diversas variantes do escalonamento TDMA são apresentadas em trabalhos científicos. Alguns destes trabalhos estão descritos a seguir e uma discussão destes com o presente trabalho é apresentado na última Seção deste Capítulo.

"Protocolo LMAC - Medium Access Protocol for Wireless Sensor Networks"[8]

Geralmente os projetistas de protocolos MAC não consideram aspectos inerentes ao hardware de transmissão no que se refere ao consumo de energia. O fato de um nó passar para o estado adormecido pode não significar efetivamente economia, uma vez que ao acordar, o restabelecimento do enlace consome uma quantidade adicional de energia e este chaveamento entre ligar e desligar o rádio, pode levar a redução do tempo de vida da rede. A ideia do protocolo LMAC [8] é levar em consideração estas propriedades da camada física no seu funcionamento. Este protocolo funciona basicamente da seguinte forma: cada nó pode requerer um slot do quadro para si e deve passar a ser o controlador deste slot. O nó só pode ser controlador de um único slot. Outros nós vizinhos podem requerer ao nó controlador o uso do slot mas é o controlador que indica qual a transmissão que ocorrerá no seu slot.

"Collision-free Time Slot Reuse in Multi-hop Wireless Sensor Networks"[10]

Van Hoesel et al. [10], utilizando como base no protocolo LMAC [8], apresenta um mecanismo para estabelecer escalonamentos livre de colisões, com reuso de *slots*, em RSSF com roteamento de múltiplos saltos. O mecanismo permite ao nó sensor escolher um intervalo de tempo para a transmissão, levando em consideração a vizinhança de dois saltos para que não ocorra interferência ou colisões com outras transmissões.

No algoritmo apresentado, à semelhança do LMAC, os nós escolhem um *slot* de tempo desde que não interfira em outras transmissões. O número de *slots* de tempo depende fortemente da máxima conectividade local (grau) na rede, designada por

Δ. O número de *slots* de tempo de um quadro deve ser adaptado de acordo com a densidade de nós da rede.

O mecanismo não tem a expectativa de usar o número mínimo de slots de tempo, pois visa permitir a adição de novos nós na rede ou mudanças de densidade devido a mobilidade. Os nós passam por diferentes estágios que resultam no nó entrar na rede, sincronizar, receber as informações da rede e transmitir.

Neste mecanismo colisões podem ocorrer quando dois ou mais nós escolhem o mesmo slot de tempo para controlar. Os nós que causam colisão não detectam a colisão, pois estão transmitindo quando elas ocorrem. Assim, é necessário que nós vizinhos informem posteriormente que houve a colisão, o que leva o nó a retornar para o estado inicial.

"Energy-Efficient, Collision-Free Medium Access Control for Wireless Sensor Networks"[6]

O protocolo TRAMA [6] apresenta um MAC que assume que o tempo é dividido em slots e que define qual nó transmite em um determinado slot, através de um esquema de eleição distribuída, baseada na informação de tráfego em cada nó. O protocolo é livre de colisões e permite ao nó comutar para o estado de menor consumo de energia quando não está transmitindo ou recebendo uma mensagem.

Os nós utilizando o TRAMA trocam suas informações de vizinhos de dois saltos e as escalas de transmissões especificando, em ordem cronológica, os pares de nós origem e destino de cada demanda. Com estes dados os nós definem qual nó transmite e qual recebe em cada slot de tempo. Basicamente o escalonamento é realizado com as informações da vizinhança de dois saltos e com as informações de tráfego da vizinhança de um salto.

O TRAMA considera um canal único formado por slots de tempo, que atende as transmissões de sinalização e de dados. A Figura 2.5 apresenta a organização destes slots. O tempo é organizado como períodos de acessos *aleatório* e *escalonado*, também referenciado como slots de *sinalização* e *transmissão*, respectivamente. O comprimento dos slots de transmissão é fixado baseado no largura do canal e no tamanho do pacote. Pacotes de sinalização são menores que os pacotes de dados e então os slots de transmissão são ajustados como múltiplos dos slots de sinalização para facilitar a sincronização.

Apesar da eficiência do TRAMA no consumo de energia, a necessidade de *broadcasts* de pacotes com informações do tráfego para adaptar o escalonamento e a complexidade do algoritmo de eleição produzem *overheads* significativos na rede. Para melhorar seu desempenho os autores estenderam o protocolo TRAMA com a formulação do *Flow Aware Medium Access* (FLAMA) [9]. O FLAMA evita o excesso de trocas de informações de tráfego empregando um algoritmo de eleição que atribui

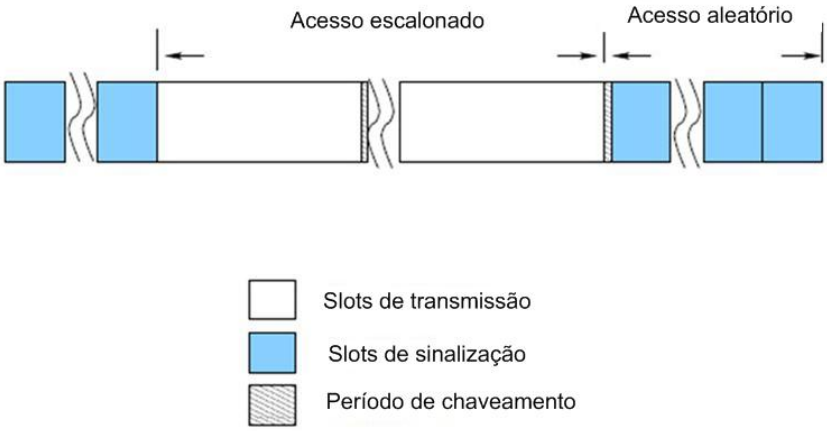


Figura 2.5: Organização do slot no TRAMA

pesos aos fluxos recebido e enviado pelo nó e estas informações passam a ser enviadas somente quando solicitadas. Apesar da redução do *overhead*, o procedimento de inicialização dos slots requer que o rádio permaneça ligado todo o tempo durante o período de acesso aleatório, elevando seu consumo de energia a valores similares aos do protocolo S-MAC [23], baseado em contenção.

"Link Scheduling in Sensor Networks: Distributed Edge Coloring Revisited"[11]

Gandham et al. [11] propõem um algoritmo distribuído para obtenção de um escalonamento MAC TDMA baseado no problema de coloração de arestas. O algoritmo é dividido em duas fases. Na primeira fase o algoritmo executa a coloração de modo que duas arestas incidentes no mesmo nó não podem ter a mesma cor associada. Para isto, atendendo a ordenação dada pelos IDs dos nós, um nó inicia a coloração de suas arestas somente após todos os seus vizinhos de 2 saltos, com IDs maiores que o seu, terem colorido suas arestas. Em seguida, o algoritmo atribui orientações aos enlaces de maneira a se estabelecer um escalonamento livre de colisões. Se isto não for possível, novos slots de tempo são alocados, até que o objetivo seja alcançado. A direção inicial das transmissões é definida de modo que não ocorram colisões, e a partir da primeira rodada, a reversão das arestas leva a uma nova configuração que também tem um escalonamento livre de colisões. O principal resultado deste trabalho é a minimização do número de cores utilizadas que é limitada a $\delta + 1$ cores, onde δ é o grau da rede, conferindo um escalonamento com alto grau de concorrência.

"Two ID-Free Distributed Distance-2 Edge Coloring Algorithms for WSNs"[12]

O trabalho de Pinho et al. [12] propõem dois algoritmos distribuídos e sem identificadores globais para escalonamento das transmissões baseados na coloração à distância-2 em grafos. O problema de determinar um escalonamento livre de colisões que permita recepções programadas é mapeado ao problema de coloração de arestas a distância-2. Neste cenário, as cores das arestas representam os intervalos de tempo de um protocolo MAC TDMA. Os dois algoritmos distribuídos, $Edge^3 - Sched$ e $Node^2 - Sched$, usam um mecanismo de eleição para determinar quais nós, em cada passo da execução, têm a permissão para colorir as arestas adjacentes ou colorir o nó, dependendo do algoritmo. Em resumo, os algoritmos funcionam da seguinte maneira: cada nó realiza o sorteio de um número e o envia para sua vizinhança de 2 saltos no grafo de conexões. Esta mensagem é enviada por meio de uma comunicação de múltiplos saltos. Os nós que tiverem um número de sorteio maior do que os recebidos de suas vizinhança de 2 saltos são denominados, naquele passo de execução, como *vencedores*, tendo o direito de colorir suas arestas adjacentes ou o próprio vértice, novamente dependendo de qual dos dois algoritmos está sendo usado. Uma vez que um nó tenha executado a coloração, o mesmo deixa de participar de futuras eleições, tomando parte no processamento apenas como retransmissor das mensagens do algoritmo. Ao final, cada nó na vizinhança de 2 saltos está colorido com uma cor diferente, garantido um escalonamento livre de colisões. Diferentemente do apresentado em [11] o escalonamento proposto em [12] considera a coloração das arestas, ou dos nós, a distância-2, utilizando possivelmente um maior número de cores do que as $\delta + 1$ cores propostas por Gandham et al., mas descartando a necessidade do uso de identificadores nos nós. Além disso, a abordagem proposta por Pinho et al. produz um número reduzido de mensagens de controle, o que impacta diretamente na economia de energia obtida, e consequentemente, na vida útil da rede.

2.4 Considerações Sobre os Trabalhos Correlatos Apresentados

Os trabalhos [16] e [18] mostram a importância da agregação de dados no que se refere ao consumo de energia. Destacam-se neles a necessidade do projetista da rede avaliar o custo local de processamento durante a agregação em relação ao custo do reenvio dos pacotes até o destino. Fica evidente a intuição de que, quanto mais distante do sink estão os nós que agregam seus dados, mais vantajosa poderá ser a agregação. Deve-se notar, entretanto, a ocorrência de latências decorrentes

das atividades de envio dos dados de uma vizinhança para um nó designado para agregar os dados, como no caso de *clusters*, e do próprio processamento da agregação. Dependendo da aplicação esta latência pode ser prejudicial ao objetivo da RSSF.

Sob o tema escoamento de dados, os trabalhos [19] e [20] apresentam algoritmos que funcionam nas redes com múltiplos sinks, que é uma demanda em RSSF para aumentar a eficiência de monitoramentos ou para atender a novas aplicações que surgem. O primeiro trabalho considera a relevância do impacto da latência no escoamento dos dados decorrente dos múltiplos sinks. Nele são levadas em conta a capacidade de armazenamento do nó (demanda) e a capacidade de transmissão no canal, cujas informações produzem o escalonamento final. Além disso, o algoritmo proposto é centralizado e requer o conhecimento de todas as informações sobre a rede. A idéia do trabalho de Chen et. al se assemelha a proposta desta tese, entretanto o funcionamento centralizado conduz a uma abordagem diferente das versões do mecanismo a serem apresentadas.

O segundo trabalho realiza uma interessante adaptação de algoritmos utilizados para redes com paradigma *múltiplas fontes - um sink* para o paradigma *múltiplas fontes - múltiplos sinks*. Analisa ainda, a economia de energia no que se refere a possibilidade de escoamento para um sink mais próximo, reduzindo o número de saltos ou transmissões necessárias, entretanto não se refere ao custo da energia para transmissões.

Algoritmos apresentados em [8], [10], [6], [11] e [12] são algoritmos que produzem mecanismos de acesso ao meio baseados em escalonamento, com constante preocupação com o consumo de energia em seus resultados. É interessante notar que estes algoritmos baseados em escalonamento realizam atividades cujos acessos ao meio ocorrem com contenções, onde colisões podem ocorrer. Entretanto, deve-se notar uma diferença: os trabalhos [8], [10] e [6] possuem acessos com contenção constantemente em cada início de slot para posteriormente gerar uma comunicação sem colisão, enquanto os dois últimos, [11] e [12], apenas realizam a contenção para a troca de mensagens de controle durante o processamento de definição de slots, passando em seguida para um regime nominal integralmente sem colisões.

O mecanismo proposto neste trabalho, apresentado em detalhes nos Capítulos 4 e 5, foi projetado para uma árvore de agregação de dados independentemente do processo de agregação escolhido. Neste caso, cada nó depende exclusivamente do número de pacotes que precisa transmitir, o que foi definido no Capítulo 1 como *demanda*.

Permeando a área do paradigma *múltiplas fontes - múltiplos sinks*, desenvolveu-se uma solução para contemplar a rede composta por um sink ou múltiplos sinks. De modo descentralizado e sem identificação global dos nós, os dados são escoados para um ou múltiplos sinks, regidos pela métrica desejada pelo projetista para a escolha

do sink de destino, como por exemplo, menor caminho ou caminho de menor custo.

Visto ainda que o problema de escalonamento é baseado no compartilhamento de recursos, aproveitando os sólidos fundamentos teóricos dos algoritmos SER e SMER apresentados no Capítulo 3, ao propor sua aplicação no espectro eletromagnético, obtém-se como resultado um escalonamento livre de colisões, a semelhança dos trabalhos apresentados na Seção 2.3 que realizam contenção apenas no processamento inicial dos algoritmos. Cabe ainda destacar, que a versão de escalonamento de enlace do mecanismo proposto pode permitir que nós vizinhos de um ou dois saltos transmitam simultaneamente, desde que atendida às condições que viabilizem tal transmissão, descrita no Capítulo 4.

Capítulo 3

Escalonamento por Reversão de Arestas

Os principais fundamentos teóricos necessários para a compreensão do mecanismo proposto neste trabalho referem-se a algoritmos de acessos a recursos compartilhados através da reversão de arestas de um grafo [24], [25], [13]. Para um melhor entendimento, dois destes algoritmos com variadas aplicações serão revistos neste Capítulo: o Escalonamento por Reversão de Arestas, do inglês *Scheduling by Edge Reversal* (SER) [26] [24], e o Escalonamento por Reversão de Múltiplas Arestas, do inglês *Scheduling by Multiple Edge Reversal* (SMER) [13].

Apesar do SER e do SMER[1] serem algoritmos que tratam do compartilhamento de recursos, de forma direta nenhum deles é capaz de lidar com questões inerentes ao compartilhamento do espectro eletromagnético.

O primeiro algoritmo, SER, foi projetado para que cada nó de um conjunto de processos acesse o mesmo número de vezes, ou em outras palavras, com a mesma frequência, cada recurso sendo compartilhado. Esta relação entre processos e recursos é representada de modo simples por um grafo G, conexo e não-dirigido, no qual cada nó representa um processo do sistema distribuído e uma aresta conecta dois nós, se e somente se, estes dois processos compartilham um recurso. Como o nome do algoritmo adianta, o escalonamento está baseado na orientação e reversão das arestas do grafo.

O segundo algoritmo, SMER, é uma generalização do SER, no qual o compartilhamento de recursos no sistema distribuído pode ser realizado pelos nós em diferentes frequências de acesso. A representação de funcionamento do algoritmo se dá sobre um multigrafo[2], que agora está baseado na reversão de múltiplas arestas, considerando o mesmo grafo G.

[1]Neste texto estão empregadas as abreviações *SER* e *SMER* originais do inglês, por serem as mais comumente utilizadas pela literatura científica.

[2]Um multigrafo é um grafo que permite mais de uma aresta entre o mesmo par de vértices.

Este Capítulo está dividido em 3 Seções. As duas primeiras apresentam detalhadamente a formulação e os conceitos por trás dos algoritmos SER e SMER. A última Seção apresenta dois algoritmos probabilísticos distribuídos para geração de orientação acíclica, uma importante etapa dos algoritmos de escalonamento por reversão de arestas, conforme visto adiante.

3.1 Escalonamento por Reversão de Arestas

O objetivo de um escalonamento de recursos é organizar o acesso de um conjunto de processos a um conjunto de recursos. Esta relação entre processos e recursos pode ser representada como um grafo G, conexo e não-dirigido, no qual cada nó representa um processo do sistema distribuído e uma aresta conecta dois nós, se e somente se, estes processos compartilham um recurso. Um recurso é representado em G como uma clique[3].

Durante o escalonamento o estado de um nó pode variar entre *ocioso* e em *operação*, estando neste último estado quando estiver usando um ou mais recursos compartilhados. Verifica-se que nós vizinhos no grafo não podem operar concorrentemente, pois estariam ambos utilizando, ao mesmo tempo, o recurso que eles compartilham. Em outras palavras, fica claro que, para que ocorra o compartilhamento de recursos a exclusão mútua é uma condição necessária.

Um exemplo que ilustra o compartilhamento de recursos é o Problema do Jantar dos Filósofos, de Dijkstra [27], ilustrado na Figura 3.1 e representado em forma de grafo na Figura 3.2. Nele, cinco filósofos estão sentados ao redor de uma mesa para comer espaguete. Um garfo é colocado entre cada par de filósofos adjacentes. Cada filósofo deve estar *pensando* ou *comendo*. Entretanto, ele só pode comer se possuir ambos os garfos, da direita e da esquerda do seu prato. O problema aqui consiste em aplicar um comportamento disciplinado, ou um algoritmo de concorrência, de modo que todos os filósofos tenham continuo acesso a comida, e possam seguir indefinidamente pensando e comendo.

O SER é um algoritmo distribuído que objetiva controlar, através do escalonamento, o acesso a recursos compartilhados entre os nós. Isto é feito através da ordem de ativação destes nós, de modo que dois processos que queiram compartilhar o mesmo recurso não sejam ativados ao mesmo tempo.

A ordem de ativação dos nós no SER é dada a partir de uma orientação acíclica inicial ω do grafo G, que determina a existência de pelo menos um nó sink[4], conforme visto na Figura 3.3. O nó sink é caracterizado por ter todas as suas arestas apon-

[3]Uma clique em um grafo G é um subgrafo de G que é completo.

[4]A designação de sink no SER e SMER não tem relação com o sink da RSSF, cujo conceito é inferido ao nó que recebe os dados de um monitoramento.

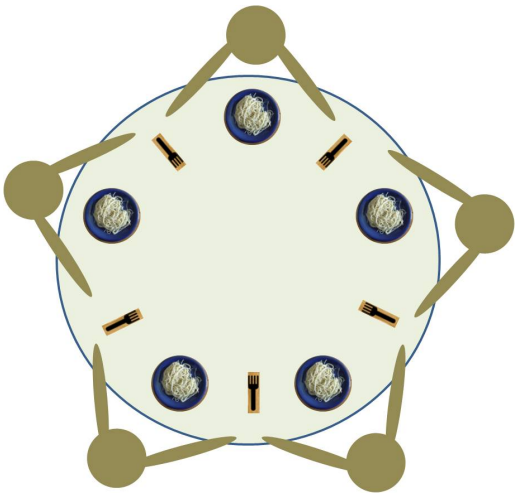


Figura 3.1: Problema do jantar dos filósofos.

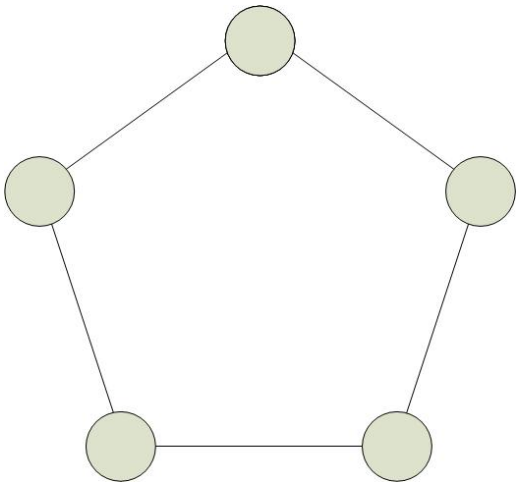


Figura 3.2: Representação em grafo do Problema do Jantar dos Filósofos.

tadas na sua direção. Neste momento, o nó utiliza todos os recursos (arestas) que compartilha, caracterizando sua *operação*, e então, reverte suas arestas, passando ao estado *ocioso*. O processo de orientação das arestas e de operação dos nós se repete, considerando que os nós operam continuamente e que precisam de todos os recursos que compartilham para operar. Este procedimento acontece apenas com a necessidade de conhecimento local e garante a não ocorrência de bloqueio permanente, ou *deadlock*, que caracteriza situações em que são criados impasses que impedem a continuação das execuções, como visto na Figura 3.4, e também de inanição, ou *starvation*, que caracteriza situações em que um grupo de processos não opera.

Além desta dinâmica simples mas poderosa, o SER ainda tem a propriedade de convergir para um período p pela repetição dos estados de orientação das arestas e de operação dos nós, o que proporciona a todos os processos o mesmo número de acessos aos recursos compartilhados dentro deste período.

O SER pode ser visto como uma dinâmica distribuída representada sobre o grafo G. O SER atua da seguinte forma:

- Iniciando a partir de uma orientação acíclica qualquer ω em G, sempre existe pelo menos um nó sink no grafo orientado, ou seja, um nó em que todas as suas arestas estão direcionadas para si;

- Todos os nós sinks devem operar enquanto os demais nós devem permanecer ociosos. Este procedimento garante a exclusão mútua a qualquer acesso aos recursos compartilhados pelos nós sinks;

- Após operar, o sink reverte a orientação de suas arestas, tornando-se fonte e permitindo que seus vizinhos tenham acesso aos recursos anteriormente sob

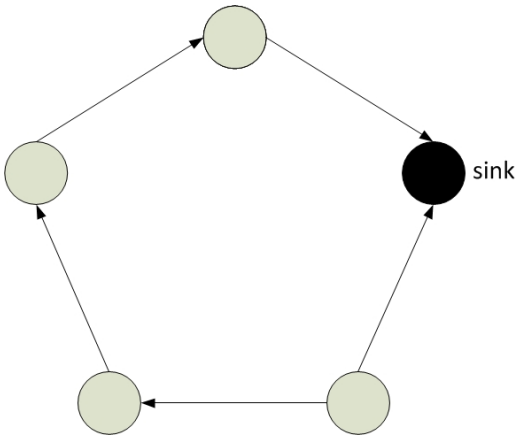


Figura 3.3: A existência de um nó sink é decorrente da orientação acíclica. Este é o nó que irá operar.

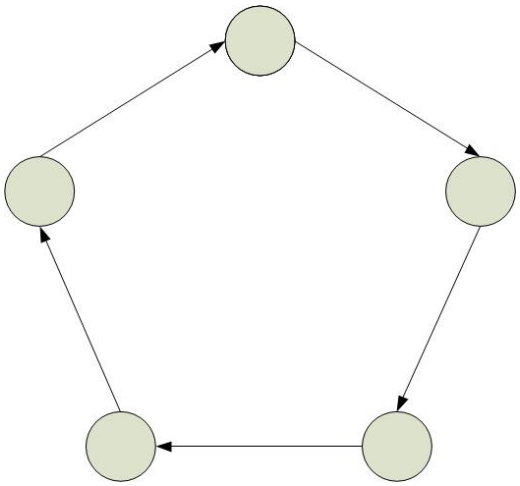


Figura 3.4: Como cada nó tem precedência sobre o vizinho, nenhum opera e está formado um *deadlock*.

seu controle. Este processo de reversão define uma nova orientação acíclica, e o processo é novamente repetido para o novo conjunto de sinks.

Considerando G um grafo finito, é de se destacar que, após um determinado tempo, um conjunto de orientações acíclicas se repete, definindo um período de comprimento p. Esta dinâmica simples garante que não ocorrerá *deadlock* ou *starvation*, uma vez que ela garante que a cada orientação acíclica sempre existirá pelo menos um nó sink, e em outras palavras, um nó em operação. É importante notar que, dentro de um período, todos os nós vão operar o mesmo número m de vezes e que, cada topologia de G tem seu conjunto particular de possíveis dinâmicas para o escalonamento por reversão de arestas. O sentido de tempo é definido pela própria operação da dinâmica SER, ou seja, o comportamento síncrono é equivalente ao caso onde cada nó em G leva uma quantidade de tempo idêntica para operar, e também uma quantidade de tempo idêntica para reverter suas arestas [28].

Um clássico exemplo da aplicação do SER é no problema do jantar dos filósofos sob alta carga [24]. Neste problema existem somente dois estados para os filósofos: *comendo* ou *com fome*. Este exemplo pode ser representado por um conjunto de processadores $\{P_1, ..., P_N\}$ de n, $n = |N|$, processos, nos quais cada um compartilha um recurso com o processador anterior e posterior a si, conforme definido no exemplo anterior. Partindo de uma orientação acíclica inicial ω, entre $n = 5$ nós em anel, a dinâmica SER resulta em um período de repetição $p = 5$ e de acesso ao recurso para cada nó de $m = 2$, conforme Figura 3.5. As propriedades do SER podem ser vistas em detalhes em [24].

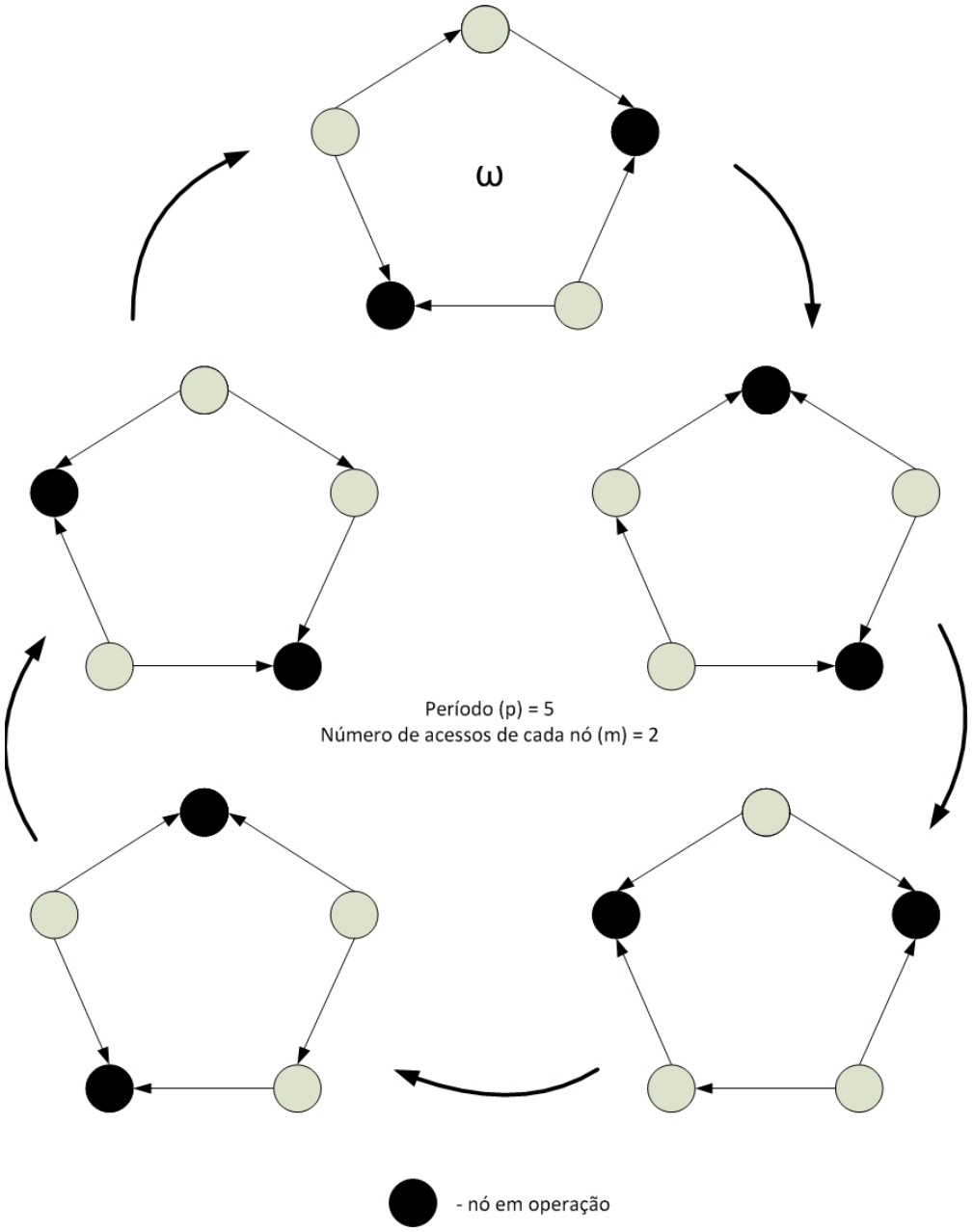


Figura 3.5: Dinâmica SER para o problema do jantar dos filósofos sob alta carga.

3.1.1 Definições do SER - Aspectos Estáticos

Seja $G = (N, E)$ um grafo conexo não dirigido, e seja $|N| = n$. Uma *orientação acíclica* de G é uma função da forma: $\omega : E \rightarrow N$ tal que nenhum ciclo não direcionado $i_0, i_1, ..., i_{k-1}, i_0$ exista para o qual $\omega(i_0, i_1) = i_1$, $\omega(i_1, i_2) = i_2$, ..., $\omega(i_{k-1}, i_0) = i_0$, isto é, nenhum ciclo dirigido é formado. Vamos denotar Ω como um conjunto de orientações acíclicas de G. Se ω é uma orientação acíclica de G, então alguns nós são tais que todas as suas arestas incidentes são dirigidas para eles. Estes nós são chamados *sinks*, e denotamos por *sinks(ω)* o conjunto de sinks induzidos por ω. Similarmente, os nós cujas arestas incidentes são orientadas para fora são

chamados *fontes*.

3.1.2 Definições do SER - Aspectos Dinâmicos

As seguintes definições conduzem ao conceito de escalonamento no SER: seja $\Gamma(\omega)$ o conjunto de orientações acíclicas que podem ser obtidas a partir de ω pela operação de vários subconjuntos não vazios de *sinks*(ω). Define-se um *schedule*, ou uma *escala*, como uma sequência de orientações infinitas $\langle \omega_k, k \geq 1 \rangle$ tal que $\omega_{k+1} \in \Gamma(\omega_k)$, $k \geq 1$. Um escalonamento genérico é denotado por σ. Do mesmo modo, Σ denota o conjunto de todas as escalas, e $\Sigma(\omega)$ o conjunto de todas as escalas que começam em ω. Uma escala é dita ser *gulosa*, ou *greedy*, se, para todo $k \geq 1$, ω_{k+1} é obtida a partir de ω_k pela reversão de todos os sinks. Σ_g denota o conjunto de todas as escalas gulosas e $g(\omega)$ a orientação obtida a partir de ω sob funcionamento guloso. A Figura 3.6 ilustra esta evolução de orientações acíclicas de G em tempo assíncrono. Ela mostra dois pares de orientações na forma $\omega \rightarrow \omega^{'}$ tal que $\omega^{'} = g(\omega)$. Em cada par, a decomposição de sinks de ω é S_0, S_1, S_2, e de $\omega^{'}$ é S'_0, S'_1, S'_2.

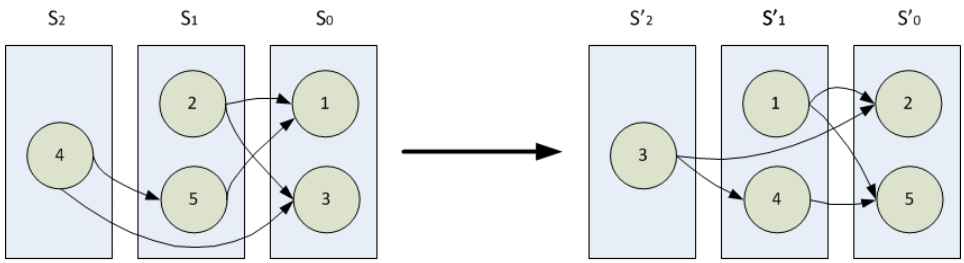


Figura 3.6: Exemplos de escalonamento por reversão de arestas

O lema a seguir estabelece a propriedade fundamental do escalonamento por reversão de arestas.

Lema 1 *Sejam ω e $\omega^{'}$ orientações acíclicas de G tal que $\omega^{'} = g(\omega)$. Um sink em $\omega^{'}$ tem, pelo menos um dos seus vizinhos, que é um sink em ω.*

3.1.3 Funcionamento do SER

Em um modelo assíncrono de computação distribuída, o escalonamento por reversão de arestas funciona da seguinte forma:

- Inicialmente, oriente G por uma orientação acíclica. Há vários métodos distribuídos na literatura para este procedimento;
- Nós que são sinks na orientação inicial são aqueles que tem o direito de operar primeiro. Pelo menos um nó é sink e então, reverte suas arestas incidentes.

No próximo pulso de sincronização, outro conjunto de sinks existirá, e assim segue;

- O escalonamento pode ser considerado como a evolução no tempo das orientações acíclicas de G. Como a orientação inicial é acíclica e tem pelo menos um sink, existe sempre um nó que pode operar.

Após a fundamentação teórica do SER, que trata do compartilhamento de recursos em frequências uniformes, Barbosa et al. [13] apresentam o compartilhamento de recursos em frequências não uniformes, através do Problema do Jantar dos Filósofos com Frequências, ou do inglês, *Dining Philosophers Problem with rates*, DPPr. Os autores apresentam duas soluções completamente distribuídas para o problema, sendo uma baseada na multiplicidade dos nós e outra baseada na multiplicidade de arestas. Esta última solução dá origem ao mecanismo do *Escalonamento por Reversão de Múltiplas Arestas* (SMER), apresentado a seguir.

3.2 Escalonamento por Reversão de Múltiplas Arestas

O *Problema do Jantar dos Filósofos com Frequências* propõe que o acesso aos recursos compartilhados pelos nós seja realizado sob diferentes frequências pré-determinadas. Similarmente ao SER, aqui também G é um grafo conexo não-dirigido no qual os nós representam processos, e as arestas, os recursos compartilhados entre os processos.

A análise da solução do DPPr é baseada na visão de G como um sistema de troca de mensagens, e sob este ponto de vista, o único meio de comunicação entre processos é enviando uma mensagem ponto-a-ponto através de canais de comunicação bidirecionais que conectam processos de acordo com a estrutura de G. Para simplificar a discussão, é adotado um modelo completamente síncrono de computação distribuída, no qual todos os processos são regidos por um relógio global comum, a computação local não gasta tempo e as mensagens enviadas entre vizinhos em um pulso tem suas entregas garantidas antes do próximo pulso. Assumir este modelo de computação distribuída permite simplificar algumas discussões, entretanto a solução pode ser estendida para o modelo assíncrono, o que pode ser visto na referência [13]. Por ora, as computações distribuídas em G acontecem em pulsos síncronos representados pelo inteiro $s \geq 0$.

O funcionamento da dinâmica de acesso aos recursos acontece basicamente da seguinte forma: a cada pulso síncrono um conjunto de processos tem garantido o acesso aos recursos compartilhados de que necessita. Em outras palavras, são todos

os recursos que compartilham com outros processos, conforme primitiva do DPP e DPPr. Tal conjunto compreende pelo menos um processo e não contém processos que são vizinhos de um salto em G. A seguir em 3.2.1, é apresentada a formalização do problema do DPPr.

3.2.1 Formalização do Problema do Jantar dos Filósofos com Frequências

- Seja N o conjunto de nós (ou processos) de G, com $n = |N|$;
- seja E o conjunto de arestas (recursos compartilhados) de G;
- Para $n_i \in N$ seja r_i um inteiro positivo e para $\{n_i, n_j\} \in E$ seja $g_{ij} = gcd(r_i, r_j)$; (obs: nota sobre gcd)[5].
- Para $s \geq 0$, DPPr requer uma sequência infinita $\langle S_s \rangle$ de subconjuntos de S_s de N a ser determinada que representa o conjunto de nós que tem acesso garantido aos recursos no pulso s.
- Se considerarmos qualquer subsequência finita de $\langle S_s \rangle$ ser chamada de uma *fase*, e que um nó n_i *aparece* em $\langle S_s \rangle$ no pulso s significa que $n_i \in S_s$, então a sequência $\langle S_s \rangle$ deve ser tal que:

  (i) S_s é um conjunto independente para todo $s \geq 0$.

  (ii) Para todo $\{n_i, n_j\} \in E$ tal que $r_i \geq r_j$ existe $s_0 \geq 0$ tal que n_i e n_j aparecem na sequência S_{s_0}, S_{s_0+1}, ... de acordo com uma sequência infinita de fases, cada uma tendo a seguinte propriedade geral: uma fase alterna $\lfloor r_i/r_j \rfloor$ ou $\lceil r_i/r_j \rceil$ aparições de n_j com uma aparição de n_i, até que tais aparições alternadas tenham acontecido r_j/g_{ij} vezes. A escolha entre $\lfloor r_i/r_j \rfloor$ ou $\lceil r_i/r_j \rceil$ deve ser tal que $\lceil r_i/r_j \rceil$ ocorre exatamente $(r_i \bmod r_j)/g_{ij}$ vezes, enquanto no restante das $(r_j - r_i \bmod r_j)/g_{ij}$ vezes, ocorre $\lfloor r_i/r_j \rfloor$.

A condição (i) representa a requisição da exclusão mútua no acesso aos recursos compartilhados, enquanto a condição (ii) incorpora as requisições de justiça (*fairness*) e vitalidade (*liveness*) quando inseridas no contexto do DDPr. Assim, é interessante analisar a condição (ii) de modo mais detalhado, uma vez que ela produz a interpretação do inteiro r de cada nó, dada a seguir. Para $\{n_i, n_j\} \in E$ e cada inteiro $K \geq 1$, existe $s > s_0$ tal que o número de conjuntos contendo n_i ou n_j em S_{s_0}, ..., S_s é $K(r_i + r_j)/g_{ij}$. Nesta sequência de conjuntos independentes, a

[5] *gcd* equivale ao MDC dos números entre parênteses.

razão do número de conjuntos contendo n_i para o número de conjuntos contendo n_j é r_j/r_i. Assim, a condição (ii) é um requerimento para que as taxas nas quais n_i e n_j apareçam na sequência de conjuntos independentes é proporcional a r_j/r_i.

Para $\{n_i, n_j\} \in E$, se denotarmos por p_{ij} o número de conjuntos independentes contendo n_i ou n_j para que a fase citada na condição (ii) seja completada, segue-se das observações feitas que:

$$p_{ij} = \frac{r_i + r_j}{g_{ij}} \tag{3.1}$$

3.2.2 Solução Baseada na Multiplicidade de Arestas

Uma das soluções do Problema do Jantar dos Filósofos com Frequências é através da transformação de G em um multigrafo, ou seja, um grafo onde é permitido múltiplas arestas entre dois nós. O número de arestas entre cada par de nós está relacionado com a ocorrência da frequência relativa desejada.

O multigrafo da solução do DPPr é designado por $M = (N'', E'')$, onde $N'' = N$ e para cada $\{n_i, n_j\} \in E$ existe e_{ij} arestas em E''. Similarmente ao SER, nesta solução também se recorre as orientações do multigrafo para indicar aqueles nós que tem acesso aos recursos que compartilham em cada pulso. Por outro lado, diferentemente do SER, aqui a noção de orientação acíclica de M fica sem significado, uma vez que ela requereria que todas as arestas entre dois nós estivessem sempre orientadas na mesma direção, o que tornaria o uso de múltiplas arestas sem sentido. Entretanto, como verificado mais a frente, a orientação acíclica de G tem um papel fundamental nesta solução.

O mecanismo que se forma nesta solução é tal que ele permite a $n_i \in N$ acessar aos recursos compartilhados se ocorrem, pelo menos, r_i arestas orientadas em sua direção, entre as e_{ij} arestas que ele compartilha com cada vizinho n_j.

Uma vez completado seu acesso aos recursos compartilhados, a orientação muda com a reversão de r_i arestas na direção de cada um dos seus vizinhos. Este mecanismo é chamado de Escalonamento por Reversão de Múltiplas Arestas, do inglês *Scheduling by Multiple Edge Reversal* (SMER). Assim como o SER, o SMER admite uma implementação extremamente simples e completamente distribuída que utiliza a troca de mensagens para indicar a reversão.

O ponto crítico do SMER é a definição do valor de e_{ij} e como orientar cada uma destas arestas de modo que a condição (i) e a condição (ii) sejam satisfeitas. Alguns limites dos valores de e_{ij} são bastante claros, como o fato de $e_{ij} \geq \max\{r_i, r_j\}$, pois se isto não ocorresse o nó com maior valor de r nunca estaria em um conjunto independente. Da mesma forma, para garantir que n_i e n_j nunca tenham acesso aos recursos compartilhados no mesmo pulso (tempo), tem-se que $e_{ij} \leq r_i + r_j - 1$.

Dado que μ_0, μ_1, ... é a sequência de orientações de M que o SMER produz a partir da orientação inicial μ_0, para $s \geq 0$, define-se que a_s^{ij} representa o maior múltiplo de g_{ij} que não excede o número de arestas orientadas a partir de n_i para n_j em μ_s. A soma $a_s^{ij} + a_s^{ji}$ é constante para $s \geq 0$, uma vez que estes dois termos variam com s transferindo um certo múltiplo de g_{ij} de um nó para outro. Esta soma é chamada de f_{ij} e toda orientação de M para a qual a soma de tais quantidades é f_{ij} é chamada *legal para* $\{n_i, n_j\}$ *dado* μ_0.

Baseados no grafo G e no multigrafo M, nos nós n_i e n_j e nos conceitos acima, dois teoremas, com provas em [13], apresentam as condições para a solução do problema do DPPr. Se a partir de G e M for definido que, respectivamente G^{ij} e M^{ij} são induzidos pelo par de vizinhos n_i e n_j, e que μ_0^{ij}, μ_1^{ij}, ... representa a sequência de orientações de M^{ij} produzida pelo SMER a partir de μ_0^{ij}, então o Teorema 1 define o valor de e_{ij} a ser utilizado no mecanismo SMER.

Teorema 1 *Se* $\max\{r_i, r_j\} \leq e_{ij} \leq r_i + r_j - 1$ *então o uso do SMER a partir de* μ_0^{ij} *em* M^{ij} *resolve a instância do DPPr dada por* G^{ij}, r_i *e* r_j, *se e somente se* $f_{ij} = r_i + r_j - g_{ij}$. *Neste caso, a sequência* μ_0^{ij}, μ_1^{ij}, ... *inclui todas as orientações de* M^{ij} *que são legais para* $\{n_i, n_j\}$ *dado* μ_0^{ij}.

O Teorema 1 tem como consequência que $e_{ij} = r_i + r_j - g_{ij}$ é um número necessário e suficiente de arestas em M^{ij}. Apesar do Teorema 1 expor o funcionamento do SMER para os nós n_i e n_j é necessária a generalização desta condição para todo o grafo M. Essa generalização tem que considerar a orientação destas arestas e_{ij} de modo que a solução do DPPr exista. Para tal, consideremos o conjunto K de ciclos simples não-dirigidos em G. Para $k \in K$, tem-se que $n_i \in k$ significa que o nó n_i é um dos nós de k, e $\{n_i, n_j\} \in k$ indica que a aresta $\{n_i, n_j\}$ é uma das arestas de k. Seja a Equação 3.2, e para $s \geq 0$ seja a Equação 3.3. k^+ e k^- são as duas direções transversais possíveis de k, e $(n_i, n_j) \in k^+$ indica que, quando percorrendo k na direção de k^+, o nó n_i é encontrado primeiro quando atravessando a aresta $\{n_i, n_j\}$.

$$\rho(k) = \sum_{n_i \in k} r_i \tag{3.2}$$

$$\sigma_s(k) = \max \left\{ \sum_{(n_i,n_j) \in k^+} a_s^{ij}, \sum_{(n_i,n_j) \in k^-} a_s^{ij} \right\} \tag{3.3}$$

Da Equação 3.3 temos que $\sigma_s(k)$ é o número de arestas do multigrafo em k orientadas por μ_s na direção transversal cuja a maioria das arestas está orientada. Como $\sigma_s(k)$ é invariante no SMER, visto que a reversão de arestas por qualquer um dos nós em k soma ou subtrai o mesmo número de arestas em ambas as direções

transversais, no lugar de $\sigma_s(k)$ pode-se ter $\sigma(k)$. Para $s \geq 0$, toda a orientação μ_s de M provoca uma orientação em G, da seguinte forma: para $\{n_i, n_j\} \in E$, afirma-se que $\{n_i, n_j\}$ está orientada na direção de n_j se e somente se $a_s^{ij} \geq r_j$. Tal orientação é denominada $\omega(\mu_s)$. De acordo com $\omega(\mu_s)$ os sinks são os nós que tem acesso aos recursos compartilhados, similarmente como no problema do DPP visto em 3.1, resolvido pelo SER. Entretanto, cabe destacar que no SMER, diferentemente do SER, após ser sink em um pulso um nó não necessariamente se transforma em um nó fonte no pulso seguinte, e nem existe a garantia de que a aciclicidade de $\omega(\mu_s)$ está mantida em $\omega(\mu_{s+1})$, e por isto são necessárias as condições do Teorema 2 para garantir o funcionamento do SMER.

Teorema 2 *O uso do SMER a partir de μ_0 resolve a instância do DPPr dado por G e $r_1, ..., r_n$ se e somente se $\sigma(k) < \rho(k)$ para todo $k \in K$.*

Em outras palavras, o uso do SMER resolve a instância do DPPr se a soma das reversibilidades $\rho(k)$ é maior que o valor máximo da soma das arestas em uma das direções de um ciclo $\sigma(k)$, dada pela orientação determinada para as arestas do multigrafo.

Ainda pelo Teorema 2 vemos que qualquer orientação de M que resulta em $\sigma(k) < \rho(k)$ para todo $k \in K$ é uma boa orientação inicial para o SMER. Por outro lado, uma orientação inicial de M na qual $\sigma(k) \geq \rho(k)$ eventualmente pode levar o SMER a falhar, motivado pela existência de um ciclo dirigido em algum $s \geq 0$. Para evitar esta orientação inicial equivocada existe uma abordagem que determina a formação de uma boa orientação inicial $\bar{\mu}_0$ para M.

- Para todo $\{n_i, n_j\} \in E$, $\bar{\mu}_0$ orienta exatamente r_i arestas na direção de n_i se e somente se $r_i \geq r_j$.

Se em G existir um ciclo não dirigido k no qual $r_i = r_j$ para todo $\{n_i, n_j\} \in k$, então adicionalmente $\bar{\mu}_0$ deve ser tal que nenhum ciclo dirigido seja criado em $\omega(\bar{\mu}_0)$. O resultado é uma orientação que atende as requisições do Teorema 2.

3.3 Visão Geral de Demandas e Reversibilidades de Arestas

O grafo G é o grafo representativo de uma rede de sensores sem fio disposta em uma área limitada. A idéia por trás do mecanismo de escalonamento é realizar *operações* no grafo G, de modo que o SER e o SMER possam ser empregados no compartilhamento do espectro eletromagnético. Para atingir este objetivo, vale a

pena rever de modo mais prático, alguns conceitos do Capítulo 3, considerando a versão broadcast do algoritmo de escalonamento.

A Seção 3.2.2, que apresenta detalhadamente o SMER, descreve que a quantidade de arestas e_{ij} entre dois nós i e j, necessárias e suficientes para a aplicação do SMER no multigrafo M, derivado de G, é dada pela Equação 3.4, onde outro importante parâmetro visto, designado por r_i, é também chamado de *reversibilidade* do nó i.

$$e_{ij} = r_i + r_j - mdc(r_i, r_j) \tag{3.4}$$

Em um nó i, a reversibilidade r_i representa a relação de i com sua frequência aos recursos compartilhados no contexto global do grafo. É um número inteiro e positivo que estabelece o relacionamento entre os nós ou processos, no que se refere as frequências de acesso, aqui designada pela demanda d_i quando se trata da demanda do nó i. A partir da teoria detalhada na Seção 3.2.1, para se obter o valor da reversibilidade de um nó é necessário o conhecimento da frequência de todos os nós do grafo. O valor da reversibilidade de um nó i em um grafo com k nós, onde $i \in \{0, ..., k\}$, é dado por:

$$r_i = \frac{mmc\,(d_0, ..., d_k)}{d_i} \tag{3.5}$$

O exemplo da Figura 3.7 apresenta como dois nós calculam suas reversibilidades. O nó i requer a frequência de acesso, ou demanda, como está sendo chamado neste trabalho, dada por $d_i = 2$, enquanto o nó j tem a demanda dada por $d_j = 3$. A partir destas informações, o cálculo da reversibilidade é feito de acordo com a Equação 3.5:

$$r_i = \frac{mmc(d_i, d_j)}{d_i} = 3$$

$$r_j = \frac{mmc(d_i, d_j)}{d_j} = 2$$

A quantidade de arestas do multigrafo e_{ij} é calculada pela Equação 3.4:

$$e_{ij} = r_i + r_j - mdc(r_i, r_j) \quad \text{que resulta em:} \quad e_{ij} = 3 + 2 - 1 = 4$$

A solução para a operação[6] do nó i é a estabelecida na Seção 3.2.2, ou seja, i opera se ocorrem, pelo menos, r_i arestas orientadas em sua direção, entre as e_{ij} arestas que ele compartilha com cada vizinho j.

Na Figura 3.7 cada token circular representa uma aresta do multigrafo. Os quatro tokens representam as e_{ij} arestas de M e são colocados de acordo com uma orientação acíclica do grafo, atendendo ao Teorema 2, da Seção 3.2.2. Como neste

[6] A *operação* do nó neste problema significa a realização de uma *transmissão* rádio.

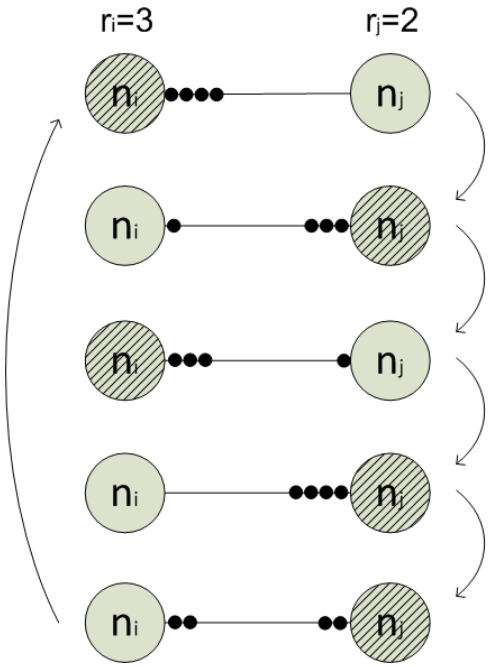


Figura 3.7: Funcionamento do SMER com dois nós i e j, com demandas de acesso de duas e três vezes, respectivamente.

grafo existem somente dois nós e não é possível a formação de ciclos, esta etapa é simplificada.

Conforme ilustrado na Figura 3.7, inicialmente o nó i é sink SMER[7] e por esta razão seu desenho está hachurado. Deve-se notar que após a operação do nó i são revertidas r_i arestas na direção do nó j, que torna-se sink SMER, e procedimento análogo ocorre após a operação de j. Após a operação do nó j no quinto estado do funcionamento do SMER um *período*, ou fase, é formada, pois ambos os nós retornam ao primeiro estado, ou seja, a orientação inicial μ_0 do multigrafo. É importante ressaltar, entretanto, que o período pode se formar repetindo alguma orientação μ_s, $s > 0$, posterior a orientação inicial do multigrafo.

A cada estado do grafo existe pelo menos um nó sink SMER, que é o nó que está *operando* ou *transmitindo*. Após o fechamento do período é possível conferir que os nós i e j operaram o número de vezes definido pelas demandas (e não pelas reversibilidades), no caso dois acessos para o nó i e três acessos para o nó j.

Após o exemplo didático da Figura 3.7, considere agora o exemplo da Figura 3.8, onde o grafo é representativo de quatro nós e das transmissões rádio entre eles, e as arestas indicam o vizinho destinatário da transmissão, ilustrando a versão unicast do mecanismo. Cada nó precisa transmitir para seu vizinho a quantidade de pacotes da demanda descrita sobre o desenho do nó e na direção das arestas do grafo. O nó n_s é o nó sink da rede, ou seja, o nó que somente recebe os dados enviados. Observe

[7]A notação *sink SMER* se refere a dinâmica estabelecida pelo algoritmo do SMER, e está sendo usada para diferenciar da notação *sink* que se refere ao sumidouro dos dados de um monitoramento.

que o nó n_s não transmite nenhuma demanda e por isso, não requer acesso ao meio eletromagnético, motivo pelo qual não participa do funcionamento do SMER.

A partir dos valores de demandas, os valores das reversibilidades dos nós encontram-se calculados sob o desenho de cada nó e o funcionamento se dá da seguinte forma: uma orientação acíclica inicial ω_0 em G resulta em obter o nó n_3 como um sink SMER. A partir de ω_0 uma orientação inicial μ_0 do multigrafo M é estabelecida de acordo com o Teorema 2.

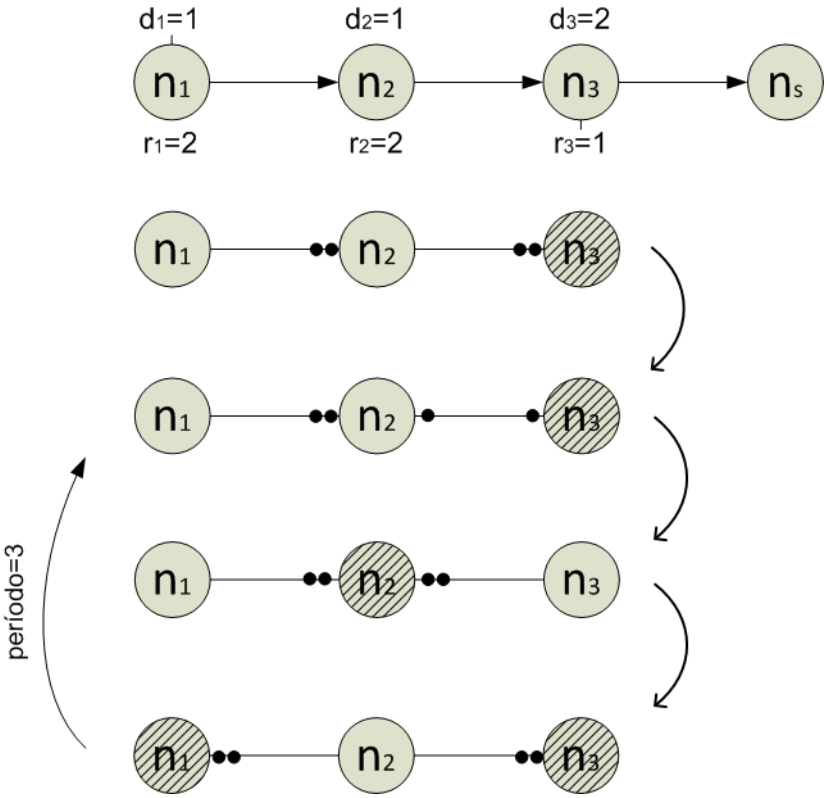


Figura 3.8: Modelagem de transmissões utilizando o SMER para definição dos slots de tempo em um grafo sem arestas de interferência.

A aplicação do SMER resulta em um período formado por três orientações do multigrafo, conforme ilustrado na Figura 3.8. Como esperado pela aplicação direta do SMER no escalonamento de acesso ao meio, consultando a terceira configuração do período formado e a orientação das transmissões no grafo G, verifica-se que n_1 transmite para n_2 no mesmo tempo em que n_3 transmite para n_s. Como n_2 também vai receber o transmitido por n_3, identifica-se a colisão no nó n_2 pelo problema do terminal oculto. Ao longo deste capítulo será visto como este problema pode ser resolvido no contexto do SMER. Em particular, o objetivo é não permitir que os nós n_1 e n_3 acessem ao recurso compartilhado simultaneamente.

A visão geral dada por estes dois exemplos apresenta um melhor entendimento do funcionamento do SMER e sua relação com o compartilhamento de recursos no contexto do acesso ao meio eletromagnético.

3.4 Algoritmos Probabilísticos Distribuídos para Geração de Orientação Acíclica

Uma das condições para o funcionamento do SER e do SMER é a necessidade do grafo G ter uma orientação acíclica inicial ω_0, de modo que haja a existência de pelo menos um nó sink, que entre em operação no inicio da dinâmica de escalonamento. Essa orientação inicial ω_0 representa o compartilhamento de recursos no instante inicial, ou ainda, a quantidade de concorrência que irá ocorrer. Verifica-se então que a eficiência do algoritmo que gera a orientação acíclica, particularmente sob o aspecto da concorrência, determina o funcionamento dos escalonamentos por reversão de arestas.

Dois algoritmos denominados *Alg-Arestas* e *Alg-Cor*, apresentados em [29] e [30], foram previamente analisados para a geração de orientação acíclica para sistemas de reversão de arestas, e serão utilizados neste trabalho. Ambos são executados sincronamente[8] e em seu funcionamento os nós são definidos como *probabilísticos* ou *determinísticos*. São chamados probabilísticos se estão participando do algoritmo, ou seja, se ainda possuem arestas incidentes não orientadas e continuam participando dos sorteios, e são chamados de determinísticos se não participam mais por terem todas as suas arestas incidentes orientadas. A seguir estão apresentados estes algoritmos.

3.4.1 Algoritmo Alg-Arestas

O funcionamento do Alg-Arestas é bastante simples. Em cada passo do algoritmo os nós probabilísticos lançam um dado de f faces, podendo ser obtido do sorteio números inteiros de 0 a $f-1$. O procedimento que orienta as arestas é o seguinte: uma aresta entre dois nós é orientada na direção daquele que tiver obtido o maior número no sorteio. Em caso de empate, a aresta permanece não orientada e seus nós adjacentes permanecem probabilísticos, participando da próxima iteração do algoritmo, onde é realizado um novo sorteio. Quando todas as arestas estão orientadas, o algoritmo termina. As Figuras 3.9 e 3.10 ilustram etapas do algoritmo e a formação de um nó sink no grafo orientado.

3.4.2 Algoritmo Alg-Cor

O algoritmo Alg-Cor tem o objetivo imediato de produzir concorrências que incrementem a geração de sinks durante o processo de orientação acíclica das arestas. O funcionamento também é bastante simples: os nós sorteiam um número e o nó

[8]Na verdade são algoritmos assíncronos, mas considerados aqui síncronos para facilitar seu entendimento.

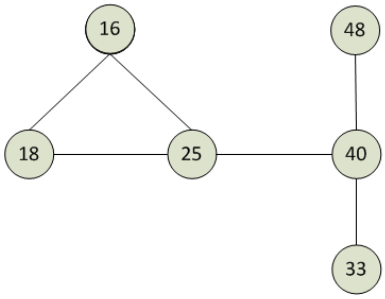


Figura 3.9: Exemplo do funcionamento do Alg-Arestas: sorteio do dado de f faces.

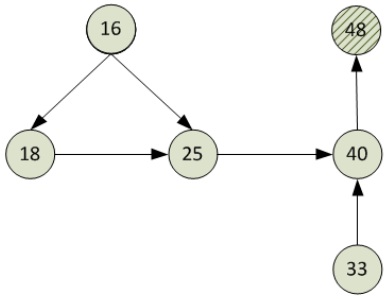


Figura 3.10: Exemplo do funcionamento do Alg-Arestas: grafo acíclico resultante.

que ganha o sorteio (maior número) em relação a seus vizinhos de um salto escolhe uma cor deterministicamente (número inteiro positivo qualquer). Esta cor não deve ser igual à de nenhum dos vizinhos já coloridos e deve ser a menor possível. Dessa forma, por exemplo, se dois vizinhos de um nó já estão coloridos com 0 e 2, o nó escolhe 1. Se já estão coloridos com 2 e 3, o nó escolhe 0. O nó vencedor informa aos vizinhos sua cor, e estes realizam novo sorteio. Também em caso de empates os nós realizam novo sorteio. Cada nó que for colorido orienta as arestas incidentes a vizinhos já coloridos na direção da maior para a menor cor. Quando todas as arestas estão orientadas, o algoritmo termina. As Figuras 3.11 e 3.12 ilustram etapas do algoritmo e a formação de três nós sinks.

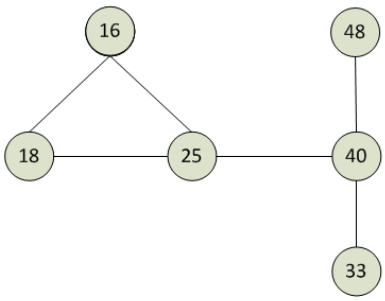


Figura 3.11: Exemplo do funcionamento do Alg-Cor: sorteio do dado de f faces. O vencedor escolhe a menor cor não existente na vizinhança.

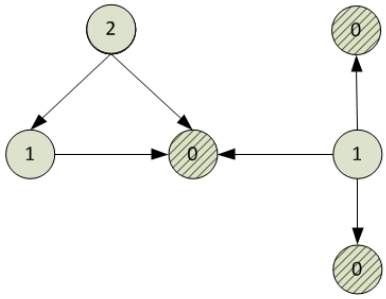


Figura 3.12: Exemplo do funcionamento do Alg-Cor: a orientação é feita da maior cor para a menor cor.

Capítulo 4

Escalonando Demandas de Transmissão Livre de Colisões Utilizando Reversão de Arestas

A partir dos fundamentos vistos nas Seções 3.1 e 3.2, apresentando o SER e o SMER, respectivamente, verifica-se o enorme potencial da reversão de arestas no campo do escalonamento de recursos compartilhados. Por outro lado, também verifica-se que o SER e o SMER não podem ser diretamente aplicados no compartilhamento do espectro eletromagnético, pois não tratam problemas inerentes à comunicação sem fio, como o problema do terminal escondido, apresentado na Seção 1.1.

Este capítulo apresenta duas versões de um mecanismo de escalonamento baseado em reversão de arestas. A primeira versão, chamada de ***broadcast***, produz um escalonamento que atende aos nós que possuem demandas a transmitir mas não necessariamente vizinhos destinatários específicos. Após uma troca inicial de mensagens para descobrir as demandas dos vizinhos, entre outras informações, os nós passam a operar de forma coordenada evitando qualquer colisão e atendendo às demandas. Nesta versão, todos os nós vizinhos escutam e recebem as transmissões de um nó, realizando uma difusão. Diferentemente da versão *broadcast*, na segunda versão, chamada de ***unicast***, os nós transmitem suas demandas para um nó vizinho específico. Desta forma, toda transmissão na rede envolve apenas dois nós: o transmissor e o receptor. Este cenário oferece condições para realizar transmissões simultâneas no espectro eletromagnético, desde que as transmissões não causem colisões nos diferentes destinatários.

Iniciando por uma visão geral, as próximas Seções descrevem as etapas da formação do mecanismo de escalonamento. Aos poucos, nomenclaturas e procedimentos são descritos até a obtenção dos resultados esperados.

4.1 Definições

4.1.1 Grafo de Conectividade

Considere G o *grafo de conectividade* resultante da descoberta dos nós vizinhos que representa a rede de sensores. Seja então:

- $G = (V, E)$ o grafo de conectividade da RSSF;
- V o conjunto de nós de G, com $|V| = n$;
- E o conjunto de arestas de G. Existe $(n_i, n_j) \in E$ se a distância entre um nó n_i e um nó n_j, dada por d_{ij} é tal que $d_{ij} \leq r_c$, onde r_c é o raio de comunicação do nó sensor;

O grafo G é conexo e não-dirigido e nele cada nó representa um nó sensor e uma aresta conecta dois nós, se e somente se, estes nós sensores estão dentro do raio de comunicação r_c um do outro, ou seja, possuem capacidade de transmissão e recepção de uma mensagem, conforme ilustrado nas Figuras 4.1 e 4.2.

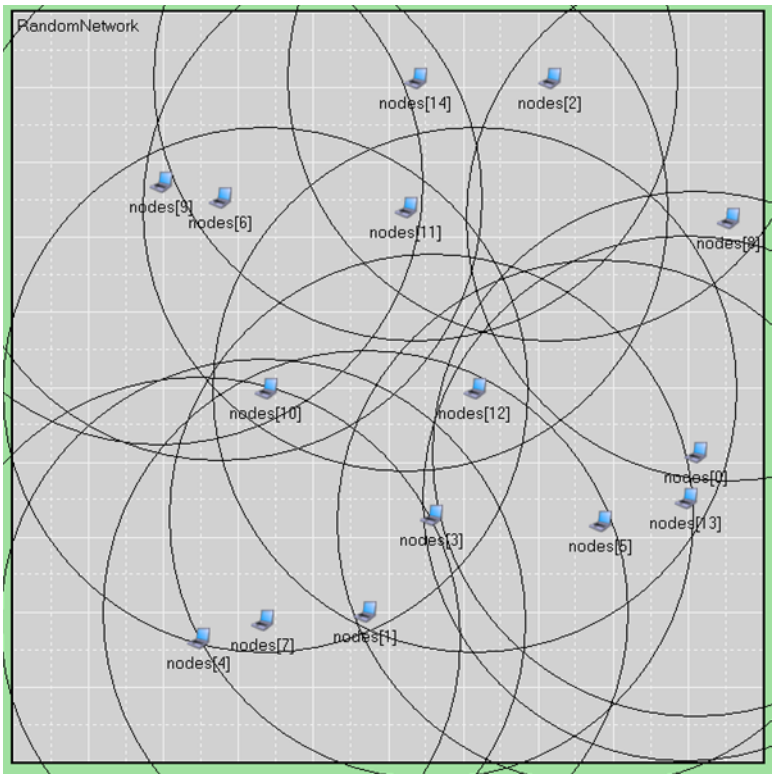


Figura 4.1: Os raios de comunicação dos nós definem as arestas do grafo G. Grafo com 15 nós.

As Figuras 4.3 e 4.4 ilustram os grafos de conectividade destas duas redes, respectivamente.

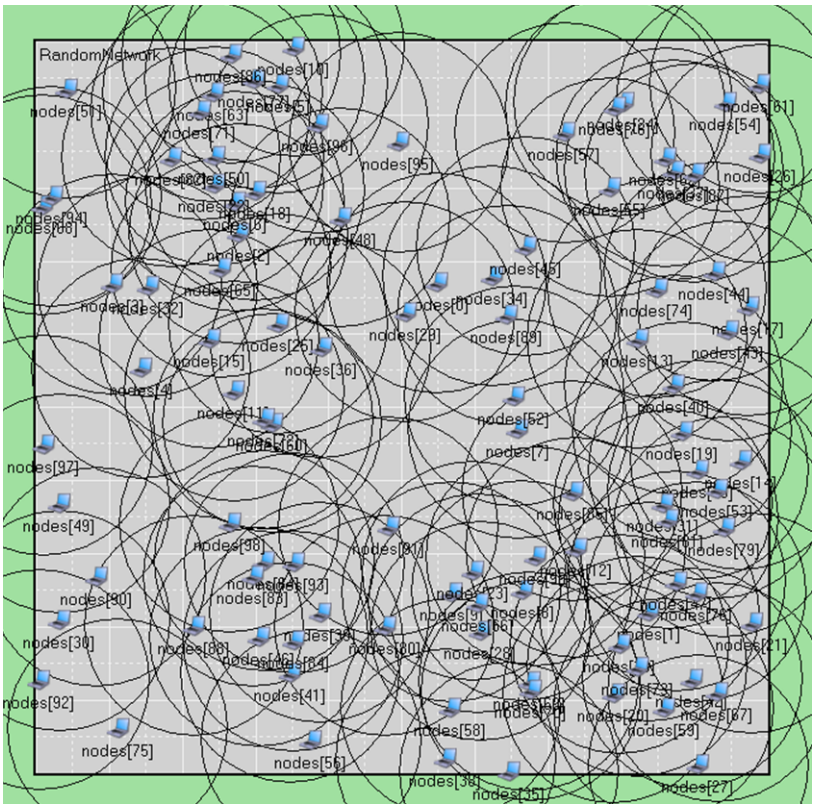


Figura 4.2: Os raios de comunicação dos nós definem as arestas do grafo G. Grafo com 100 nós.

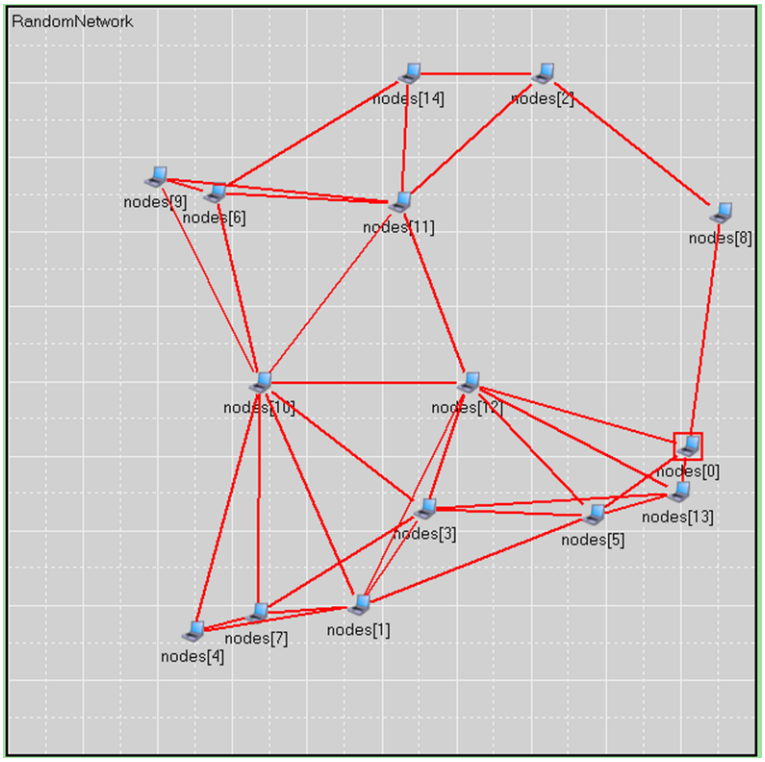


Figura 4.3: Grafo G formado a partir dos nós e seus raios de comunicação. Grafo com 15 nós.

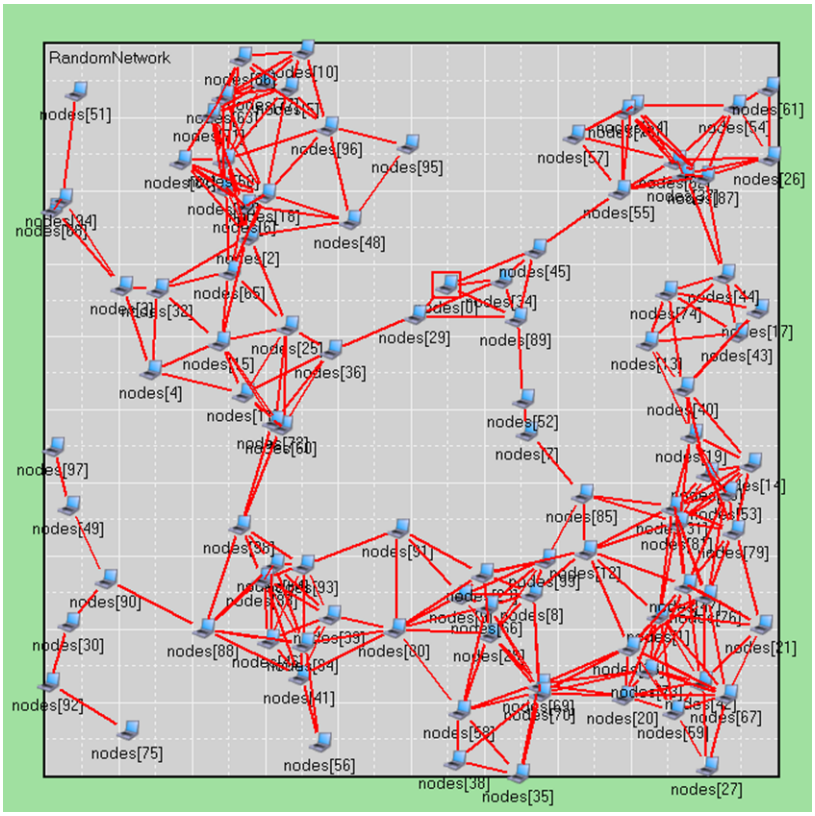


Figura 4.4: Grafo G formado a partir dos nós e seus raios de comunicação. Grafo com 100 nós.

Iremos inicialmente assumir um modelo síncrono de computação distribuída para construção dos algoritmos propostos. Assim, todos os processos são regidos por um relógio global comum, cuja computação local pode, ou não, ter seu tempo considerado, e as mensagens entre nós vizinhos enviadas em um pulso do relógio tem sua entrega garantida no pulso seguinte. A adoção deste modelo síncrono permite uma simplificação das discussões, entretanto não cria impedimentos para a posterior expansão das conclusões obtidas para o modelo totalmente assíncrono, como mostrado em [13] e que pode ser estendido para este trabalho. As computações distribuídas em G são assumidas para acontecerem em pulsos síncronos, representados por t, sendo $t \geqslant 0$ e $t \in \mathbb{Z}$.

Neste ambiente, o resultado esperado do mecanismo proposto é visto da seguinte forma: a cada pulso síncrono, um conjunto de processos tem garantido o acesso ao meio, de modo que suas transmissões ocorram livre de colisões para os nós receptores. Este conjunto é formado sempre por, pelo menos, um nó que irá realizar a transmissão. A melhor visualização a que se refere um nó ou processo ter acesso ao meio, é que no intervalo de tempo alocado a ele, o nó pode realizar a transmissão dos dados para outro nó, sem que ocorra o problema do terminal oculto atingindo nós que estão escutando. A computação distribuída que ocorre tem o objetivo de utilizar, sem colisões, o meio eletromagnético como um recurso compartilhado.

4.1.2 Formação da Vizinhança de um Salto e de dois Saltos

Para poder formar o grafo G cada nó i deve ter o conhecimento dos seus vizinhos de um salto, definindo o conjunto de nós V_i^1. Com o objetivo de considerar as possíveis interferências que um vizinho de dois saltos possa causar, cada nó i solicita a cada vizinho j a lista de vizinhos deste, e cria sua própria lista de vizinhos de dois saltos, formando o conjunto V_i^2. Neste conjunto são excluídas as redundâncias de vizinhos de dois saltos comuns a vizinhos de um salto. Formalizando estas vizinhanças, temos:

- V_i^1 o conjunto de nós vizinhos de um salto do nó n_i, tal que $n_j \in V_i^1$ se e somente se $d_{ij} \leq r_c$;
- V_i^2 o conjunto de nós vizinhos de dois saltos do nó n_i, tal que $n_k \in V_i^2$ se e somente se $n_k \notin V_i^1$ e $\exists n_j$ tal que $(n_j \in V_i^1) \wedge (n_k \in V_j^1)$;

4.1.3 Vizinhança de dois Saltos - Grafo de Interferência

Após a formação das vizinhanças de um e dois saltos o próximo passo é a formação do *grafo de interferência*, G_I. O objetivo do grafo de interferência é evitar o problema do terminal oculto. Seja $G_I = (V_I, E_I)$, onde V_I é o conjunto dos nós da rede que realizam transmissões e assim podem produzir interferências, ou seja, $V_I = V - S$. O conjunto de arestas E_I indica as relações de interferência entre os nós. Em particular, considere dois nós não sinks i e j. A aresta $(n_i, n_j) \in E_I$ se e somente se $n_i \in V_j^2$, ou equivalentemente, $n_j \in V_i^2$.

De um modo geral, uma aresta de interferência bloqueia o acesso simultâneo ao recurso compartilhado vista na reversão de arestas e retratado na Figura 4.5. Esta medida cria um impedimento para que n_i opere ao mesmo tempo que n_k. Desta forma, arestas de interferência devem fazer parte da dinâmica SMER. Entretanto, um nó i tem arestas em G_I de forma diferente para as versões *broadcast* e *unicast* do mecanismo. Estas diferenças estão expostas a seguir.

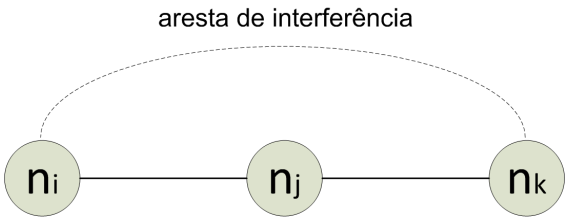


Figura 4.5: Inserção da aresta de interferência entre vizinhos de dois saltos.

Grafo de Interferência - Versão Broadcast

Na versão *broadcast* um nó n_i tem arestas de interferência com todos os vizinhos de dois saltos (exceto nós sinks), possibilitando o nó a realizar uma difusão sem colisões. Um nó n_i utiliza o conjunto V_i^2 para determinar quais arestas de G_I devem fazer parte da dinâmica SMER. Em particular temos:

- G_I o grafo de interferência derivado de G;
- $G_I = (V_I, E_I)$ tal que $V_I = V - S$;
- na versão *broadcast*: $E_I = \{(n_i, n_k) | n_k \in V_i^2 \text{ e } \{n_i, n_k\} \in V_I\}$;

Grafo de Interferência - Versão Unicast

Na versão *unicast* nem todas as arestas de interferência indicam uma relação de mutualidade no acesso ao recurso compartilhado. Uma aresta de interferência ocorre quando um vizinho de dois saltos n_k, se o vizinho default v_k de n_k é vizinho de um salto de n_i, ou seja, $v_k \in V_i^1$. O nó passa a ter uma relação seletiva de vizinhos de dois saltos para construção do escalonamento baseado no SMER.

Para exemplificar uma destas situações, a Figura 4.6 ilustra o caso em que dois nós n_i e n_k realmente precisam ter aresta de interferência entre eles. Esta situação é provocada pelo fato de n_i ter como v_i o nó n_j, que pertence a V_k^1. Então, n_k ao transmitir para outro destinatário v_k, fora da figura, tem sua transmissão recebida por n_j, que poderia estar recebendo a transmissão de n_i. A aresta de interferência evita que esta colisão ocorra.

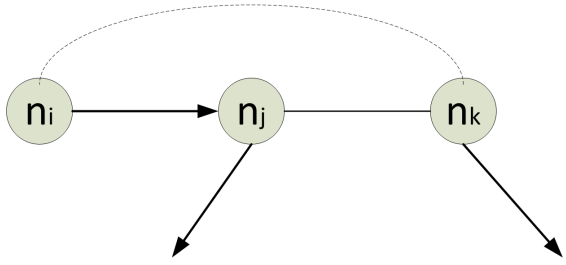


Figura 4.6: Situação em que é necessária a aresta de interferência entre vizinhos de dois saltos.

A figura 4.7 ilustra uma situação em que não é necessária a existência da aresta de interferência. Tanto n_i como n_k estão transmitindo para destinatários não pertencentes a V_k^1 e V_i^1, respectivamente. Assim, em uma transmissão *unicast* não é necessária a aresta de interferência entre estes nós.

Para um melhor entendimento da versão *unicast*, seja então:

- G_I o grafo de interferência derivado de G;

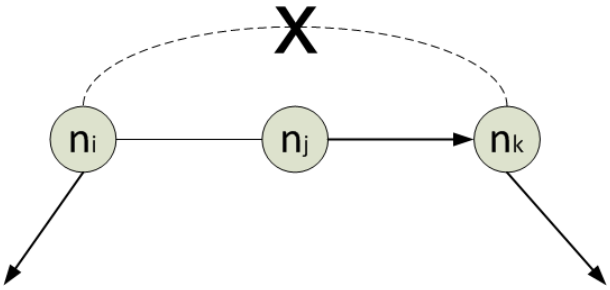


Figura 4.7: Situação em que não é necessária a aresta de interferência entre vizinhos de dois saltos.

- $G_I = (V_I, E_I)$ tal que $V_I = V - S$;
- na versão *unicast*: $E_I = \{(n_i, n_k) | (n_k \in V_i^2) \wedge (v_k \in V_i^1)$ e $\{n_i, n_k\} \in V_I\}$;

Revendo a Figura 3.8, verifica-se que a falta de medidas de coordenação entre as transmissões dos nós n_1 e n_3, daquele grafo, levou a colisão registrada no último estado da operação do SMER. A partir de agora, consideraremos uma nova abordagem para evitar este tipo de colisão.

O grafo G da Figura 4.8, é composto por cinco nós fontes e um nó sink, n_s. Cada nó fonte tem a demanda descrita sobre sua representação, o que seguindo o procedimento do SMER, permite o cálculo das reversibilidades. O grafo segue o fluxo ilustrado na figura, gerando uma topologia em que o nó sink situa-se como um nó raiz em uma árvore. Após esta fase, em um passo anterior a aplicação do SMER, é realizada a inserção das arestas de interferência, resultando na formação do grafo G_I composto pelos vértices e arestas de interferência da figura. Uma vez que o nó sink apenas recebe dados, ou seja, não transmite, deve ser retirado da dinâmica do SMER. Assim, dado o multigrafo e a orientação das arestas de M, representadas pelos tokens circulares de acordo com os conceitos do Capítulo 3, é realizada a dinâmica do SMER, conforme ilustrado na figura.

Cumprindo o previsto na teoria do algoritmo do SMER, ocorre a formação de um período após cinco orientações do multigrafo. É interessante notar agora que, mesmo em estados em que há mais de um nó operando, não há nenhum tipo de colisão nos nós receptores, o que é consequência direta da inserção das arestas de interferência.

No que se refere a criação das arestas de interferência, a abordagem feita na versão *unicast* requer especial atenção. Primeiro, porque evita que dois nós vizinhos de dois saltos impeçam a transmissão um do outro, sem de fato as transmissões provocarem o problema do terminal oculto em um terceiro nó. E segundo, porque essa abordagem tende a reduzir o período SMER para o escoamento das demandas dos nós, uma vez que permite um maior número de concorrências.

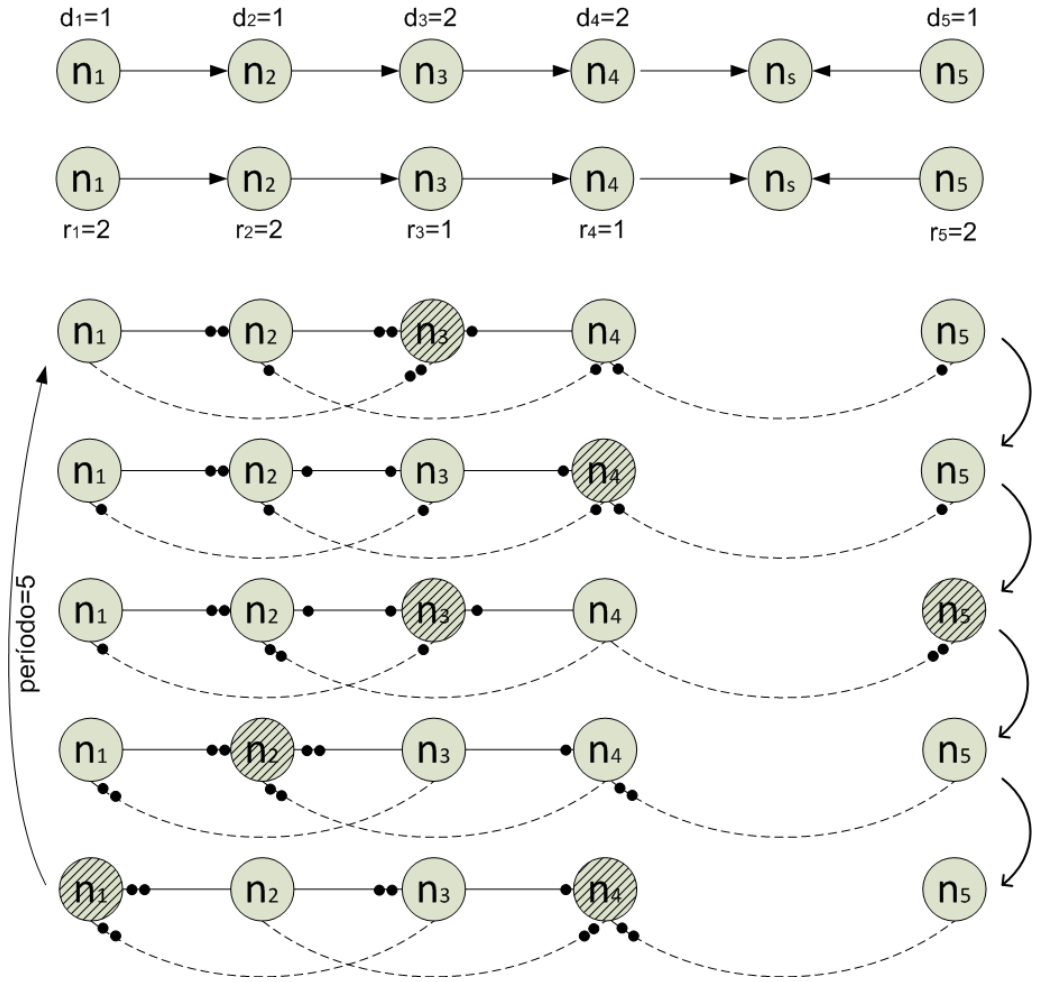


Figura 4.8: Modelo de transmissões utilizando o SMER para definição do período em um grafo com arestas de interferência.

As Figuras 4.9 e 4.10 apresentam o grafo de interferência gerado em duas instâncias da simulação deste trabalho, uma *broadcast* e uma *unicast*, respectivamente. Os nós sinks não participam do grafo de interferência, uma vez que não transmitem e somente realizam recepções.

Além do modo de criação das arestas de interferência, a versão unicast ainda apresenta outras particularidades. Sabe-se que nesta versão quando um nó transmite existe apenas um nó de destino. Por esta razão, outros vizinhos de um salto do transmissor podem ser habilitados a realizar transmissões, desde que atendidas condições específicas que evitem colisões nos receptores. Estas condições são derivadas das informações dos sinks e da definição dos caminhos até o sink adotado por cada nó sensor, estabelecidos no chamado grafo de fluxo.

Grafo de Fluxo e Conjunto de Nós Sinks

O grafo G_F, denominado *grafo de fluxo de dados*, é o grafo representativo dos caminhos no grafo de conectividade G, pelos quais efetivamente possam os dados do

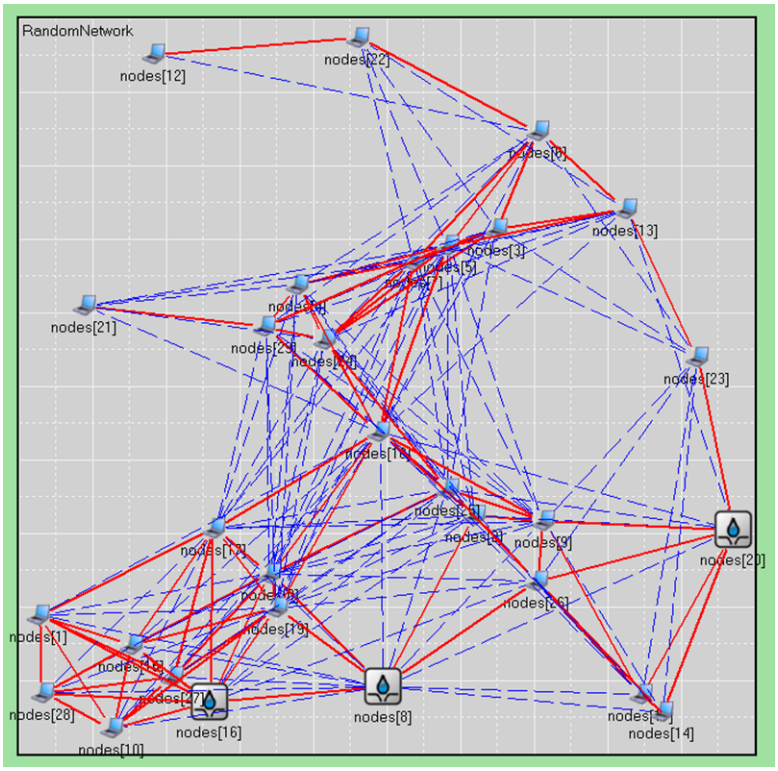


Figura 4.9: Grafo de interferência G_I (arestas de interferência tracejadas), versão *broadcast*. Grafo com 30 nós e 3 sinks.

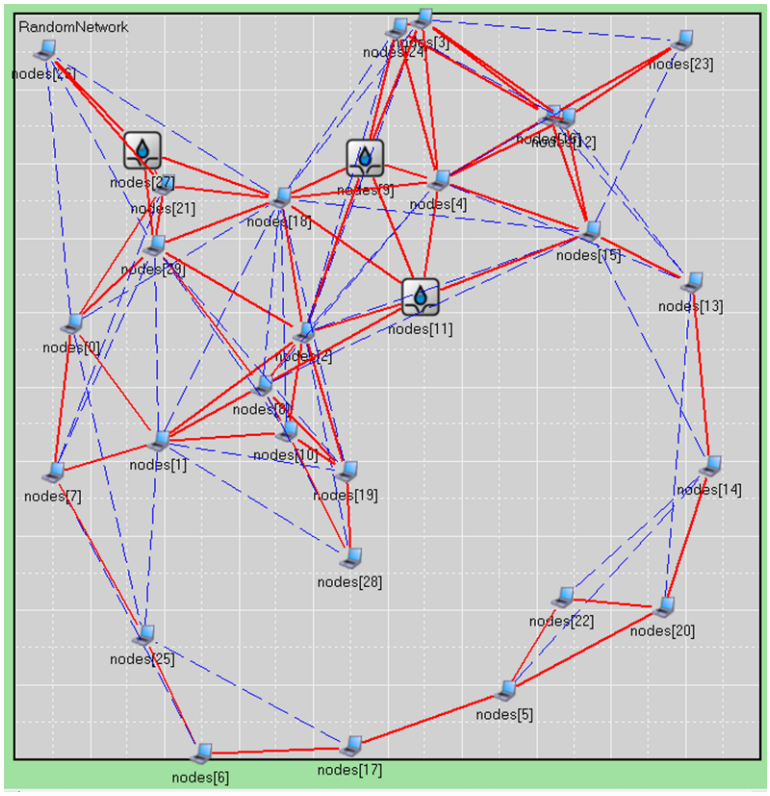


Figura 4.10: Grafo de interferência G_I (arestas de interferência tracejadas), versão *unicast*. Grafo com 30 nós e 3 sinks.

monitoramento desejado pela RSSF. O grafo de fluxo é por onde escoa, ou flui, os dados dos nós sensores para os nós sinks. Os nós sinks formam o conjunto S, composto de, pelo menos, um nó sink. As Figuras 4.11 e 4.12 ilustram o exemplo de um grafo de conectividade G e de um grafo de fluxo G_F, definido pelos nós de G e pelas arestas orientadas e em negrito da Figura 4.12. Repare que cada nó sink é raiz de uma árvore por onde os dados trafegam. O grafo de fluxo é utilizado somente na versão *unicast* do mecanismo. A versão *broadcast* não usa informações de rotas para sinks, preocupando-se somente com o bloqueio de transmissões de vizinhos de um e dois saltos, utilizando para isto o grafo de conectividade G.

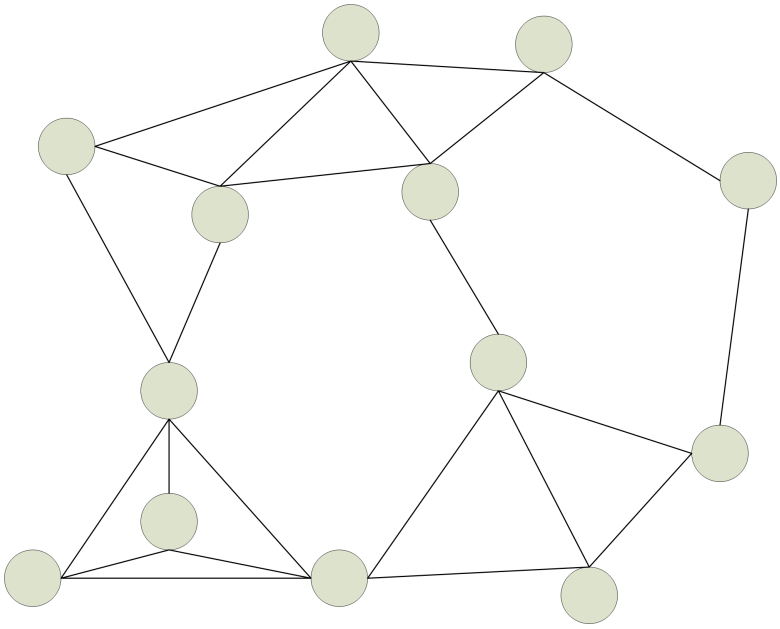


Figura 4.11: Exemplo de grafo de conectividade G.

Existem diversas formas de construir a árvore de escoamento de dados até cada um dos nós sinks da rede. Neste trabalho iremos considerar uma forma pragmática, baseada em caminhos mínimos definidos da seguinte forma: cada nó sink inicia uma difusão na rede para informar aos outros nós a sua condição de nó sink. Cada sink envia uma mensagem de difusão inicialmente para seus vizinhos de 1 salto. Estes, por sua vez, replicam a mensagem para seus vizinhos de 1 salto, e assim se segue, incrementando a distância ao sink e registrando-se no caminho da mensagem. Ao receber uma mensagem originária de um sink, um nó n_i grava em sua memória o seguinte:

- o identificador do sink que originou a mensagem;
- o vizinho que lhe entregou a mensagem;
- o número de saltos até o sink que originou a mensagem, que é um valor contido na mensagem.

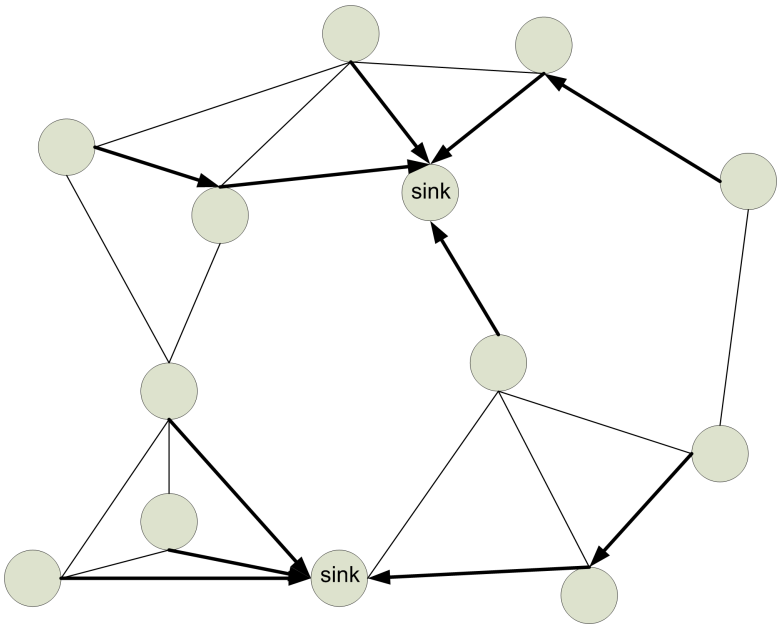


Figura 4.12: Exemplo de grafo de fluxo G_F derivado do grafo de conectividade.

O conjunto S reúne os nós com função de sink, sabendo-se que $S \subset V$ e $|S| \geq 1$. Conhecendo quantos sinks existem na rede, após receber mensagem de todos os sinks, o nó i escolhe como *default* aquele sink que melhor atende a métrica estabelecida pelo projetista, como por exemplo, o menor caminho medido em saltos até o sink. Apesar da escolha do sink parecer relevante, ela é meramente ilustrativa, e o algoritmo sendo proposto é ortogonal a como é feita esta escolha. A principal informação que um nó i guarda, neste momento, é a informação de qual vizinho de um salto leva ao sink, ou seja, para qual nó vizinho o nó i deve transmitir. Este nó vizinho é definido como o *vizinho default* de i, representado por v_i. Obviamente, os nós sinks não possuem vizinhos default, pois não possuem demanda a transmitir e apenas recebem dados.

É importante realçar novamente neste momento a diferença entre nós sinks no grafo de fluxo G_F e nós sink no SER ou SMER, ou seja, decorrentes da dinâmica do escalonamento por reversão de arestas, conforme visto no Capítulo 3. Os nós sinks em G_F são aqueles que recebem as informações coletadas de um monitoramento desejado, enquanto que os nós que estão na condição de sink SMER no escalonamento por reversão de arestas simplesmente representam os nós que possuem um número suficiente de arestas incidentes orientadas na sua direção dando condições para que o nó realize sua transmissão.

Do exposto no texto desta Seção, define-se que:

- Seja G_F o grafo de fluxo derivado de G;
- $G_F = (V, E_F)$, onde $E_F \subset E$ e $E_F = \bigcup_{k=0}^{n-1} (n_k, v_k)$, $n_k \notin S$. Logo, $(n_k, v_k) \in E$.
- Seja E_s o conjunto das arestas incidentes em nós sinks, ou seja:

- $E_s = \{(n_i, n_s)/\forall n_s \in S \text{ e } n_i \in V_s^1\}$.

Nesta versão unicast do protocolo, após um nó i definir o vizinho default v_i em G_F, ele envia uma mensagem aos vizinhos em V_i^1 e V_i^2 informando a estes nós o seu vizinho default v_i escolhido. Então, esta mensagem é utilizada pelos nós destas vizinhanças para que cada um analise se está em condição similar ao *problema do terminal exposto* ou o *problema do terminal oculto* com n_i. A vizinhança de um salto realiza o procedimento descrito a seguir.

Vizinhança de um Salto - Versão *unicast*

O *problema do terminal exposto* ocorre em protocolos de acessos ao meio onde os nós *sentem* o meio para detectar a existência de uma transmissão no espectro eletromagnético, como ilustrado na Figura 4.13. Considere que, no instante em que o nó n_j transmite para n_i, o nó n_k também deseja transmitir para n_l. Seguindo o procedimento do suposto protocolo, antes de iniciar a transmissão n_k verifica se há uma transmissão em andamento. Ao detectar a transmissão de n_j em andamento o nó n_k deixa de transmitir. Entretanto, analisando a figura, é possível verificar que a transmissão desejada por n_k poderia ocorrer, pois n_l e n_i estão fora do alcance de n_j e n_k, respectivamente.

Figura 4.13: Problema do terminal exposto.

A detecção de uma portadora não tem sentido neste trabalho, mas aqui dois nós vizinhos podem causar semelhante efeito pela existência de uma aresta do grafo de conectividade, conforme explicado a seguir.

Como já visto no Capítulo anterior, a existência de uma aresta entre dois nós estabelece uma restrição ao recurso que está sendo compartilhado, não permitindo mais o acesso simultâneo dos nós ligados pela aresta ao recurso. Quando há a formação do grafo de conectividade G, a partir do raio de comunicação dos nós, naturalmente são formadas as arestas em G. Entretanto, nem toda aresta formada em G é utilizada para o escoamento de dados pois vários caminhos alternativos são formados e as rotas escolhidas são definidas em G_F. Existem situações em que as arestas de G não pertencentes a G_F tornam-se desnecessárias sob o enfoque do escalonamento, pois, na verdade, impedem uma concorrência que não produz interferência.

A Figura 4.14 ilustra um exemplo em que dois nós n_i e n_j, vizinhos de um salto, enviam seus dados para dois outros nós, tais que v_i não pertence a V_j^1 e v_j também não pertence a V_i^1.

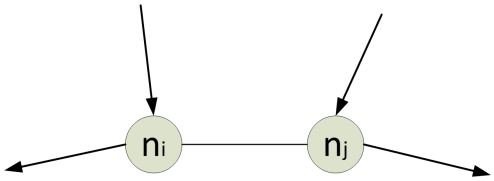


Figura 4.14: Efeito semelhante ao problema do terminal exposto.

A aresta que une os nós n_i e n_j não é utilizada para o escoamento de dados e também não realiza um bloqueio necessário de acesso simultâneo ao recurso, uma vez que a transmissão simultânea de n_i e n_j não gera nenhum tipo de colisão nos respectivos destinatários. Por esta razão, esta aresta torna-se desnecessária quando da aplicação do SMER.

Assim, a condição exposta a seguir permite a retirada de uma aresta de conectividade, na versão *unicast*, para o funcionamento do SMER. O conjunto das arestas que podem ser retiradas forma o conjunto E^*.

- Dado que $((n_i, n_j) \in G) \wedge ((n_i, n_j) \notin G_F)$ pode-se retirar uma aresta (n_i, n_j) para o funcionamento do SMER quando $(v_i \notin V_j^1) \wedge (v_j \notin V_i^1)$.
- Define-se E^* como o conjunto das arestas de G retiradas para evitar um bloqueio desnecessário entre dois nós vizinhos de um salto.

A consequência da condição acima, é que um nó i determina com quais nós em V_i^1 se relacionará durante a dinâmica do SMER. Estas informações das arestas de um salto se unem as informações das arestas de dois saltos produzidas pelo grafo de interferência G_I. Concluindo esta etapa, o grafo da versão unicast passa a fase de troca de informações sobre demandas. O mesmo procedimento é executado na versão broadcast.

4.1.4 Troca de Informações sobre Demanda

Ao seguir os passos anteriores, de acordo com a versão *broadcast* ou *unicast* do mecanismo, os nós já possuem a relação dos vizinhos de um salto e dois saltos que irão participar da dinâmica do SMER. Iremos chamar estes nós de vizinhos participantes da dinâmica do SMER. A comunicação com estes nós pode ser direta, para os vizinhos de um salto, ou indireta, através de algum vizinho, para os vizinhos de dois saltos. Cada nó inicia uma troca de mensagens com seus vizinhos participantes informando suas demandas. Com estas informações, os nós realizam o cálculo

da reversibilidade localmente, conforme apresentado após a revisão do cálculo da reversibilidade utilizando o conhecimento global das demandas. O cálculo local das reversibilidades é uma das contribuições deste trabalho.

4.1.5 Determinação das Reversibilidades Utilizando Informação Global de Demandas

Como visto na Equação 3.5, o cálculo da reversibilidade inerente a cada nó depende do conhecimento, por cada nó, das demandas de todos os nós que formam a rede. Ou seja, os nós precisam ter conhecimento global das demandas. O exemplo a seguir ilustra este procedimento.

A Figura 4.15 mostra um grafo com três nós, n_1, n_2 e n_3 que requerem o acesso a um recurso compartilhado, sob as seguintes demandas: $d_1 = 1$, $d_2 = 2$ e $d_3 = 3$. Para a determinação das reversibilidades, realizam-se os seguintes cálculos:

$$r_1 = \frac{mmc(d_1, d_2, d_3)}{d_1} = 6$$

$$r_2 = \frac{mmc(d_1, d_2, d_3)}{d_2} = 3$$

$$r_3 = \frac{mmc(d_1, d_2, d_3)}{d_3} = 2$$

De posse das reversibilidades $r_1 = 6$, $r_2 = 3$ e $r_3 = 2$, conforme visto na Figura 4.15, a orientação inicial do multigrafo deve ser estabelecida de forma a atender aos Teoremas 1 e 2 (veremos em breve como fazer esta orientação inicial). A dinâmica do SMER segue conforme a Figura 4.16. Inicialmente o nó n_2 é sink SMER e conforme as arestas são revertidas novos sinks SMER se formam. Após seis orientações de multigrafo a orientação inicial se repete, formando um período.

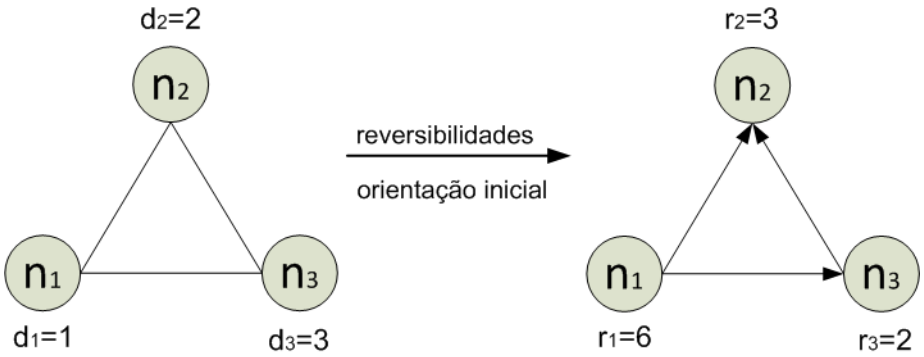


Figura 4.15: Representação das demandas e reversibilidades de um grafo com 3 nós.

Examinando a Figura 4.16 verifica-se que o cálculo global da reversibilidade garante a cada nó atender a demanda de acesso ao recurso compartilhado conforme

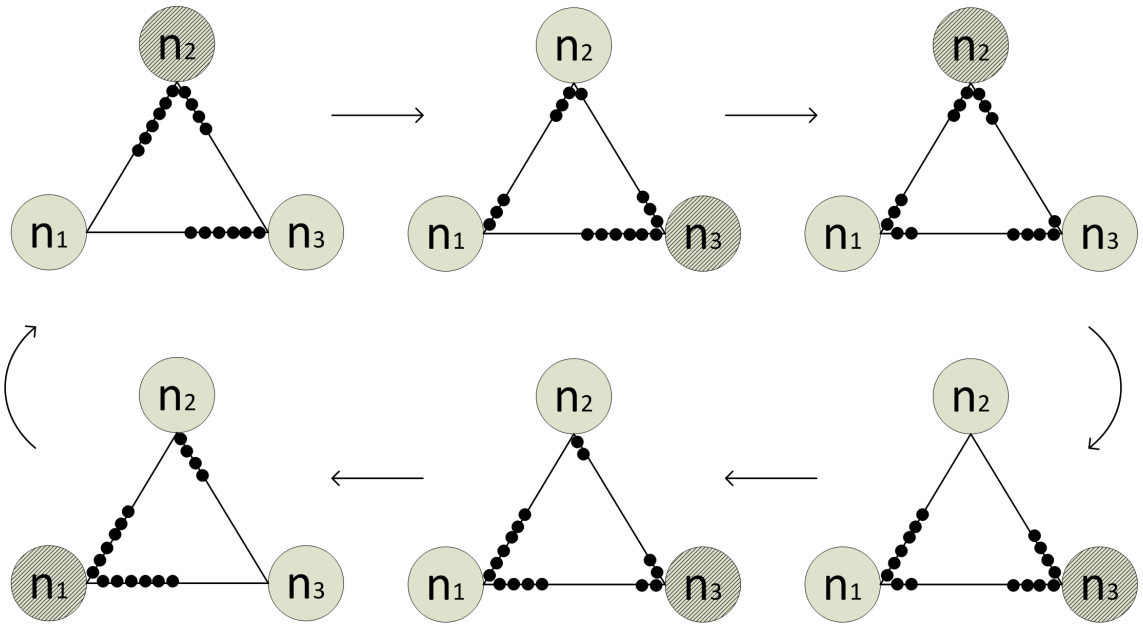


Figura 4.16: exemplo da dinâmica SMER com período $p = 6$.

desejado. Entretanto, o cálculo global requer a difusão de demanda por parte dos nós ou a existência de um elemento central que contabilize globalmente a demanda de todos os nós.

Baseado nos fundamentos estabelecidos no Capítulo 3, a Seção seguinte apresenta uma abordagem derivada da global para o cálculo da reversibilidade, de modo que sejam utilizadas informações locais para que um nó descubra sua reversibilidade localmente e de modo a determinar seu período dentro da dinâmica do SMER.

4.1.6 Determinação das Reversibilidades Utilizando Informação Local de Demandas

O cálculo referente a determinação da reversibilidade de um nó n_i em uma rede com apenas dois nós n_i e n_j, como explicado na Seção 3.3, é dado por:

$$r_i = \frac{mmc\,(d_i, d_j)}{d_i} \tag{4.1}$$

Considere agora o mesmo cálculo mas com a inserção, na Equação 4.1, do índice referente ao nó vizinho no grafo, o que não produz qualquer alteração para os valores de demanda e reversibilidade para este caso de dois nós:

$$r_{ij} = \frac{mmc\,(d_i, d_j)}{d_i} \tag{4.2}$$

Assim, para ilustrar esta modificação, a Figura 4.17 apresenta a descrição antes e depois da alteração na representação das reversibilidades.

Figura 4.17: Inclusão da referência ao nó vizinho no índice da reversibilidade.

O cálculo da quantidade de arestas e_{ij} entre n_i e n_j é o mesmo dado pela Equação 3.4, com a designação das reversibilidades com a inserção do índice de ambos os nós que compartilham a aresta, representado na Equação 4.3:

$$e_{ij} = r_{ij} + r_{ji} - mdc(r_{ij}, r_{ji}) \tag{4.3}$$

O nó n_i opera sempre que o número de arestas apontadas na sua direção for igual ou maior que sua reversibilidade. Se a_{ij} indicar o número de arestas apontadas na direção de n_i, e a_{ji} indicar o número de arestas apontadas na direção de n_j, tal que $a_{ij} + a_{ji} = e_{ij}$, o nó n_i opera quando $r_{ij} \leq a_{ij}$, o que garante exclusão mútua com o vizinho em questão.

Considerando a existência de outro nó, n_k, que está conectado a n_i, seguindo a mesma sequência descrita anteriormente, fica claro que:

$$r_{ik} = \frac{mmc\,(d_i, d_k)}{d_i}$$

$$r_{ki} = \frac{mmc\,(d_i, d_k)}{d_k}$$

. O número de arestas entre n_i e n_k é dado por:

$$e_{ik} = r_{ik} + r_{ki} - mdc(r_{ik}, r_{ki})$$

. A Figura 4.18 ilustra a generalização da idéia anterior para uma rede com três nós em linha. Repare que r_{ij} é calculado apenas utilizando as demandas de n_i e nj, de acordo com a Equação 4.2.

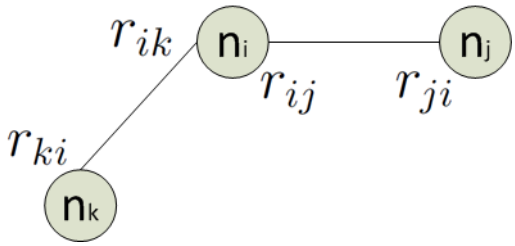


Figura 4.18: Relacionamento das reversibilidades entre o nó n_i e os nós n_j e n_k.

A solução para a operação do nó n_i é a mesma estabelecida na Seção 3.2.2 desta

vez considerando as reversibilidades locais, ou seja, n_i opera se ocorrem, pelo menos, r_{ix} arestas orientadas em sua direção, entre as e_{ix} arestas que ele compartilha para todo vizinho n_x. A relação de n_i com cada um dos vizinhos deve atender a condição básica de operação. Essa operação pode ser vista como uma decomposição da relação do nó n_i com cada vizinho n_x. A tabela verdade 4.1 ilustra quando o nó n_i pode operar. Repare que apenas quando as duas condições forem verdades a condição de operação será verdade.

Nó n_i		
(n_i, n_j)	(n_i, n_k)	
$r_{ij} \leq a_{ij}$	$r_{ik} \leq a_{ik}$	*funcionamento*
V	V	opera
V	F	não opera
F	V	não opera
F	F	não opera

Tabela 4.1: Tabela verdade para operação do nó n_i

Para ilustrar o funcionamento descrito, considere o mesmo grafo da Figura 4.15 com os nós n_1, n_2 e n_3 e suas respectivas demandas $d_1 = 1$, $d_2 = 2$ e $d_3 = 3$, apresentado novamente na Figura 4.20. Os cálculos para determinar as reversibilidades serão realizados de acordo com a abordagem local onde cada par de nós vizinhos estabelece a relação direta de reversibilidade:

$$r_{12} = \frac{mmc(d_1,d_2)}{d_1} = 2;$$

$$r_{13} = \frac{mmc(d_1,d_3)}{d_1} = 3;$$

$$r_{21} = \frac{mmc(d_1,d_2)}{d_2} = 1;$$

$$r_{23} = \frac{mmc(d_2,d_3)}{d_2} = 3;$$

$$r_{31} = \frac{mmc(d_1,d_3)}{d_3} = 1;$$

$$r_{32} = \frac{mmc(d_2,d_3)}{d_3} = 2;$$

Para o cálculo das arestas do multigrafo é utilizada a Equação 4.3. Então, tem-se:

$$e_{12} = e_{21} = 2 + 1 - 1 = 2;$$
$$e_{13} = e_{31} = 3 + 1 - 1 = 3;$$
$$e_{23} = e_{32} = 3 + 2 - 1 = 4$$

Uma forma interessante de visualizar este conceito de reversibilidade local é

através da definição de um *hiper-nó*, conforme ilustrado na Figura 4.19. Nesta visão, um *hiper-nó* é um nó composto por nós internos com relações individuais com os nós internos do *hiper-nó* vizinho. Um hiper-nó só opera quando todos os nós internos que o compõem, estiverem em condições de operar.

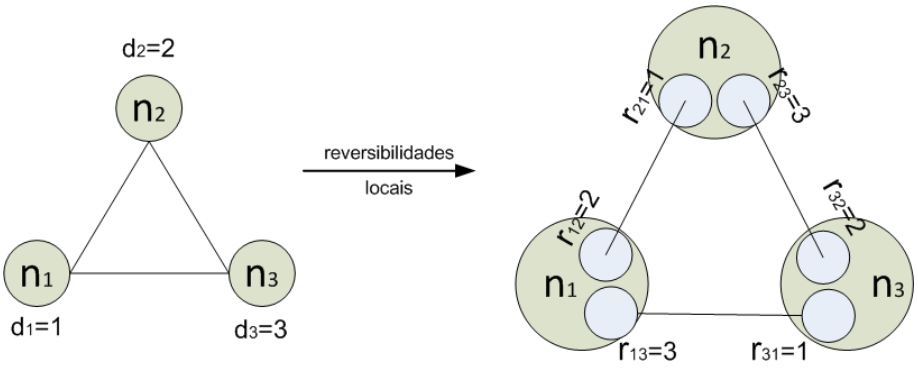


Figura 4.19: Representação das reversibilidades locais através de hiper-nós.

Na visualização resumida dada pela Figura 4.20 os nós estão prontos para a dinâmica do SMER. O funcionamento ocorre de acordo com as restrições de operações locais e um período é obtido a partir da repetição de uma orientação anterior do multigrafo, o que ocorre após a sexta orientação, conforme ilustrado na Figura 4.21. Repare que o período formado atende as demandas de operação dos nós.

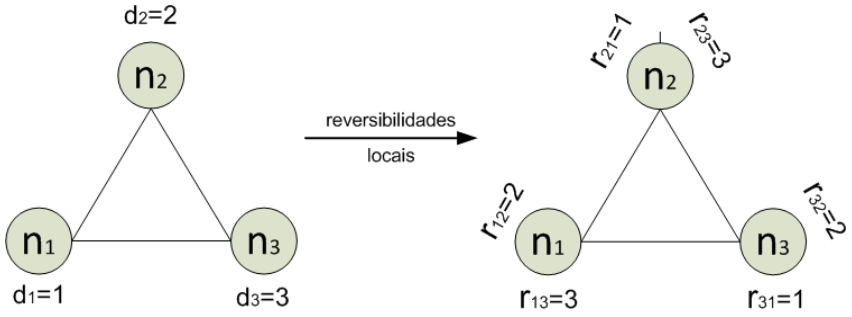


Figura 4.20: Representação das demandas e reversibilidades locais de um grafo com 3 nós.

4.1.7 Multigrafo para Aplicação do SER e SMER

Como visto em 3.2.2, a partir do grafo de conectividade G é formado o multigrafo M para o funcionamento do SMER. Aqui, M é gerado de forma diferente em cada uma das versões propostas (*broadcast* e *unicast*) da seguinte forma:

Multigrafo para Aplicação do SER e SMER - Versão Broadcast

Na versão *broadcast* $M = (V_b, E_b)$, tal que:

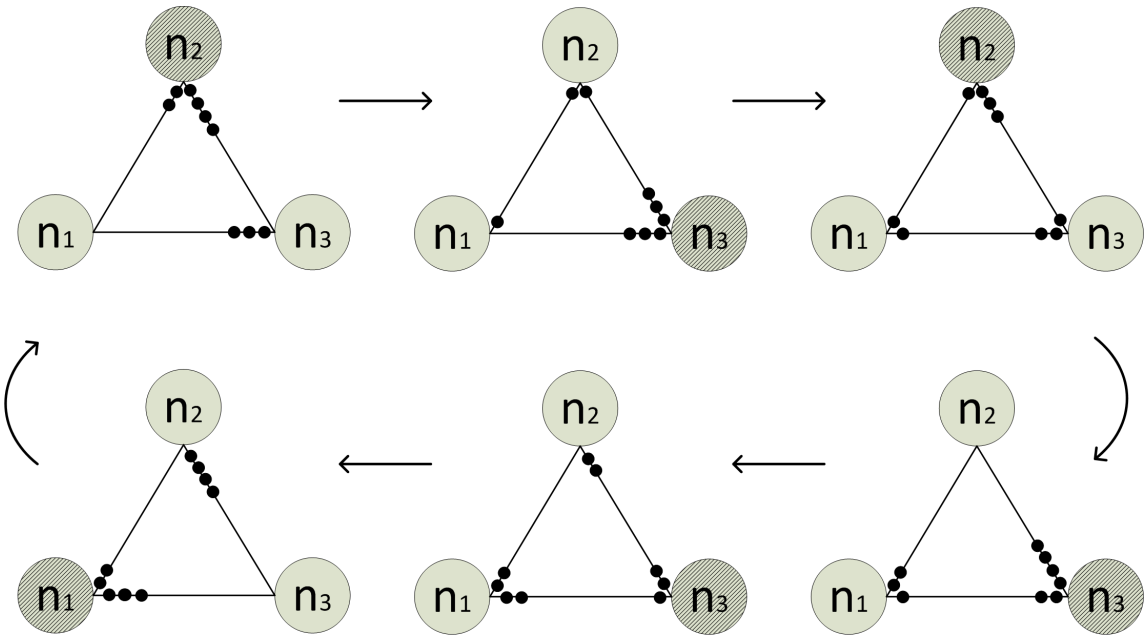


Figura 4.21: exemplo da dinâmica SMER de período $p = 6$, cuja reversibilidade é localmente calculada.

- $V_b = V - S$;
- E_b é obtido a partir da determinação de e_{ij} arestas entre (n_i, n_j), tal que $(n_i, n_j) \in (E + E_I - E_s)$, lembrando que E_I é o conjunto das arestas de interferência da versão broadcast e E_s é o conjunto das arestas incidentes em nós sinks.

Multigrafo para Aplicação do SER e SMER - Versão Unicast

Na versão *unicast* $M = (V_u, E_u)$, tal que:

- $V_u = V - S$;
- E_u é obtido a partir da determinação de e_{ij} arestas entre (n_i, n_j), tal que $(n_i, n_j) \in (E + E_I - E^* - E_s)$, lembrando que E_I é o conjunto das arestas de interferência da versão unicast, e que E^* é o conjunto das arestas de conectividade que podem ser retiradas para evitar o bloqueio de transmissão similar ao problema do terminal exposto.

4.2 Dinâmicas do SER e SMER no Multigrafo M

Após todos os procedimentos que mapearam o grafo de conectividade G em um multigrafo M, a última etapa do funcionamento do mecanismo de escalonamento é determinar o período de repetição da dinâmica do SMER correspondente. Antes que

isto ocorra, é necessário definir uma orientação acíclica ω que implica na orientação inicial μ_0 em M. Para determinar ω, são utilizados os algoritmos distribuídos Alg-Cor e Alg-Arestas vistos no final do Capítulo 3.

Realizada a orientação acíclica, a principal preocupação se torna a inserção das e_{ij} arestas entre dois nós n_i e n_j com a certeza do atendimento do Teorema 2, dos fundamentos do escalonamento por reversão de múltiplas arestas, que garante que o SMER irá operar sem a ocorrência de *deadlocks*. A questão pendente é como orientar as e_{ij} arestas entre dois nós em μ_0 de modo a satisfazer as condições (i) e (ii) previstas na Seção 3.2.1.

Para evitar completamente esta preocupação, uma nova abordagem apresentada a seguir inicia o processo de descoberta do período pelo SER e dinamicamente comuta para o SMER, através da alteração da reversibilidade durante a operação dos nós. É mostrado que este processo mantém as premissas que atendem ao Teorema 2.

4.2.1 Inicialização da Dinâmica SMER

A idéia desta Seção é mostrar que é possível inicializar a dinâmica do SMER utilizando o SER, e assim evitar a difícil tarefa de inserir as múltiplas arestas de M de modo a atender a condição do Teorema 2, e desta forma, evitar que ocorra a formação de ciclos, pela inversão das arestas de M, ao longo da dinâmica do SMER.

A inicialização do SER ocorre a partir da orientação acíclica inicial. De acordo com a orientação acíclica ω, uma única aresta entre dois nós é apontada na direção determinada pelo algoritmo de orientação. Não é difícil observar que ω pode ser vista como a orientação inicial μ_0 de um multigrafo M no qual todas as reversibilidades possuem valor unitário. O cálculo de e_{ij} para qualquer dois nós n_i e n_j, ambos com reversibilidades unitárias, leva a existência de somente uma aresta entre os nós, exatamente como ocorre no SER. Na orientação inicial ω são definidos os nós sinks SMER. Uma vez que possuem todas as arestas incidentes direcionadas para si, ou seja, uma aresta com cada vizinho no multigrafo, o nó sink SMER pode operar na configuração inicial dada pelo SER.

Esta configuração inicial atende ao Teorema 2, uma vez que qualquer orientação de M que resulta em $\sigma(k) < \rho(k)$, para todo $k \in K$, é uma boa orientação inicial para o SMER. Se o valor da reversibilidade é unitário, e em ω está garantida a inexistência de um ciclo pelo algoritmo de orientação acíclica, então também fica garantida a validade da inequação, visto que a soma das reversibilidades sempre será maior que a soma das arestas em uma direção transversal do ciclo, conforme as Equações 3.2 e 3.3. A Figura 4.22 ilustra esta análise através de um exemplo bem simples. A validade da inequação $\sigma(k) < \rho(k)$, neste exemplo, é dada pelos cálculos

que se seguem.

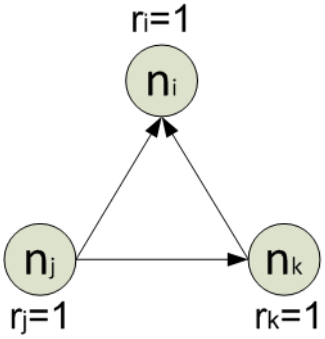


Figura 4.22: Grafo para análise da inequação $\sigma(k) < \rho(k)$.

$$\rho(k) = \sum_{n_i \in k} r_i = 3$$

$$\sigma_s(k) = max \left\{ \sum_{(n_i,n_j) \in k^+} a_s^{ij} = 2, \sum_{(n_i,n_j) \in k^-} a_s^{ij} = 1 \right\}$$

$$\sigma(k) < \rho(k) \Longrightarrow 2 < 3$$

Descrita esta validade da inicialização do SMER através da configuração do SER o próximo passo é verificar como os nós podem passar da operação do SER para a dinâmica do SMER, com as reversibilidades definidas pelo cálculo em função da demanda. É importante notar que a reversibilidade de um nó pode ser dinamicamente alterada quando o nó está na condição de sink SMER. Neste momento, conhecendo a reversibilidade dos nós da vizinhança, o nó pode recalcular o número de arestas incidentes com seus vizinhos no multigrafo, e reverter as arestas sob a contagem da nova reversibilidade. Para o vizinho nenhuma alteração ocorrerá, entretanto a rede redimensionará o período de acordo com a dinâmica do SMER a partir daquele momento. Este cálculo do novo número de arestas é feito baseado na própria Equação 3.4 e para o novo valor da reversibilidade desejada.

Na verdade, a reversibilidade de um nó pode ser alterada para mais, ou para menos, no momento em que o nó está na condição de sink SMER. Para a dinâmica do multigrafo M proposto neste trabalho, esta alteração de reversibilidade precisa ocorrer uma única vez, saindo da condição inicial unitária é indo para a condição de operação com as reversibilidades calculadas entre os nós. Deve-se notar que a validade da inequação $\sigma(k) < \rho(k)$ não se altera, visto que incrementos, ou decrementos, na reversibilidade, se reproduzem em ambas as parcelas da inequação, respeitando deste modo a condição do Teorema 2.

Resumidamente, esta operação é feita da seguinte forma:

1. Os nós iniciam a dinâmica com uma reversibilidade unitária;

2. Ao ser sink SMER pela primeira vez o nó recalcula o número de arestas com cada vizinho de acordo com a reversibilidade desejada. O nó utiliza o número de arestas incidentes como valor da reversibilidade do nó vizinho para fins do cálculo do novo número de arestas entre os nós;

3. O nó envia uma mensagem para cada vizinho indicando a reversão de arestas em quantidade idêntica a sua reversibilidade, conforme previsto no SMER;

4. O nó passa a operar na reversibilidade desejada;

A Figura 4.23 apresenta passo-a-passo a conversão da reversibilidade em um multigrafo que inicia a dinâmica do SMER através do SER. Pode-se verificar na figura que a inicialização é feita com a orientação acíclica inicial considerando a reversibilidade unitária nos nós (as arestas de M são representadas pelos tokens circulares). Na primeira orientação do multigrafo, o nó n_2 é sink SMER pela primeira vez. Utilizando a Equação 4.3, dada por $e_{ij} = r_{ij} + r_{ji} - mdc(r_{ij}, r_{ji})$, ele recalcula as arestas com os vizinhos n_1 e n_3.

O nó n_2 tem uma aresta com cada vizinho, e portanto, identifica o valor unitário como a reversibilidade dos vizinhos, uma vez que a reversibilidade inicial de n_2 é unitária. Realizam-se, então, as seguintes operações:

$$e_{21} = r_{21} + r_{12} - mdc(r_{21}, r_{12}) = 1 + 1 - 1 = 1$$

$$e_{23} = r_{23} + r_{32} - mdc(r_{23}, r_{32}) = 3 + 1 - 1 = 3$$

Entre os nós n_2 e n_1 mantém-se o valor unitário de arestas, pois o cálculo de e_{21} baseado nas reversibilidades, resultou na unidade. Entretanto, entre os nós n_2 e n_3 a quantidade de arestas e_{23} modificou-se para três arestas. Como já existia uma aresta, duas novas arestas foram adicionadas entre os nós para atender ao número de arestas calculado pela Equação 4.3. Visto que a reversibilidade r_{23} tem valor igual a três, então as três arestas foram revertidas para o nó n_3, e como a reversibilidade r_{21} tem valor unitário, uma aresta foi revertida na direção do nó n_1, tudo conforme prescreve a dinâmica do SMER.

Considerando a reversibilidade unitária inicial, agora o nó n_3 é sink SMER, pois possui pelo menos uma aresta apontada na sua direção oriundas dos vizinhos n_1 e n_2. Analogamente ao procedimento de n_2, n_3 muda sua reversibilidade para a desejada através do novo cálculo do número de arestas com seus vizinhos.

$$e_{31} = r_{31} + r_{13} - mdc(r_{31}, r_{13}) = 1 + 1 - 1 = 1$$

$$e_{32} = r_{32} + r_{23} - mdc(r_{32}, r_{23}) = 2 + 3 - 1 = 4$$

Repetindo o procedimento anterior, as quantidades de arestas são ajustadas de acordo com o novo cálculo e revertidas de acordo com a nova reversibilidade do nó n_3. Como existiam três arestas entre o nó n_3 e o n_2, e agora e_{32} indica que devem existir quatro arestas, uma aresta é adicionada entre os nós. Em seguida a reversibilidade $r_{32} = 2$ indica que duas arestas devem ser revertidas na direção do nó n_2, conforme ilustrado na figura.

Na sequência, o nó n_1 torna-se sink SMER pela primeira vez e os mesmos cálculos são realizados:

$$e_{12} = r_{12} + r_{21} - mdc(r_{12}, r_{21}) = 2 + 1 - 1 = 2$$

$$e_{13} = r_{13} + r_{31} - mdc(r_{13}, r_{31}) = 3 + 1 - 1 = 3$$

O nó n_1 acrescenta uma aresta na sua relação com o nó n_2 e duas arestas na sua relação com o nó n_3, revertendo para ambos os nós, as respectivas reversibilidades com estes vizinhos. Ao final deste passo, em que todos os nós estão operando com a reversibilidade relacionada com sua demanda, o multigrafo segue a dinâmica SMER até a definição do período.

A etapa da inicialização no SER e posterior conversão para o SMER, até a obtenção do período dos nós, finaliza a elaboração do mecanismo de escalonamento proposto neste trabalho. A seção seguinte explica a relação entre o período formado pela dinâmica do SMER e um quadro de transmissão composto por slots, de acordo com uma transmissão do meio rádio no espectro eletromagnético.

4.2.2 Relação entre Período SMER e Quadro de Transmissão

Já foi visto que a dinâmica do SMER aplicada a um multigrafo resulta em um período que reflete um escalonamento para o acesso a recursos compartilhados por parte dos nós que compõem o multigrafo, inclusive com diferentes necessidades de frequências de acessos aos recursos. Ao tratar das transmissões rádio a operação de um nó nada mais significa do que o acesso ao espectro eletromagnético, guardadas as restrições de acessos pelos nós vizinhos, cujo compartilhamento simultâneo traria algum tipo de interferência.

Em uma visão global, ao se definir o período da rede também se define o período no qual todos os nós têm suas demandas de transmissão atendidas. Estas transmissões vão ocorrendo a medida que as orientações do multigrafo mudam, criando a

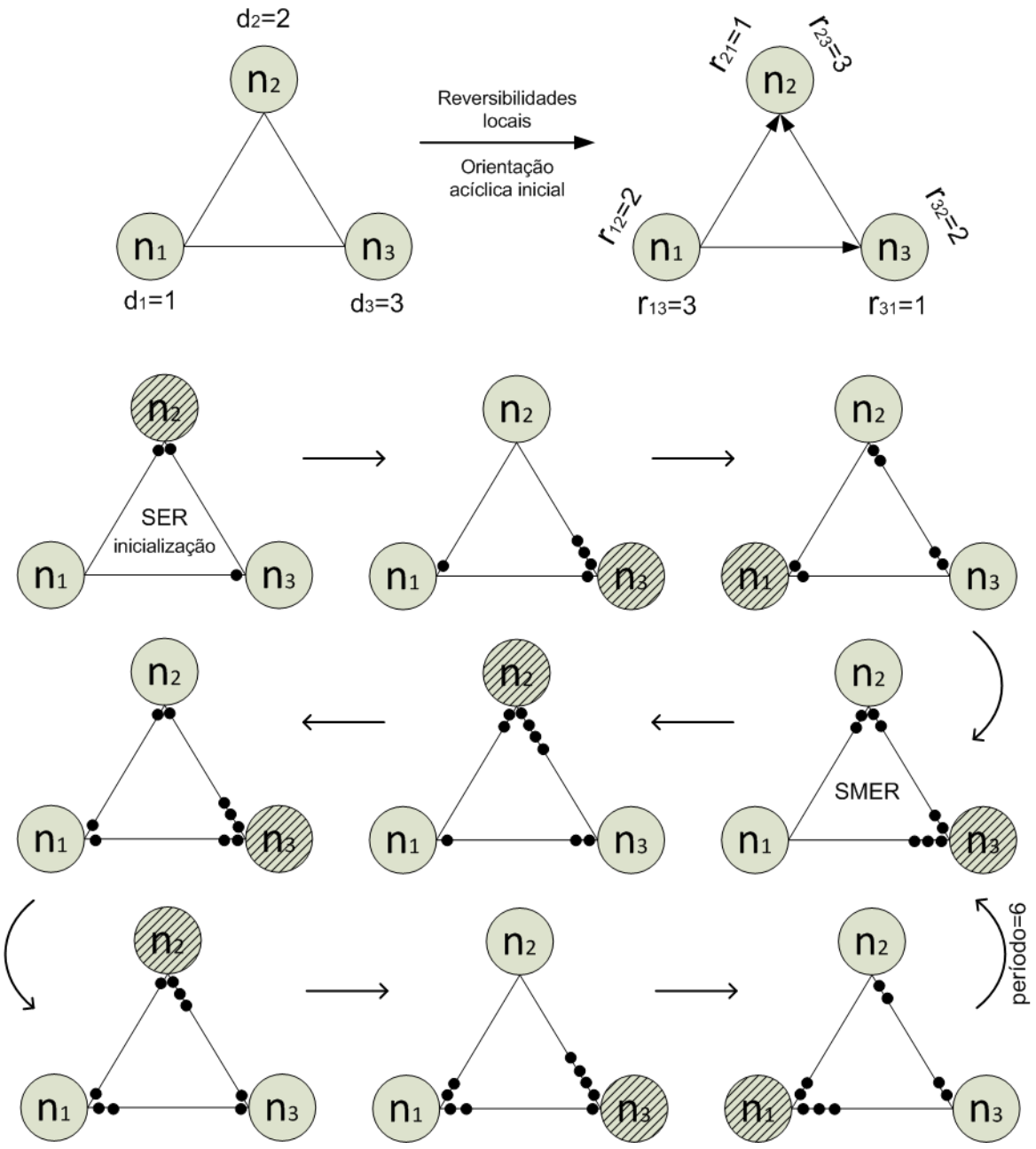


Figura 4.23: exemplo da inicialização do SER e posterior transição para o SMER. Formação de um período $p = 6$ no SMER.

cada orientação um novo conjunto de nós sinks SMER, que operam naquele instante. Este procedimento permite que a quantidade de orientações dada pelo período p seja mapeada para o número de slots de um quadro de transmissão, capaz de escoar todas as demandas dos nós da rede. Cada slot do quadro é uma orientação μ_s do multigrafo, e em cada orientação sabe-se que existe um conjunto de nós (pelo menos um nó) que opera, ou em outras palavras, nós que transmitem naquele slot. Este comportamento nada mais é do que o procedimento para um escalonamento TDMA, livre de colisões. A figura 4.24 reapresenta a figura 4.8 desta vez com o

quadro de transmissões que retrata em slots o escalonamento decorrente do período e da operação de nós a cada orientação do multigrafo. Pode-se verificar através do grafo que as demandas de transmissão dos nós foram atendidas de modo livre de colisões.

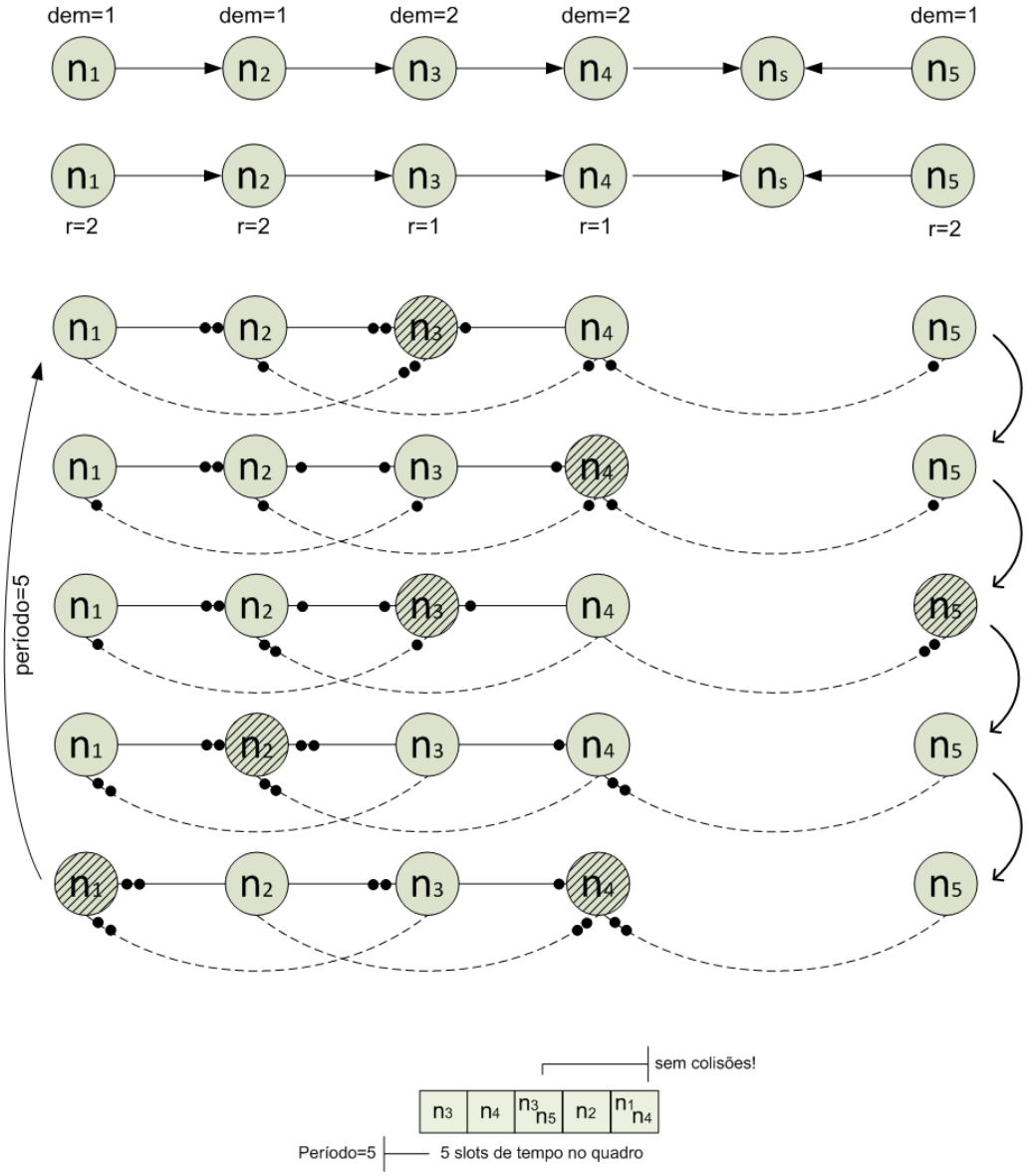


Figura 4.24: Descoberta do quadro de transmissões a partir do período da dinâmica do SMER.

Em uma visão local, cada nó identifica o seu período de operação pelo repetição de uma sequência de estados onde é verificado, além da sua condição de sink SMER ou não, se há a repetição individual dos estados das arestas que são compartilhadas com seus vizinhos na dinâmica do SMER. Localmente ocorre o mesmo ciclo de repetição de estados da condição global, e ainda verifica-se o cumprimento das transmissões equivalente a demanda neste período. Nos experimentos realizados os valores de períodos global e local se mostraram idênticos, entretanto não foi identi-

ficada a relação que conduz a tal resultado.

Esta seção encerra a apresentação da parte conceitual desta tese. O Capítulo 5 apresenta a seguir os resultados das implementações realizadas com fins de verificar e avaliar o funcionamento do mecanismo em diversos contextos e cenários.

4.2.3 Análise de Complexidade

A análise de complexidade do mecanismo de escalonamento é consequência das complexidades dos algoritmos que o formam, tais como a descoberta de rotas para os sinks, o algoritmo de orientação acíclica das arestas e a própria dinâmica do escalonamento por reversão de múltiplas arestas.

O algoritmo de descoberta de rotas para os sinks é baseado no algoritmo de busca em profundidade, ou seja, a partir de um nó sink as arestas são percorridas, sempre a partir do vértice encontrado mais recentemente e que ainda tenha arestas não exploradas saindo dele. Cada vértice é visitado uma vez e cada aresta é percorrida duas vezes, e portanto a complexidade de tempo do algoritmo é $O(|V| + |E|)$, sendo $|V|$ e $|E|$ relativos ao multigrafo de uma das versões do mecanismo.

A análise detalhada da complexidade dos algoritmos de orientação acíclica é encontrada na referência [29]. O algoritmo para orientação acíclica *Alg-Arestas* considera o lançamento de um dado de f faces e a comparação dos valores obtidos do sorteio do dado entre cada dois nós vizinhos. Em um sorteio, a probabilidade de empate entre dois nós vizinhos é dada por $1/f$. Quando o número de arestas é grande esta é a proporção de empates ocorridos a cada passo do algoritmo. Então, se a cada passo temos $1/f$ empates, o Alg-Arestas é capaz de gerar orientações acíclicas em $O(\log_f |E_x|)$, onde $|E_x|$ representa o número de total de arestas dadas por E_b ou a E_u, dependendo da escolha do multigrafo da versão broadcast ou da versão unicast, respectivamente. Cabe destacar que a escolha adequada de f no qual seu valor é muito maior que o número total de arestas $|E_b|$ ou $|E_u|$, leva a consideração de que o algoritmo converge em apenas um passo.

O segundo algoritmo para a realização da orientação acíclica, *Alg-Cor*, também tem seu funcionamento baseado em um sorteio de um dado de f faces, entretanto o nó deve comparar seu valor com toda a sua vizinhança, e somente será vencedor se o seu valor sorteado for maior que os demais nós vizinhos. Este funcionamento também pode ser beneficiado da escolha de f muito maior que o número de arestas do multigrafo, como apresentado na referência [30], demostrando que a complexidade do *Alg-Cor* é da ordem de $O(n)$.

O algoritmo de escalonamento por reversão de múltiplas arestas (SMER) tem seu funcionamento baseado na troca de mensagens entre os nós. A complexidade do encaminhamento das mensagens entre os nós sensores ocorre através das arestas,

decorrendo deste processo uma complexidade de comunicação da ordem de $O(|E|)$.

Capítulo 5

Cenários de Testes

As simulações realizadas para verificação e avaliação do funcionamento e desempenho do mecanismo de escalonamento foram realizadas no simulador OMNeT++ (*Objective Modular Network Testbed in C++*) [31], versão 4.2. O OMNET é um ambiente modular, baseado em eventos discretos, orientado à objeto, que fornece uma infraestrutura e diversas ferramentas para a simulação em redes. De modo simples, o modelo do OMNET consiste de módulos, representativos dos nós, que se comunicam através da troca de mensagens. Juntos integram um módulo maior, que define a própria *rede*. A topologia da rede e os atributos dos nós são descritos na linguagem NED enquanto que o comportamento de mensagens e nós durante a simulação são descritos em classes, implementadas na linguagem C++.

5.1 Considerações sobre as Implementações

Foram realizados duas implementações derivadas do mesmo código-fonte base de acordo com cada uma das versões do mecanismo: *broadcast* ou *unicast*. Existe um arquivo de configuração que permite ajustar os seguintes dados da simulação:

- número de nós;
- número de nós sinks;
- raio de comunicação de um nó (baseado na conectividade média);
- intervalo para o sorteio da demanda de um nó;
- algoritmo de orientação acíclica: Alg-Cor ou Alg-Arestas.

Além disto, há ainda a seleção entre as versões *broadcast* e *unicast* do mecanismo, que é feita a partir da utilização de uma das implementações.

É importante destacar que o objetivo primário desta simulação é mostrar a viabilidade da proposta, de modo que, com um cenário aleatório, definida a topologia

da rede, os nós fontes, os sinks e as demandas, cada nó seja capaz de construir, através da troca de informações com a sua vizinhança, um período de operação que passe a transmitir as demandas aos nós sinks sem que ocorram colisões.

Uma métrica utilizada para avaliar o desempenho do mecanismo após a determinação do período é a análise da *Concorrência* denominado γ.

Define-se γ como a relação entre a *demanda total* de dados que a rede precisa escoar, o número de nós da rede n e o *período* p, determinado pelo SMER.

$$\gamma_i = \frac{1}{np_i} \sum_{k=1}^{n} (d_k) \tag{5.1}$$

5.1.1 Topologia da Rede

A formação da topologia da rede a ser avaliada ocorre da seguinte forma: cada nó sorteia uniformemente as coordenadas x e y referentes à sua posição, em uma área lxl de $1000m$ x $1000m$. Após a inserção de n, os mesmos iniciam uma difusão estabelecendo arestas com outros nós dentro do raio de comunicação r_c, ou seja, cada nó descobre a vizinhança de um salto. O valor de r_c é consequência do modelo de Bernoulli para a distribuição randômica dos nós com uma conectividade[1] média K previamente definida. A Equação 5.2 apresenta a relação entre K, n, r_c e l:

$$K = \frac{\pi r_c^2}{l^2}(n-1) \tag{5.2}$$

Após estabelecidas as arestas de conectividade, o nó[0] inicia um algoritmo para verificar a existência de uma única componente conexa, ou seja, se a rede realmente conecta todos os nós estabelecidos inicialmente. Caso negativo, um novo sorteio é realizado produzindo um novo posicionamento para todos os nós. Este procedimento se repete até a formação de uma rede com uma única componente conexa.

Os nós sinks da rede podem ser definidos a priori, no momento em que os nós entram na rede, ou posteriormente, por algum nó central. Após determinar que a rede possui uma única componente conexa, o nó[0] escolhe aleatoriamente os nós que serão os sinks da rede, de acordo com o parâmetro do simulador. Este procedimento habilita em implementações futuras a conversão de um nó sensor para um sink pela existência de algoritmos que definam sinks mais eficientes de acordo com alguma métrica, como por exemplo, posicionamento geográfico central em um *cluster*.

Uma vez escolhidos os nós sinks, o nó[0] envia uma mensagem para cada um deles utilizando a árvore de difusão construída para detectar a existência de uma única componente conexa. Ao receber a mensagem o nó passa a operar como nó sink, como por exemplo, cancelando sua demanda de transmissão. Com isto, está

[1]Por *conectividade* entenda-se número de vizinhos.

formada a topologia da rede com sensores e sinks representada pelo grafo G.

A partir daí, a escolha pela utilização da implementação com a versão broadcast ou unicast, leva ao funcionamento do mecanismo descrito ao longo do Capítulo 4.

5.2 Resultados

5.2.1 Tabela de Escalonamento

Com o objetivo de acompanhar um experimento realizado, a figura 5.1 apresenta um *snapshot* da tela de uma simulação feita utilizando a versão broadcast, em um grafo com dez nós sendo destes, dois nós sinks, sorteados aleatoriamente. A orientação inicial foi realizada através do algoritmo Alg-Cor e as demandas dos nós variaram entre 1 e 5 pacotes de dados. Após a figura da simulação, a tabela A.1 e as demais tabelas do Apêndice A reproduzem a saída do programa computacional que detecta o período para o escalonamento.

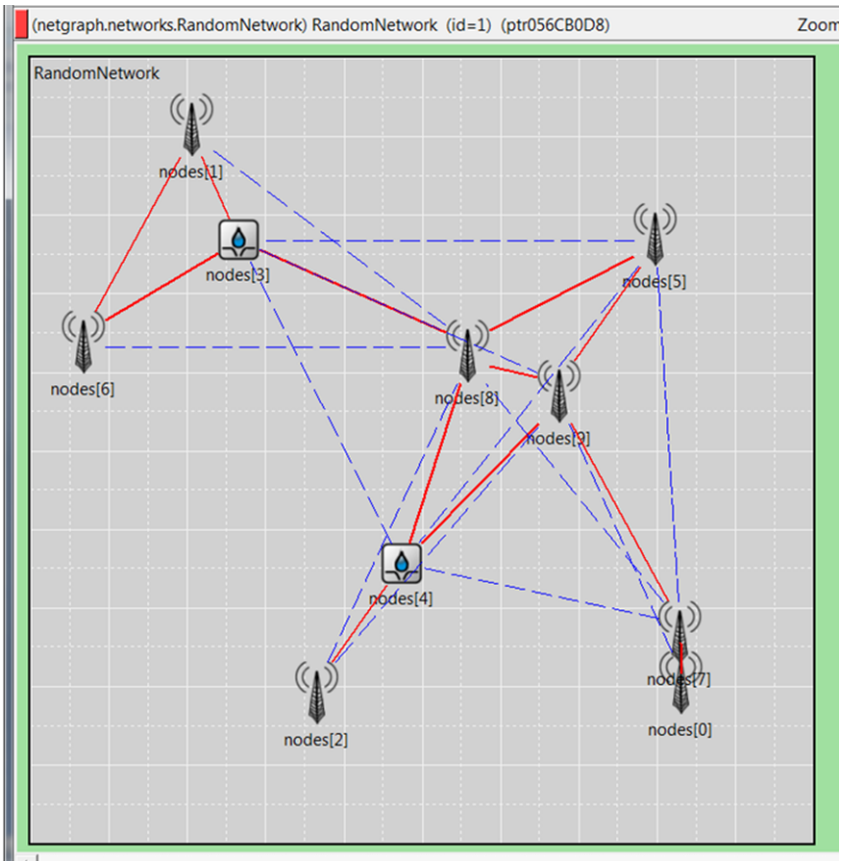


Figura 5.1: Instância da simulação no OMNET, versão broadcast, com 10 nós, 2 sinks, orientação pelo Alg-Cor e demanda entre 1 e 5.

A figura 5.2 ilustra o grafo representativo do experimento, destacando-se os nós sinks n_3 e n_4, escolhidos aleatoriamente, e as arestas de conectividade e interferência, desenhadas em linhas sólidas e tracejadas, respectivamente.

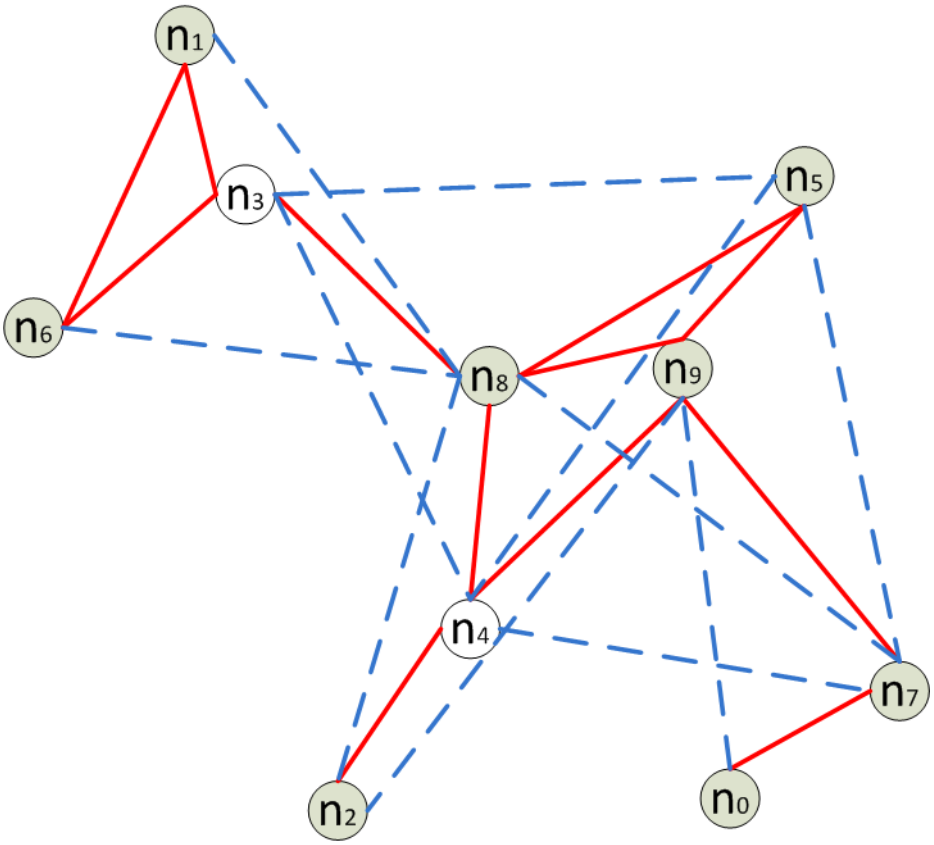


Figura 5.2: Desenho do grafo do experimento da Figura 5.1.

A figura 5.3 ilustra o multigrafo para a dinâmica SMER no experimento da Figura 5.1. É fácil observar a orientação acíclica inicial através das setas das arestas, mas sua referência está descrita na Tabela A.1 do Apêndice A como a primeira rodada do SMER, especificada na tabela como a *foto[0]* de cada nó. Uma *foto* é o registro do *status* do nó e da orientação das arestas do multigrafo que o nó compartilha com seus vizinhos. Nesta figura deixam de ser desenhadas as arestas incidentes aos nós sinks, uma vez que não há participação destas arestas na dinâmica do SMER.

Acompanhando as tabelas do Apêndice A, pode-se verificar que a partir da quinta orientação do multigrafo, representada na tabela A.2 pelo registro que especifica a *foto[4]* de cada nó, inicia-se a formação de um período de onze orientações, que se repete a partir do registro da *foto[15]*. Os momentos em que os nós operam estão assinaladas nas colunas.

A soma das demandas dos nós do grafo da Figura 5.1 é de 25 pacotes para transmissão. Esta demanda total foi alocada em 11 slots para transmissão conforme Figura 5.4. A título de informação sobre a extrapolação deste cenário relativamente simples, foram realizados experimentos com até 100 nós, associados a demandas totais para transmissão de mais de 3100 pacotes de dados que formaram quadros de até 800 slots de transmissão.

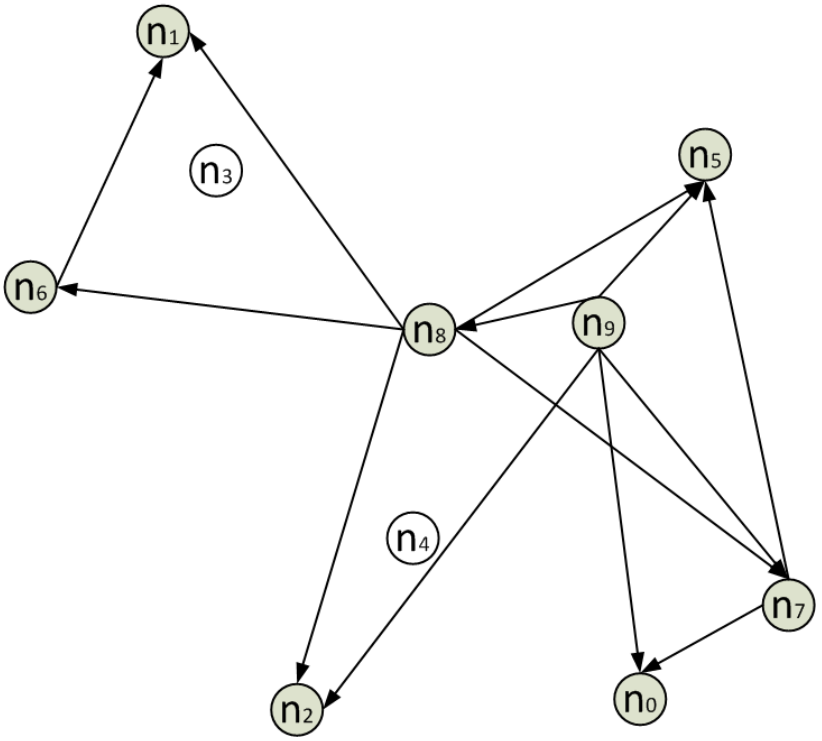


Figura 5.3: Desenho do multigrafo para dinâmica SMER do experimento da Figura 5.1.

Período SMER = 11
11 slots de tempo no quadro de transmissão

slot 1	slot 2	slot 3	slot 4	slot 5	slot 6	slot 7	slot 8	slot 9	slot 10	slot 11
n0 n8	n2 n6 n7	n0 n8	n0 n2 n5 n6	n8	n2 n6 n7	n0 n8	n0 n1 n2 n5	n6 n7	n8	n9

Figura 5.4: Tabela de escalonamento resultante do mecanismo broadcast no experimento da Figura 5.1. Este escalonamento é o resumo do resultado apresentado no Apêndice A.

A seguir estão apresentados resultados obtidos e análises realizadas a partir da variação de cenários dos experimentos. Junto a cada gráfico há a descrição do objetivo da simulação, a apresentação de uma tabela com as configurações do cenário avaliado e uma conclusão parcial.

5.2.2 Análise do Impacto da Versão do Mecanismo e da Conectividade na Concorrência

Objetivo da Simulação

As versões broadcast e unicast atuam de modo diferente na formação do grafo de interferência G_I e na relação com a vizinhança de um salto por ocasião da formação do multigrafo M de aplicação do SMER. O objetivo do gráfico da Figura 5.5 é comparar a concorrência nestas versões, avaliando ainda os efeitos da variação da conectividade entre $K = 4$ e $K = 8$ no resultado. Nestes experimentos os nós possuem demandas unitárias, fazendo desta avaliação aplicação análoga ao previsto no SER.

Número de nós:	variado de 10, 20, 30, 40, 50
Conectividade média:	$K = 4$ e $K = 8$
variação da demanda:	demanda unitária = 1
versão:	Comparação entre *Unicast* e *Broadcast*
orientação:	*Alg-Cor*

Tabela 5.1: Tabela com a configuração do cenário referente ao gráfico da Figura 5.5.

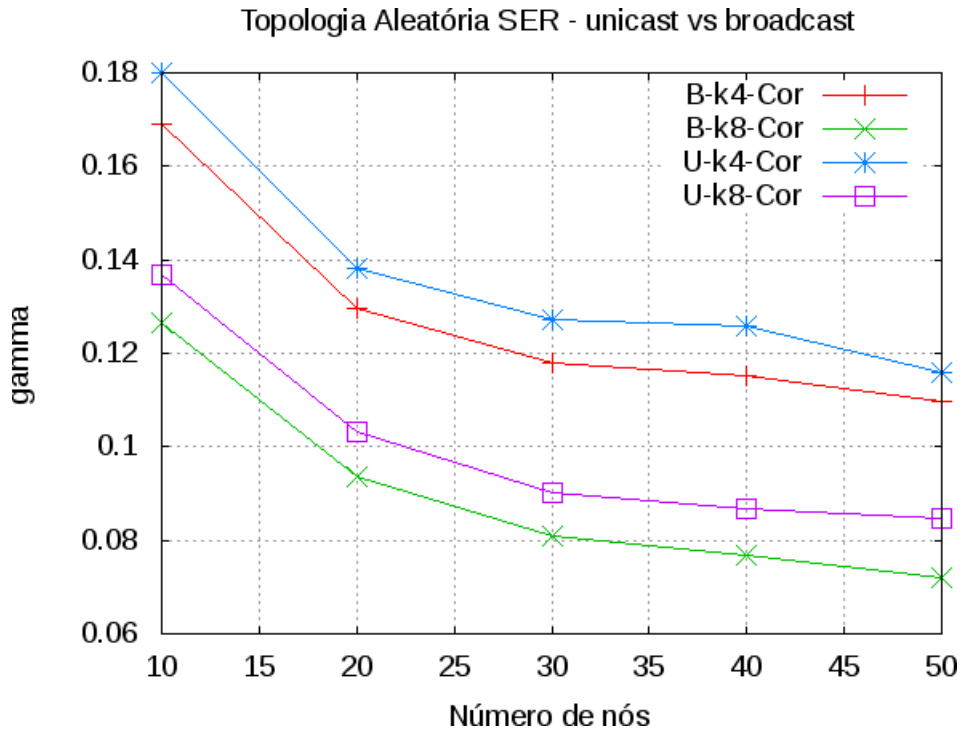


Figura 5.5: Análise da concorrência entre as versões e a partir da variação da conectividade média.

Conclusão Parcial

A principal percepção que se tem do resultado apresentado na Figura 5.5 é através da concorrência γ. Uma vez que a demanda é rigorosamente a mesma em cada valor de abscissa, o período médio reflete em γ que a versão unicast, em ambas as conectividades, tem um desempenho superior a versão broadcast, pela redução do tamanho do quadro produzido para o escalonamento. Outro aspecto analisado, a variação da conectividade entre $K = 4$ e $K = 8$, retrata que quanto menor a conectividade maior a concorrência, e por conseguinte, menores são os períodos.

5.2.3 Análise do Impacto do Algoritmo de Orientação Acíclica na Formação do Quadro de Transmissão

Objetivo da Simulação

O objetivo desta simulação, cujos resultados se apresentam nos gráficos das Figuras 5.6 e 5.7, é analisar a influência do algoritmo de orientação acíclica no tamanho do quadro de escalonamento, através da concorrência γ. Foram feitas simulações utilizando os algoritmos *Alg-Cor* e *Alg-Arestas*, estudados e comparados previamente na tese de doutorado da referência [29].

Número de nós:	10, 20, 30, 40, 50
Conectividade média:	$K = 4$ e $K = 8$
variação da demanda:	$1 - 5$
versão:	Broadcast
orientação acíclica:	Comparação entre *Alg-Cor* e *Alg-Arestas*

Tabela 5.2: Tabela com a configuração do cenário referente ao gráfico da Figura 5.6

Número de nós:	10, 20, 30, 40, 50
Conectividade média:	$K = 4$ e $K = 8$
variação da demanda:	$1 - 5$
versão:	Unicast
orientação acíclica:	Comparação entre *Alg-Cor* e *Alg-Arestas*

Tabela 5.3: Tabela com a configuração do cenário referente ao gráfico da Figura 5.7

Conclusão Parcial

Os resultados apresentados nos gráficos das Figuras 5.6 e 5.7 demonstram que a orientação acíclica inicial realizada através do algoritmo Alg-Cor produz maior concorrência do que aquela produzida pelo algoritmo Alg-Arestas. Esta diferença na concorrência inicial é mantida ao longo da variação do número de nós, o que traduz

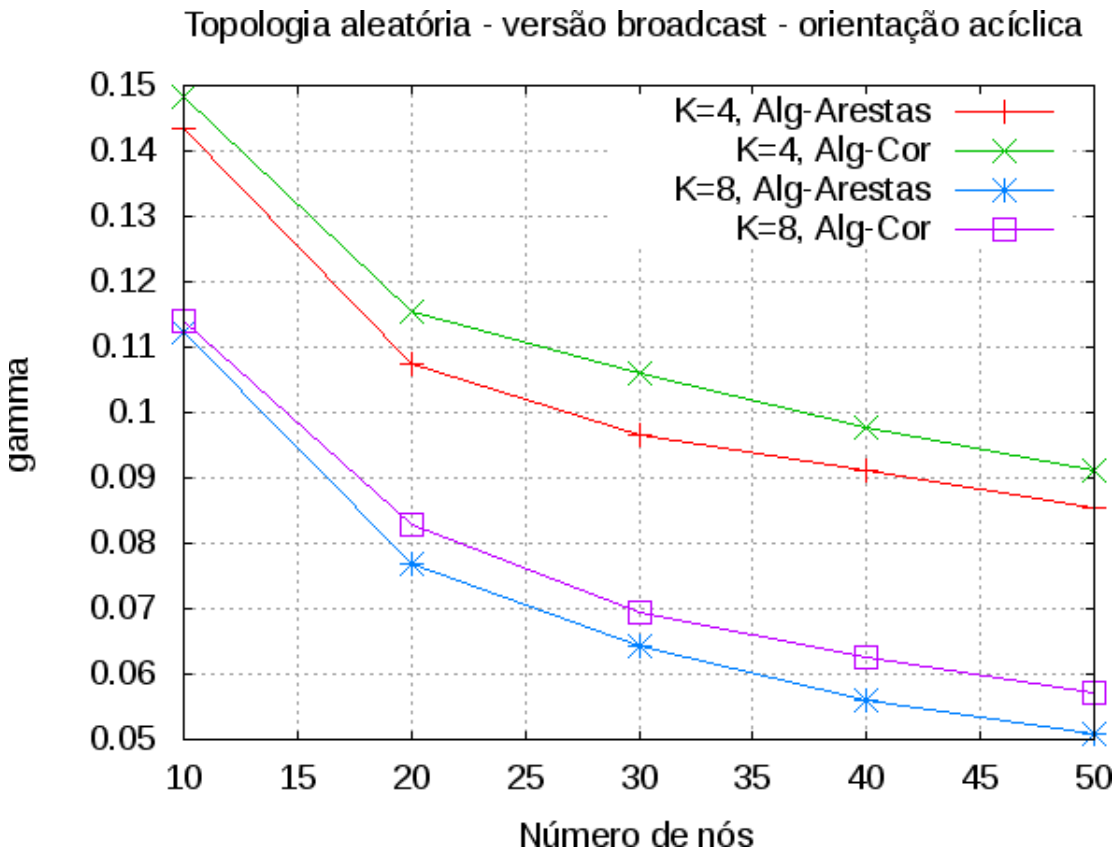


Figura 5.6: Comparação de resultados com alteração de algoritmo de orientação inicial - versão broadcast.

a importância da escolha do algoritmo de orientação acíclica. A relação da conectividade com o algoritmo de orientação acíclica não afeta o comportamento já visto na Seção 5.2.2.

5.2.4 Análise do Impacto da Conectividade na Formação do Quadro de Transmissão

Objetivo da Simulação

O aumento da conectividade tende a impactar o compartilhamento de recursos no sentido de reduzir o paralelismo de acessos. O objetivo desta simulação é verificar como se comporta este impacto em uma rede com poucos nós e em uma rede com muitos nós. Também deseja-se avaliar como este comportamento se dá quando utiliza-se uma ou outra versão do mecanismo. O gráfico da Figura 5.8 ilustra as curvas para as versões broadcast e unicast com 3 conectividades diferentes para cada uma das versões.

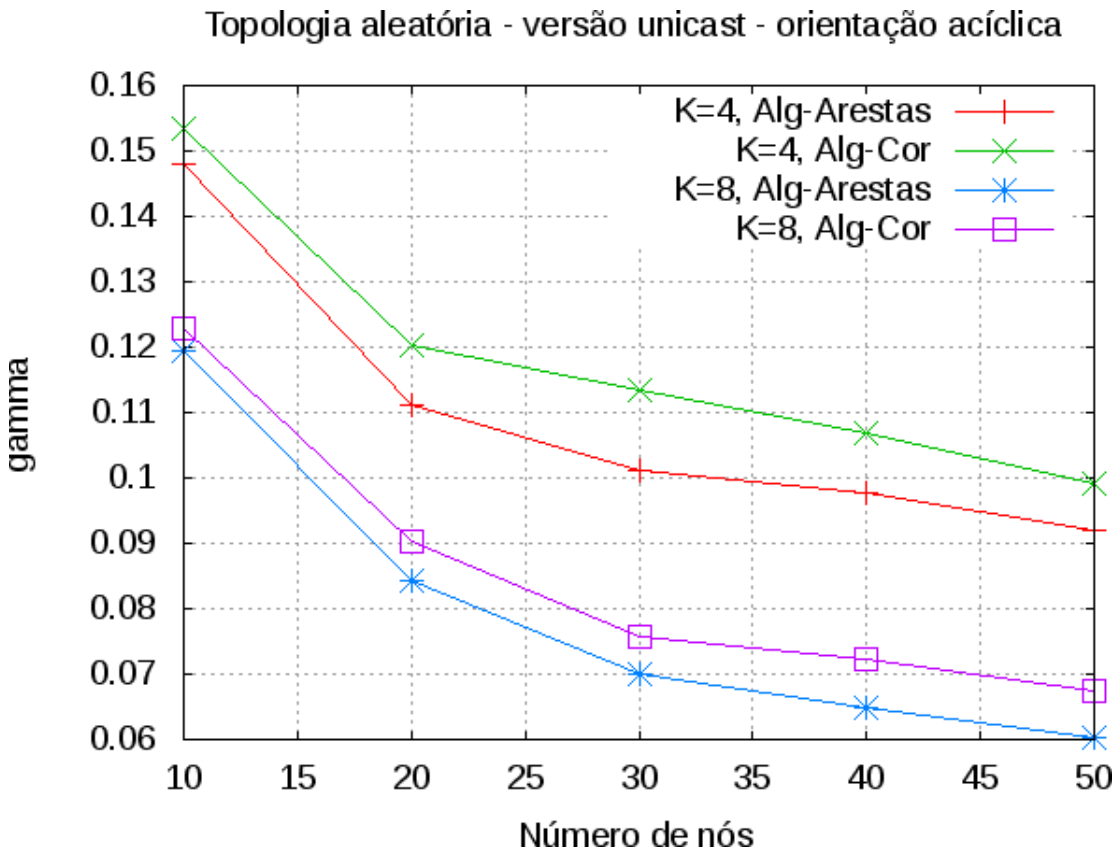


Figura 5.7: Comparação de resultados com alteração de algoritmo de orientação inicial - versão unicast.

Número de nós:	20, 40, 60, 80 e 100
Conectividade média:	$K = 6$, $K = 8$ e $K = 10$
variação de demanda:	$1 - 5$
versão:	*Broadcast* e *Unicast*
orientação:	*Alg-Cor*

Tabela 5.4: Tabela com a configuração do cenário referente ao gráfico da Figura 5.8.

Conclusão Parcial

O resultado desta simulação manteve as inferências anteriores em relação as consequências do aumento da conectividade, como visto nestes experimentos com $K = 6$, $K = 8$ e $K = 10$. Este aumento implica em uma redução da concorrência γ, produzindo maiores períodos e, consequentemente, quadros com maiores quantidades de slots. Ainda é possível verificar, considerando os gráficos da versão broadcast e unicast, que o desempenho da versão unicast com conectividade $K = 10$ é similar ao desempenho da versão broadcast com conectividade $K = 8$. Este resultado pode ser atribuído ao reuso espacial decorrente das regras do grafo de interferência para a versão unicast.

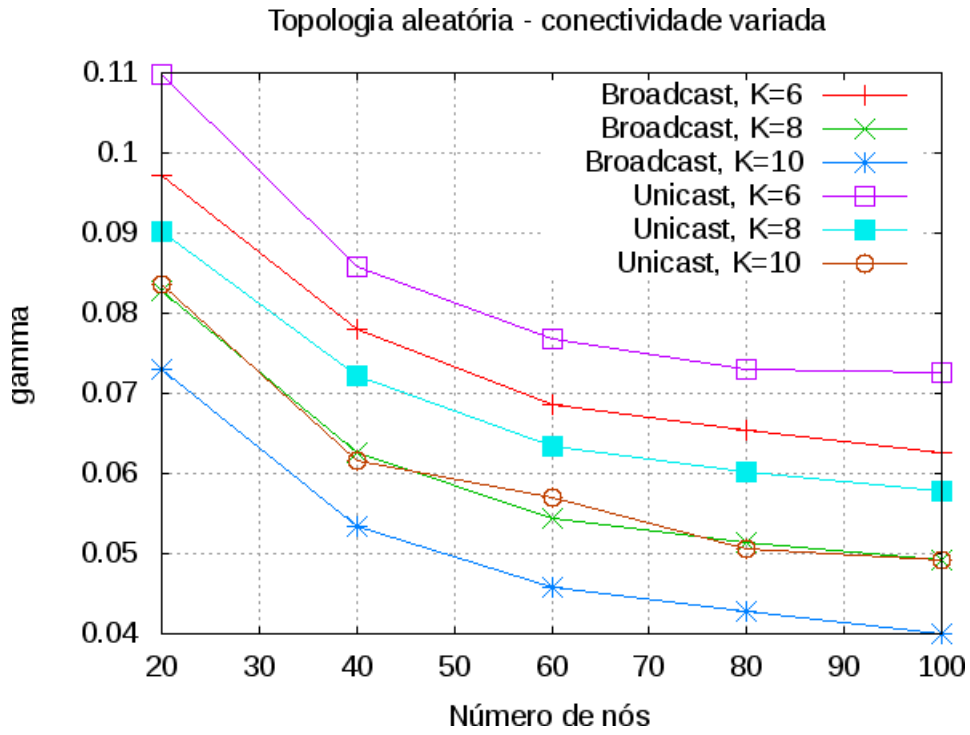


Figura 5.8: Avaliação de resultados de concorrência com a variação de versões, conectividades e do número de nós.

5.2.5 Análise do Impacto da *Versão do Mecanismo vs Múltiplos Sinks* na Formação do Quadro de Transmissão

Objetivo da Simulação

A possibilidade de múltiplos nós sinks é outro importante aspecto do mecanismo proposto neste trabalho. Dependendo da aplicação a escolha dos nós sinks pode atender a determinados critérios, como o posicionamento geográfico de interesse ou o centro de um *cluster* de nós. O objetivo desta simulação é verificar o impacto do aumento do número de nós sinks na geração de quadros e, por conexão, na concorrência. Os nós sinks desta simulação são escolhidos aleatoriamente, de maneira uniforme, e o experimento é realizado com cenários de 80 nós, com conectividade $K = 6$.

Número de nós:	variado de 20, 40, 60, 80, 100;
Conectividade média:	$K = 6$
variação do total de nós sinks:	1, 3, 5, 7, 9;
variação de demanda:	$1 - 10$
versão:	Comparação entre *Unicast* e *Broadcast*
orientação:	*Alg-Cor*

Tabela 5.5: Tabela com a configuração do cenário referente ao gráfico da Figura 5.9

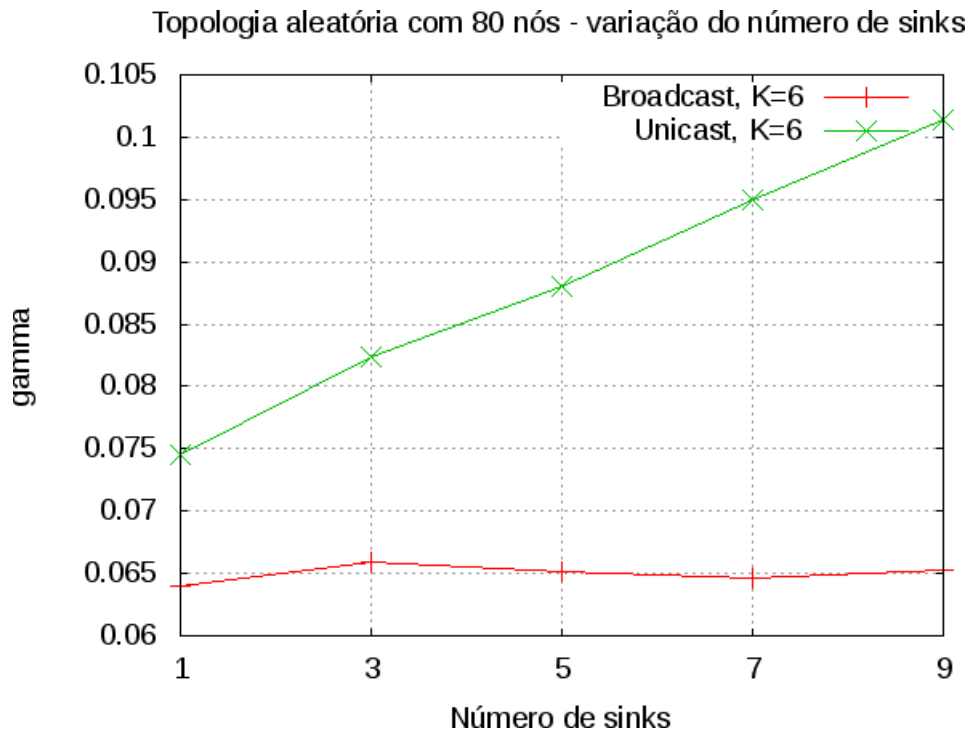


Figura 5.9: Comparação das versões do mecanismo em relação a variação do número de sinks.

Conclusão Parcial

Os resultados mostram que a variação dos nós sinks na versão broadcast não implica na variação da concorrência ou do tamanho do quadro. Este comportamento era esperado, uma vez que esta versão não utiliza informações dos sinks para a descoberta do período do nó e do escalonamento da rede. Por outro lado, a versão unicast se beneficia do emprego de múltiplos sinks, visto que um nó qualquer determina qual o caminho para o nó sink mais próximo e com esta informação passa a tratar seletivamente possíveis colisões para transmitir no caminho que leva ao sink.

5.2.6 Análise do Impacto do *Número de Nós vs Múltiplos Sinks* na Formação do Quadro de Transmissão

Objetivo da Simulação

O objetivo desta simulação é analisar o comportamento da concorrência, quando da variação do número de nós e da variação do número de sinks. Foram realizados experimentos com 40, 60 e 80 nós, todos com conectividade $K = 10$, na versão unicast.

Número de nós:	alternado entre 40, 60 e 80
Conectividade média:	$K = 10$
variação do total de nós sinks:	1, 3, 5, 7, 9;
variação de demanda:	$1 - 10$
versão:	*Unicast*
orientação:	Utilizando *Alg-Cor*

Tabela 5.6: Tabela descritiva da configuração referente ao gráfico da Figura 5.10

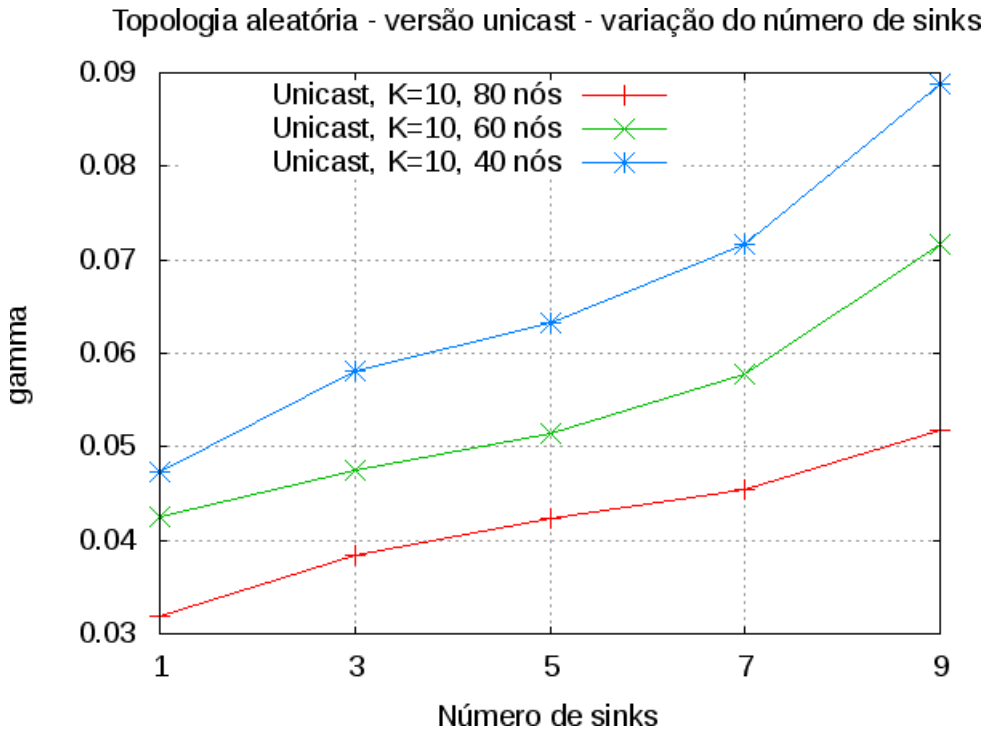


Figura 5.10: Comparação de resultados com a variação do número de nós em relação a variação do número de sinks.

Conclusão Parcial

O comportamento das curvas representativas do número de nós corroboram o resultado do gráfico da Figura 5.9, indicando que o aumento do número de nós sinks reduz o período resultante da dinâmica do SMER, aumentando a concorrência. Também fica aparente que as curvas representativas da quantidade de nós na rede resultam em concorrência maior do modo inversamente proporcional ao número de nós.

-

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Os estudos realizados para esta tese mostraram que a área de rede de sensores sem fio é vasta, está em permanente evolução e em busca de soluções para várias questões discutidas na comunidade científica.

Convergindo rapidamente destas questões em RSSF para o tema deste trabalho, os estudos teóricos realizados para o desenvolvimento desta tese mostrou como é crítico o consumo de energia na vida útil de uma rede de sensores, e por consequência, como são importantes soluções que diminuam o gasto de energia.

Alguns procedimentos para minimizar o consumo de energia foram destacados ao longo do texto. Diretamente, ou indiretamente, estes procedimentos impactam no escalonamento produzido pelo mecanismo proposto, particularmente pelo número de transmissões de um quadro. Dentre estes procedimentos destaca-se a agregação ou fusão de dados, que tem como uma das suas principais características o estudo para avaliar os custos das agregações e das transmissões visando minimizar o consumo de energia. Mostrou-se que agregações realizadas em áreas afastadas geograficamente dos sinks, particularmente em clusters, tendem a ser mais eficientes do ponto de vista do consumo de energia, mas podem produzir latências que não devem impactar na aplicação.

Os pesquisadores de rede de sensores em geral, assim como os trabalhos utilizados como referência nesta tese, possuem o consenso de que o maior responsável pelo consumo de energia em RSSF é o sistema de comunicação, tanto no que se refere a transmissão como a recepção. Por esta razão, cresce de importância soluções que permitam que os nós permaneçam o maior tempo possível no estado adormecido[1]. Entre estas soluções está o uso de protocolos com mecanismos de acesso ao meio baseados em escalonamento, como os apresentados em [6, 8, 10–12]. Ainda

[1]Também descrito como *stand-by*, *sleep* ou desligado, dependendo do protocolo.

dentre estes, somente os dois últimos não possuem contenções periódicas durante o funcionamento do escalonamento, similarmente ao mecanismo apresentado neste trabalho.

Ainda sob o enfoque da economia de energia, o mecanismo desta tese viabiliza o escalonamento em uma RSSF com a ocorrência de múltiplos sinks. A ocorrência de múltiplos sinks é um fator básico de economia, conforme citado nas referências [19, 20], pois também visa possibilitar a redução do número de saltos até o sink mais próximo para o envio dos dados de um monitoramento. Na versão unicast do mecanismo desta tese múltiplos sinks permitem um maior reuso espacial nas transmissões, e consequentemente, uma redução no tamanho do quadro de transmissões.

Os experimentos realizados no Capítulo anterior comprovam que é possível utilizar o SMER para gerar um escalonamento livre de colisões para uso do espectro eletromagnético. Verificou-se que a escolha do algoritmo de orientação acíclica influi na quantidade de concorrência, o que é refletido no tamanho do quadro.

Outro parâmetro estudado foram as conectividades. Notou-se que nas conectividades mais baixas, ($K = 4$), a versão unicast tem pouco destaque em relação a versão broadcast na formação de concorrências na transmissão. Isto decorre da existência de poucas arestas incidentes em um nó, e portanto, estas tendem a sempre participarem da dinâmica do SMER. Por outro lado, a medida que a conectividade aumenta, assim como o número de nós, o modelo de interferência da versão unicast permite um maior reuso espacial do espectro para as transmissões, destacando a versão unicast em relação a versão broadcast.

Outro aspecto importante a ser observado é o limite de abrangência da interferência para a inserção da aresta de interferência. Neste trabalho as arestas de interferência seguiram um padrão muito utilizado na literatura de considerar a distância em que há interferência igual ao raio de comunicação de um nó, ou seja, o raio em que uma transmissão realizada é plenamente recebida por um receptor. Entretanto, é sabido que por razões de sensibilidade dos receptores rádios, a distância entre a fonte de emissão e a distância de recepção com integridade da informação é distinta da distância até onde uma emissão pode sensibilizar um outro receptor e interferir em outras transmissões. A abordagem que considera estes valores de raio de comunicação e raio de interferência diferentes, pode ser realizada neste modelo do mecanismo de escalonamento deste trabalho. Para tal, basta definir um raio de interferência diferente do raio de comunicação estipulado para o nó, e assim, realizar a inserção das arestas de interferência baseada nos limites dado pelo valor desejado de raio de interferência.

6.2 Trabalhos Futuros

Ao longo do desenvolvimento desta tese outros cenários e novas perspectivas de abordagens foram visualizadas para a aplicação dos conceitos do SMER e de variações do mecanismo deste trabalho. Uma proposta aos estudos em redes de sensores sem fio que foi identificada, particularmente pela pouca existência de literatura científica neste sentido, é o que trata de múltiplos sinks ortogonais em rede de sensores sem fio.

A idéia por trás da utilização de tráfegos ortogonais é a realização de monitoração de diferentes variáveis ambientais, ou mesmo diferentes tipos de sensores, através de uma rede única capaz de transmitir os tipos de dados coletados para os correspondentes sinks, através da mesma infra-estrutura de rede. Um exemplo deste cenário é uma rede que possui vários tipos de sensores, como nós sensores que coletam dados de temperatura, nós sensores que coletam dados de umidade, nós sensores que coletam dados de luminosidade, e até sensores que coletam mais do que um tipo de dado. Estas informações devem ser dirigidas para múltiplos sinks, podendo estes reunirem dados de somente um tipo ou de mais de um tipo.

O conceito de hiper-nó visto no Capítulo 4, onde um nó é composto de nós internos, pode ser aplicado na solução deste tipo de problema, considerando que cada nó interno lida com um tipo de monitoramento. A versão unicast também se destaca neste cenário, pois a seletividade da transmissão pode direcionar o tipo de dado monitorado para o vizinho mais adequado.

Referências Bibliográficas

[1] HALL, D., LLINAS, J. *Handbook of Multisensor Data Fusion*. CRC Press, 2001.

[2] AKYILDIZ, I. F., SU, W., SANKARASUBRAMANIAM, Y., et al. "Wireless Sensor Networks: A Survey", *Computer Networks*, v. 38, pp. 393–422, 2002.

[3] LOUREIRO, A. A. F., NOGUEIRA, J. M. S., RUIZ, L. B., et al. "Rede de Sensores sem fio", *XXI Simpósio Brasileiro de Redes de Computadores*, pp. 179–226, 2003.

[4] LUO, H., LIU, YONGHE ANDDAS, S. K. "Routing Correlated Data with Fusion Cost in Wireless Sensor Networks", *IEEE Transactions on Mobile Computing*, v. 5, n. 11, pp. 1620–1632, 2006.

[5] YE, W., HEIDEMANN, J., ESTRIN, D. "An energy-efficient MAC protocol for wireless sensor networks". In: *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, v. 3, pp. 1567 – 1576, 2002.

[6] RAJENDRAN, V., OBRACZKA, K., GARCIA-LUNA-ACEVES, J. J. "Energy-Efficient Collision-Free Medium Access Control for Wireless Sensor Networks". In: *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pp. 181–192, New York, NY, USA, 2003. ACM.

[7] NIEBERG, T., DULMAN, S., HAVINGA, P., et al. "Collaborative Algorithms for Communication in Wireless Sensor Networks". In: *Ambient Intelligence: Impact on Embedded Sytem Design*, Springer US, pp. 271–294, 2004.

[8] VAN HOESEL, L. F. W., HAVINGA, P. J. M. "A Lightweight Medium Access Protocol (LMAC) for Wireless Sensor Networks: Reducing Preamble Transmissions and Transceiver State Switches". In: *1st International*

Workshop on Networked Sensing Systems (INSS, Tokio, Japan, pp. 205–208, Tokio, Japan, 2004. Society of Instrument and Control Engineers (SICE).

[9] RAJENDRAN, V., GARCIA-LUNA-AVECES, J., OBRACZKA, K. "Energy-efficient, application-aware medium access for sensor networks". In: *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pp. 8 pp. 622–630, nov. 2005.

[10] VAN HOESEL, L., HAVINGA, P. "Collision-free Time Slot Reuse in Multi-hop Wireless Sensor Networks". In: *Proceedings of the 2005 International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 101–107, 2005.

[11] GANDHAM, S., DAWANDE, M., PRAKASH, R. "Link scheduling in sensor networks: distributed edge coloring revisited". In: *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, v. 4, pp. 2492 – 2501 vol. 4, march 2005.

[12] PINHO, A., SANTOS, A., FIGUEIREDO, D., et al. "Two ID-Free Distributed Distance-2 Edge Coloring Algorithms for WSNs". In: *NETWORKING 2009*, v. 5550, *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 919–930, 2009.

[13] BARBOSA, V. C., BENEVIDES, M. R. F., FRANÇA, F. M. G. "Sharing Resources at Nonuniform Access Rates", *Theory Comput. Syst.*, v. 34, n. 1, pp. 13–26, 2001.

[14] NAKAMURA, E. F., LOUREIRO, A. A. F., FRERY, A. C. "Information Fusion for Wireless Sensor Networks: Methods, Models, and Classifications", *ACM Comput. Surv.*, v. 39, n. 3, pp. 9, 2007.

[15] SANTOS, A. A., WAINER, J. "Inferência Abdutiva na Avaliação de Ameaças na Defesa Aeroespacial". In: *Anais do IV ENIA/SBC*, v. 1. SBC, August 2003.

[16] LUO, H., LUO, J., LIU, Y., et al. "Adaptive Data Fusion for Energy Efficient Routing in Wireless Sensor Networks", *IEEE Trans. Comput.*, v. 55, n. 10, pp. 1286 – 1299, 2006.

[17] HEINZELMAN, W., CHANDRAKASAN, A., BALAKRISHNAN, H. "Energy-efficient communication protocol for wireless microsensor networks". In: *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, p. 10 pp. vol.2, jan. 2000.

[18] KRISHNAMACHARI, L., ESTRIN, D., WICKER, S. "The impact of data aggregation in wireless sensor networks". In: *Distributed Computing Systems Workshops, 2002. Proceedings. 22nd International Conference on*, pp. 575 – 578, 2002.

[19] CHEN, S., COOLBETH, M., DINH, H., et al. "Data Collection with Multiple Sinks in Wireless Sensor Networks". In: *Wireless Algorithms, Systems, and Applications*, v. 5682, *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 284–294, 2009.

[20] DAS, A., DUTTA, D. "Data acquisition in multiple-sink sensor networks", *SIGMOBILE Mob. Comput. Commun. Rev.*, v. 9, pp. 82–85, July 2005.

[21] ABRAMSON, N. "THE ALOHA SYSTEM: another alternative for computer communications". In: *Proceedings of the November 17-19, 1970, fall joint computer conference*, AFIPS '70 (Fall), pp. 281–285, New York, NY, USA, 1970. ACM.

[22] LI, G., DOSS, R. "Energy-Efficient Medium Access Control in Wireless Sensor Networks". In: *Guide to Wireless Sensor Networks*, Computer Communications and Networks, Springer London, pp. 419–438, 2009.

[23] YE, W., HEIDEMANN, J., ESTRIN, D. "An energy-efficient MAC protocol for wireless sensor networks". In: *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, v. 3, pp. 1567 – 1576 vol.3, 2002.

[24] BARBOSA, V. C., GAFNI, E. "Concurrency in Heavily Loaded Neighborhood-Constrained Systems", *ACM Trans. Program. Lang. Syst.*, v. 11, n. 4, pp. 562–584, 1989.

[25] BARBOSA, V. C. *An introduction to distributed algorithms*. Cambridge, MA, USA, MIT Press, 1996. ISBN: 0-262-02412-8.

[26] GAFNI, E. M., BERTSEKAS, D. "Distributed Algorithms for Generating Loop-free Routes in Networks with Frequently Changing Topology". In: *IEEE Trans. Commun. COM-29*, 1981.

[27] DIJKSTRA, E. W. "Cooperating Sequential Processes", *Programming Languages*, pp. 43–112, 1968.

[28] PAILLARD, G., LAVAULT, C., FRANCA, F. "A Distributed Prime Sieving Algorithm based on Scheduling by Multiple Edge Reversal". In: *ISPDC '05: Proceedings of The 4th International Symposium on Parallel and*

Distributed Computing, pp. 139–146, Washington, DC, USA, 2005. IEEE Computer Society.

[29] ARANTES JR, G. M. *Trilhas, Otimização de Concorrência e Inicialização Probabilística em Sistemas sob Reversão de Arestas.* Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2006.

[30] ARANTES, JR., G. M., FRANÇA, F. M. G., MARTINHON, C. A. "Randomized generation of acyclic orientations upon anonymous distributed systems", *J. Parallel Distrib. Comput.*, v. 69, n. 3, pp. 239–246, 2009.

[31] VARGA, A., HORNIG, R. "An overview of the OMNeT++ simulation environment". In: *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, Simutools '08, pp. 60:1–60:10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

Apêndice A

Escalonamento do Experimento da Figura 5.1

No[0]	No[1]	No[2]	No [3]	No [4]	No[5]	No[6]	No[7]	No[8]	No[9]
a demanda de nodes[0] é 5	a demanda de nodes[1] é 1	a demanda de nodes[2] é 4			a demanda de nodes[5] é 2	a demanda de nodes[6] é 4	a demanda de nodes[7] é 3	a demanda de nodes[8] é 5	a demanda de nodes[9] é 1
reversibilidade com o nó 7 eh 3 reversibilidade com o nó 9 eh 1	reversibilidade com o nó 6 eh 4 reversibilidade com o nó 8 eh 5	reversibilidade com o nó 8 eh 5 reversibilidade com o nó 9 eh 1			reversibilidade com o nó 8 eh 5 reversibilidade com o nó 9 eh 1 reversibilidade com o nó 7 eh 3	reversibilidade com o nó 1 eh 1 reversibilidade com o nó 8 eh 5	reversibilidade com o nó 0 eh 5 reversibilidade com o nó 9 eh 1 reversibilidade com o nó 5 eh 2 reversibilidade com o nó 8 eh 5	reversibilidade com o nó 5 eh 2 reversibilidade com o nó 9 eh 1 reversibilidade com o nó 1 eh 1 reversibilidade com o nó 6 eh 4 reversibilidade com o nó 2 eh 4 reversibilidade com o nó 7 eh 3	reversibilidade com o nó 5 eh 2 reversibilidade com o nó 7 eh 3 reversibilidade com o nó 8 eh 5 reversibilidade com o nó 2 eh 4 reversibilidade com o nó 0 eh 5
SmerPos[4 , 6 , 7 , 10 , 11 , 15 , 17 , 18 , 21 , 22]	SmerPos[11 , 22 , 33]	SmerPos[5 , 7 , 9 , 11 , 16 , 18 , 20 , 22]			SmerPos[7 , 11 , 18 , 22]	SmerPos[5 , 7 , 9 , 12 , 16 , 18 , 20 , 23]	SmerPos[5 , 9 , 12 , 16 , 20 , 23]	SmerPos[2 , 4 , 6 , 8 , 10 , 13 , 15 , 17 , 19 , 21 , 24]	SmerPos[14 , 25 , 36]
PERIODO ------->>> 11 from node nodes[0]	PERIODO ------->>> 11 from node nodes[1]	PERIODO ------->>> 11 from node nodes[2]			PERIODO ------->>> 11 from node nodes[5]	PERIODO ------->>> 11 from node nodes[6]	PERIODO ------->>> 11 from node nodes[7]	PERIODO ------->>> 11 from node nodes[8]	PERIODO ------->>> 11 from node nodes[9]
Foto[0] Neighbor[7].tokens 1 Neighbor[9].tokens 1	Foto[0] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 1	Foto[0] Neighbor[4].tokens 0 Neighbor[8].tokens 1 Neighbor[9].tokens 1			Foto[0] Neighbor[8].tokens 1 Neighbor[9].tokens 1 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 1	Foto[0] Neighbor[1].tokens 0 Neighbor[3].tokens 0 Neighbor[8].tokens 1	Foto[0] Neighbor[0].tokens 0 Neighbor[9].tokens 1 Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[8].tokens 1	Foto[0] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[9].tokens 1 Neighbor[1].tokens 0 Neighbor[6].tokens 0 Neighbor[2].tokens 0 Neighbor[7].tokens 0	Foto[0] Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[7].tokens 0 Neighbor[8].tokens 0 Neighbor[2].tokens 0 Neighbor[3].tokens 0 Neighbor[0].tokens 0
Foto[1] Neighbor[7].tokens 0 Neighbor[9].tokens 0	Foto[1] Neighbor[3].tokens 0 Neighbor[6].tokens 0 Neighbor[8].tokens 0	Foto[1] Neighbor[4].tokens 0 Neighbor[8].tokens 0 Neighbor[9].tokens 0			Foto[1] Neighbor[8].tokens 0 Neighbor[9].tokens 0 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 0	Foto[1] Neighbor[1].tokens 4 Neighbor[3].tokens 0 Neighbor[8].tokens 1	Foto[1] Neighbor[0].tokens 3 Neighbor[9].tokens 1 Neighbor[4].tokens 0 Neighbor[5].tokens 3 Neighbor[8].tokens 1	Foto[1] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 5 Neighbor[9].tokens 1 Neighbor[1].tokens 5 Neighbor[6].tokens 0 Neighbor[2].tokens 5 Neighbor[7].tokens 0	Foto[1] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 0 Neighbor[8].tokens 0 Neighbor[2].tokens 1 Neighbor[3].tokens 0 Neighbor[0].tokens 1
Foto[2] Neighbor[7].tokens 5 Neighbor[9].tokens 0	Foto[2] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 0	Foto[2] Neighbor[4].tokens 0 Neighbor[8].tokens 0 Neighbor[9].tokens 0			Foto[2] Neighbor[8].tokens 0 Neighbor[9].tokens 0 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 2	Foto[2] Neighbor[1].tokens 3 Neighbor[3].tokens 0 Neighbor[8].tokens 0	Foto[2] Neighbor[0].tokens 2 Neighbor[9].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[8].tokens 0	Foto[2] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 5 Neighbor[9].tokens 1 Neighbor[1].tokens 5 Neighbor[6].tokens 5 Neighbor[2].tokens 5 Neighbor[7].tokens 5	Foto[2] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 1 Neighbor[8].tokens 0 Neighbor[2].tokens 1 Neighbor[3].tokens 0 Neighbor[0].tokens 1

Figura A.1: Figura com extrato da tabela de escalonamento - parte 1.

Figura A.2: Figura com extrato da tabela de escalonamento - parte 2.

Figura A.3: Figura com extrato da tabela de escalonamento - parte 3.

Foto[13] Neighbor[7].tokens 5 Neighbor[9].tokens 0	Foto[13] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 0	Foto[13] Neighbor[4].tokens 0 Neighbor[8].tokens 0 Neighbor[9].tokens 0		Foto[13] Neighbor[8].tokens 0 Neighbor[9].tokens 0 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 2	Foto[13] Neighbor[1].tokens 3 Neighbor[3].tokens 0 Neighbor[8].tokens 0	Foto[13] Neighbor[0].tokens 2 Neighbor[9].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[8].tokens 0	Foto[13] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 6 Neighbor[9].tokens 1 Neighbor[1].tokens 5 Neighbor[6].tokens 8 Neighbor[2].tokens 8 Neighbor[7].tokens 7	Foto[13] Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[7].tokens 3 Neighbor[8].tokens 4 Neighbor[2].tokens 4 Neighbor[3].tokens 0 Neighbor[0].tokens 5
Foto[14] Neighbor[7].tokens 5 Neighbor[9].tokens 0	Foto[14] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 1	Foto[14] Neighbor[4].tokens 0 Neighbor[8].tokens 4 Neighbor[9].tokens 0		Foto[14] Neighbor[8].tokens 2 Neighbor[9].tokens 0 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 2	Foto[14] Neighbor[1].tokens 3 Neighbor[3].tokens 0 Neighbor[8].tokens 4	Foto[14] Neighbor[0].tokens 2 Neighbor[9].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[8].tokens 3	Foto[14] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 4 Neighbor[9].tokens 0 Neighbor[1].tokens 4 Neighbor[6].tokens 4 Neighbor[2].tokens 4 Neighbor[7].tokens 4	Foto[14] Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[7].tokens 3 Neighbor[8].tokens 5 Neighbor[2].tokens 4 Neighbor[3].tokens 0 Neighbor[0].tokens 5
Foto[15] Neighbor[7].tokens 5 Neighbor[9].tokens 5	Foto[15] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 1	Foto[15] Neighbor[4].tokens 0 Neighbor[8].tokens 4 Neighbor[9].tokens 4		Foto[15] Neighbor[8].tokens 2 Neighbor[9].tokens 2 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 2	Foto[15] Neighbor[1].tokens 3 Neighbor[3].tokens 0 Neighbor[8].tokens 4	Foto[15] Neighbor[0].tokens 2 Neighbor[9].tokens 3 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[8].tokens 3	Foto[15] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 4 Neighbor[9].tokens 5 Neighbor[1].tokens 4 Neighbor[6].tokens 4 Neighbor[2].tokens 4 Neighbor[7].tokens 4	Foto[15] Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[7].tokens 0 Neighbor[8].tokens 0 Neighbor[2].tokens 0 Neighbor[3].tokens 0 Neighbor[0].tokens 0
Foto[16] Neighbor[7].tokens 2 Neighbor[9].tokens 4	Foto[16] Neighbor[3].tokens 0 Neighbor[6].tokens 1 Neighbor[8].tokens 2	Foto[16] Neighbor[4].tokens 0 Neighbor[8].tokens 8 Neighbor[9].tokens 4		Foto[16] Neighbor[8].tokens 4 Neighbor[9].tokens 2 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 2	Foto[16] Neighbor[1].tokens 3 Neighbor[3].tokens 0 Neighbor[8].tokens 8	Foto[16] Neighbor[0].tokens 5 Neighbor[9].tokens 3 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[8].tokens 6	Foto[16] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[9].tokens 4 Neighbor[1].tokens 3 Neighbor[6].tokens 0 Neighbor[2].tokens 0 Neighbor[7].tokens 1	Foto[16] Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[7].tokens 0 Neighbor[8].tokens 1 Neighbor[2].tokens 0 Neighbor[3].tokens 0 Neighbor[0].tokens 1
Foto[17] Neighbor[7].tokens 7 Neighbor[9].tokens 4	Foto[17] Neighbor[3].tokens 0 Neighbor[6].tokens 2 Neighbor[8].tokens 2	Foto[17] Neighbor[4].tokens 0 Neighbor[8].tokens 3 Neighbor[9].tokens 3		Foto[17] Neighbor[8].tokens 4 Neighbor[9].tokens 2 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 4	Foto[17] Neighbor[1].tokens 2 Neighbor[3].tokens 0 Neighbor[8].tokens 3	Foto[17] Neighbor[0].tokens 0 Neighbor[9].tokens 2 Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[8].tokens 1	Foto[17] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 2 Neighbor[9].tokens 4 Neighbor[1].tokens 3 Neighbor[6].tokens 5 Neighbor[2].tokens 5 Neighbor[7].tokens 6	Foto[17] Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[7].tokens 1 Neighbor[8].tokens 1 Neighbor[2].tokens 1 Neighbor[3].tokens 0 Neighbor[0].tokens 1

Figura A.4: Figura com extrato da tabela de escalonamento - parte 4.

Foto[18] Neighbor[7].tokens 4 Neighbor[9].tokens 3	Foto[18] Neighbor[3].tokens 0 Neighbor[6].tokens 2 Neighbor[8].tokens 3	Foto[18] Neighbor[4].tokens 0 Neighbor[8].tokens 7 Neighbor[9].tokens 3		Foto[18] Neighbor[8].tokens 6 Neighbor[9].tokens 2 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 4	Foto[18] Neighbor[1].tokens 2 Neighbor[3].tokens 0 Neighbor[8].tokens 7	Foto[18] Neighbor[0].tokens 3 Neighbor[9].tokens 2 Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[8].tokens 4	Foto[18] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[9].tokens 3 Neighbor[1].tokens 2 Neighbor[6].tokens 1 Neighbor[2].tokens 1 Neighbor[7].tokens 3	Foto[18] Neighbor[4].tokens 0 Neighbor[5].tokens 0 Neighbor[7].tokens 1 Neighbor[8].tokens 2 Neighbor[2].tokens 1 Neighbor[3].tokens 0 Neighbor[0].tokens 2
Foto[19] Neighbor[7].tokens 1 Neighbor[9].tokens 2	Foto[19] Neighbor[3].tokens 0 Neighbor[6].tokens 3 Neighbor[8].tokens 3	Foto[19] Neighbor[4].tokens 0 Neighbor[8].tokens 2 Neighbor[9].tokens 2		Foto[19] Neighbor[8].tokens 1 Neighbor[9].tokens 1 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 1	Foto[19] Neighbor[1].tokens 1 Neighbor[3].tokens 0 Neighbor[8].tokens 2	Foto[19] Neighbor[0].tokens 6 Neighbor[9].tokens 2 Neighbor[4].tokens 0 Neighbor[5].tokens 3 Neighbor[8].tokens 4	Foto[19] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 5 Neighbor[9].tokens 3 Neighbor[1].tokens 2 Neighbor[6].tokens 6 Neighbor[2].tokens 6 Neighbor[7].tokens 3	Foto[19] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 1 Neighbor[8].tokens 2 Neighbor[2].tokens 2 Neighbor[3].tokens 0 Neighbor[0].tokens 3
Foto[20] Neighbor[7].tokens 1 Neighbor[9].tokens 2	Foto[20] Neighbor[3].tokens 0 Neighbor[6].tokens 3 Neighbor[8].tokens 4	Foto[20] Neighbor[4].tokens 0 Neighbor[8].tokens 6 Neighbor[9].tokens 2		Foto[20] Neighbor[8].tokens 3 Neighbor[9].tokens 1 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 1	Foto[20] Neighbor[1].tokens 1 Neighbor[3].tokens 0 Neighbor[8].tokens 6	Foto[20] Neighbor[0].tokens 6 Neighbor[9].tokens 2 Neighbor[4].tokens 0 Neighbor[5].tokens 3 Neighbor[8].tokens 7	Foto[20] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 3 Neighbor[9].tokens 2 Neighbor[1].tokens 1 Neighbor[6].tokens 2 Neighbor[2].tokens 2 Neighbor[7].tokens 0	Foto[20] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 1 Neighbor[8].tokens 3 Neighbor[2].tokens 2 Neighbor[3].tokens 0 Neighbor[0].tokens 3
Foto[21] Neighbor[7].tokens 6 Neighbor[9].tokens 2	Foto[21] Neighbor[3].tokens 0 Neighbor[6].tokens 4 Neighbor[8].tokens 4	Foto[21] Neighbor[4].tokens 0 Neighbor[8].tokens 1 Neighbor[9].tokens 1		Foto[21] Neighbor[8].tokens 3 Neighbor[9].tokens 1 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 3	Foto[21] Neighbor[1].tokens 0 Neighbor[3].tokens 0 Neighbor[8].tokens 1	Foto[21] Neighbor[0].tokens 1 Neighbor[9].tokens 1 Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[8].tokens 2	Foto[21] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 3 Neighbor[9].tokens 2 Neighbor[1].tokens 1 Neighbor[6].tokens 7 Neighbor[2].tokens 7 Neighbor[7].tokens 5	Foto[21] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 2 Neighbor[8].tokens 3 Neighbor[2].tokens 3 Neighbor[3].tokens 0 Neighbor[0].tokens 3
Foto[22] Neighbor[7].tokens 3 Neighbor[9].tokens 1	Foto[22] Neighbor[3].tokens 0 Neighbor[6].tokens 4 Neighbor[8].tokens 5	Foto[22] Neighbor[4].tokens 0 Neighbor[8].tokens 5 Neighbor[9].tokens 1		Foto[22] Neighbor[8].tokens 5 Neighbor[9].tokens 1 Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[7].tokens 3	Foto[22] Neighbor[1].tokens 0 Neighbor[3].tokens 0 Neighbor[8].tokens 5	Foto[22] Neighbor[0].tokens 4 Neighbor[9].tokens 1 Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[8].tokens 5	Foto[22] Neighbor[3].tokens 0 Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[1].tokens 0 Neighbor[6].tokens 3 Neighbor[2].tokens 3 Neighbor[7].tokens 2	Foto[22] Neighbor[4].tokens 0 Neighbor[5].tokens 1 Neighbor[7].tokens 2 Neighbor[8].tokens 4 Neighbor[2].tokens 3 Neighbor[3].tokens 0 Neighbor[0].tokens 4

Figura A.5: Figura com extrato da tabela de escalonamento - parte 5.

Figura A.6: Figura com extrato da tabela de escalonamento - parte 6.

Figura A.7: Figura com extrato da tabela de escalonamento - parte 7.

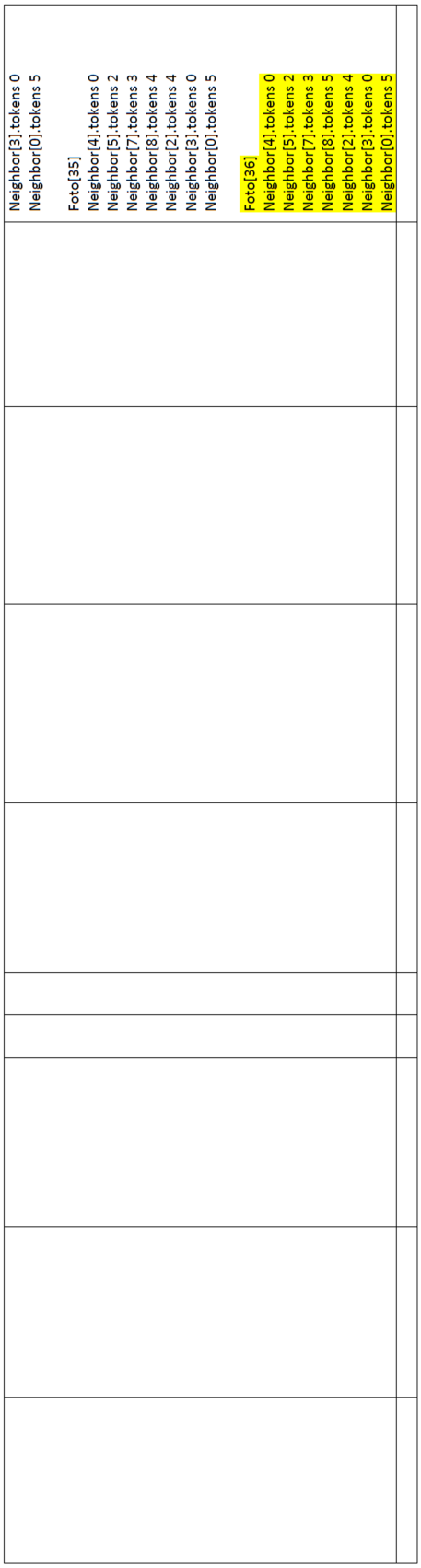


Figura A.8: Figura com extrato da tabela de escalonamento - parte 8.