Um Algoritmo de Codificação Diferenciada para Redes de Sensores Sem Fio*

Juliana F. S. Aquino[1], Eduardo F. Nakamura[2], Antonio A. F. Loureiro[1]

[1] Departamento de Ciência da Computação (DCC)
Universidade Federal de Minas Gerais (UFMG)
Belo Horizonte, MG – Brasil

[2]Fundação Centro de Análise, Pesquisa e Inovação Tecnológica (FUCAPI)
Manaus, AM – Brasil

{jfsa, loureiro}@dcc.ufmg.br, eduardo.nakamura@fucapi.br

Abstract. *A typical problem of wireless sensors networks is how to collect and send historical information from all sensor nodes to the base station. Due to the resource limitations such as energy, it may be impracticable to transmit a full-resolution data feed from each sensor node. Thus, it is important to apply techniques for the reduction of the data so that the fewest bits can be transmitted in the wireless medium. In this work, a differential coding algorithm for wireless sensor networks is proposed. The algorithm reports just the differences among their readings for a common database. Simulation results reveal that the algorithm has a good performance in applications that the sensor nodes collect similar readings.*

Resumo. *Um problema típico em redes de sensores sem fio é como coletar e enviar informações históricas de todos os nós sensores da rede para a estação base. Como a energia disponível desses nós é um recurso crítico, é impraticável transmitir todo o conjunto de dados de cada nó sensor para o nó sorvedouro. Sendo assim, é importante aplicar técnicas de redução dos dados para que menos bits possam ser transmitidos. Neste trabalho, é proposto um algoritmo de codificação diferenciada, onde os nós de sensoriamento enviam apenas as diferenças de suas leituras para uma base comum de dados. Os resultados de simulação mostraram que o algoritmo apresenta um bom desempenho em aplicações onde os nós sensores coletam leituras similares.*

1. Introdução

Uma rede de sensores sem fio (RSSF) é um tipo de rede *ad hoc* composta por uma grande quantidade de nós, dotados de capacidade de sensoriamento, de processamento de dados e de disseminação de dados, coletados e processados para um ou mais observadores. RSSFs podem ser usadas com o propósito de coletar dados em aplicações de diversas áreas, como: monitoramento ambiental, monitoramento de pacientes, gerenciamento de desastres e aplicações militares [Akyildiz et al. 2002].

As RSSFs possuem limitações de recursos como: energia, largura de banda para comunicação, capacidade de processamento e quantidade de memória. Uma vez que

*Os autores agradecem ao CNPq pelo apoio financeiro dado ao desenvolvimento deste trabalho.

o processamento dos dados consome muito menos energia do que a transmissão desses dados em um canal de comunicação sem fio, é interessante aplicar técnicas de compressão de dados lógica (também conhecida como supressão de dados) para reduzir o consumo de energia total por nó sensor.

O problema de energia disponível dos nós sensores torna-se ainda mais crítico em grandes redes de disseminação de dados contínua, cujas aplicações requerem o conteúdo histórico das leituras de cada nó sensor. Baseado no fato de que redes de nós sensores em larga escala possuem tanto redundância espacial, quanto temporal, apresenta-se, neste trabalho, uma solução que explora essa redundância de dados ainda na fonte: um algoritmo de compressão de dados lógica, sem perda de dados, baseado em codificação diferenciada para redes de disseminação contínua de dados, cujas aplicações requerem conteúdo histórico de todos os dispositivos sensores.

Este trabalho está organizado da seguinte forma. A Seção 2 apresenta alguns trabalhos que usam técnicas de codificação diferenciada em redes de sensores sem fio. A Seção 3 apresenta o algoritmo de codificação diferenciada de dados proposto, bem como o protocolo de comunicação de dados usado para a incorporação desse algoritmo. A Seção 4 apresenta uma série de experimentos realizados para avaliar o desempenho do algoritmo de codificação diferenciada. A Seção 5 trata das conclusões sobre o estudo desenvolvido, limitações do algoritmo proposto e sugestões para trabalhos futuros.

2. Trabalhos Relacionados

Há diversos trabalhos que objetivam comprimir dados em redes de sensores sem fio, como métodos de codificação distribuída [Pradhan et al. 2002], métodos de transformada [Ciancio and Ortega 2005] e métodos de dicionário [Gunopulos et al. 2005]. Porém, neste trabalho, o foco estudado é sobre alguns métodos de codificação diferenciada.

Os métodos de codificação diferenciada usam as leituras enviadas anteriormente para codificarem novas leituras baseadas na diferença entre elas ou baseadas na diferença de uma base comum definida por um nó central.

Ju e Cui propuseram um mecanismo de compressão de pacotes para RSSFs, chamado de EasiPC [Ju and Cui 2005]. A técnica de codificação diferenciada não é aplicada somente às leituras dos nós sensores, mas também ao cabeçalho dos pacotes. Campos de cabeçalho que não se modificam ou que se modificam raramente durante um fluxo de dados são enviados somente no início do fluxo. Ao se transmitir novos pacotes, os valores dos campos de cabeçalho e dos dados são enviados baseados na diferença dos valores enviados anteriormente. Testes em aplicações reais mostraram a viabilidade deste trabalho, porém, como a computação das diferenças é realizada sempre em relação às últimas informações enviadas, a perda de pacotes intermediários pode comprometer a correta codificação dos pacotes subseqüentes.

Hoang e Motani apresentaram uma abordagem que explora a característica natural de *broadcast* do canal sem fio para realizar em conjunto compressão de dados [Hoang and Motani 2005] . Em uma rede de sensores baseada em *clusters*, quando um nó particular envia seus dados para o *cluster-head*, os outros sensores podem receber esses dados e utilizá-los para comprimir os seus próprios dados através de codificação diferenciada. Nesse mesmo trabalho, foi proposto um problema de otimização em que os

nós sensores em cada *cluster* colaboram entre si transmitindo, recebendo e comprimindo dados para maximizar o tempo de vida útil da rede de sensores sem fio. Porém, este trabalho apresenta o mesmo problema do EasiPC: a possibilidade de codificação incorreta dos dados em virtude da perda de pacotes bases.

O algoritmo PINCO [Altunbasak et al. 2003] combina os dados coletados pelos nós sensores em grupos de dados, satisfazendo a um limite de latência definido pelo usuário. Um grupo de dados consiste em um prefixo compartilhado, que contém os bits mais significativos comuns às medidas de todos os nós sensores do grupo, e uma lista de sufixos, onde cada sufixo contém os restantes dos bits menos significativos. A principal desvantagem deste método é que o tamanho deste prefixo compartilhado não é atualizado de acordo com a variação das leituras coletadas na rede.

3. Protocolo de Comunicação de Dados

O algoritmo de codificação diferenciada requer um protocolo de comunicação de dados para que os nós sensores possam fornecer e distribuir o resultado da consulta para a estação base. Para isso, utilizou-se o protocolo *Data Funneling* [Petrovic et al. 2003]. O *Data Funneling* foi escolhido por permitir a divisão da área de monitoramento em regiões menores. Essa divisão pode ser baseada na proximidade de leituras dos nós sensores, o que melhora a compressão dos dados nessas regiões. Além disso, os nós sensores que executam esse protocolo não enviam as suas leituras imediatamente durante a fase de comunicação dos dados. Esses nós esperam um tempo inversamente proporcional à sua distância para os nós líderes dessas regiões, o que permite o acúmulo de leituras de outros nós sensores para realizar em conjunto a compressão dessas leituras.

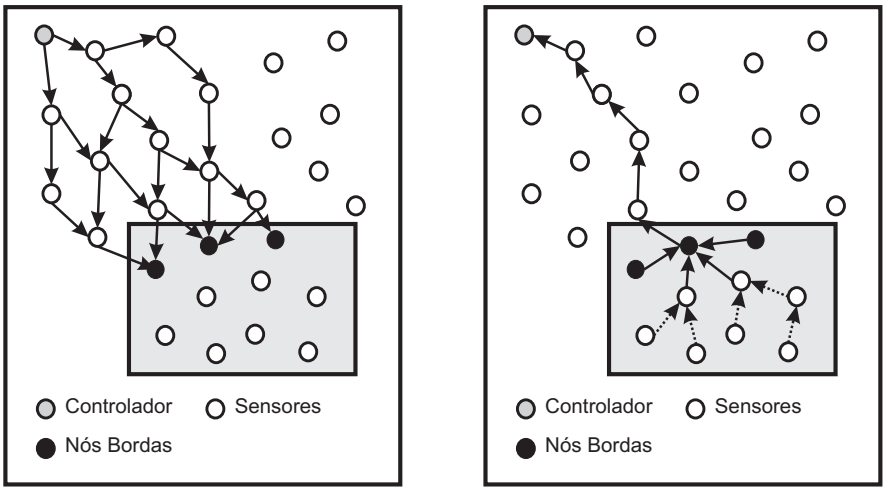
O *Data Funneling* possui dois algoritmos principais: um algoritmo de roteamento e um algoritmo de codificação por ordenação, chamado *Coding by Ordering*. Esse protocolo incorporou o algoritmo de roteamento ciente de energia dos nós sensores, chamado *Energy Aware Routing* [Shah and Rabaey 2002]. A idéia básica desse algoritmo de roteamento é aumentar o tempo de vida útil da RSSF através do uso de caminhos sub-ótimos, ocasionalmente, para assegurar que os caminhos ótimos não sejam extintos mais rapidamente. Para isso, múltiplos caminhos são encontrados entre a origem e o destino e cada caminho é associado a uma probabilidade de ser escolhido. Toda vez que um dado é enviado da origem ao destino, um dos caminhos é escolhido aleatoriamente dependendo dessas probabilidades.

O *Data Funneling* pode ser dividido em três fases: fase de configuração, fase de comunicação dos dados e fase de manutenção. Para a incorporação do algoritmo de codificação diferenciada à esse protocolo, adicionou-se a fase de formação de base comum, logo após a fase de configuração. Na fase de comunicação dos dados, substitui-se o algoritmo de codificação por ordenação pelo algoritmo de codificação diferenciada aqui proposto. À fase de manutenção, foi adicionado uma etapa para manter atualizada a base comum dos dados.

O tempo de execução do protocolo *Data Funneling* é dividido em rodadas e o tempo de uma rodada é definido pela estação base. Nas subseções seguintes, apresenta-se cada uma das fases desse protocolo de comunicação de dados modificado.

3.1. Fase de configuração

A estação base, ou nó controlador, é o elemento da rede responsável por dividir o espaço a ser monitorado em diferentes regiões quadradas e enviar pacotes de controle, chamados pacotes de interesse, para cada região a ser monitorada, como mostrado na Figura 1(a). Nesta figura, também observa-se que nem todos os nós sensores propagam o pacote de interesse em direção à região alvo. Isso caracteriza a fase de inundação direcional, também chamada de fase de propagação de interesses. Nesse pacote, há um campo de custo de comunicação para o nó controlador, cujo valor é definido como zero antes do envio desse pacote em direção à região a ser monitorada.



(a) O nó controlador envia pacotes de interesse em direção à região alvo durante a fase de configuração.

(b) Os nós sensores da região alvo enviam suas leituras para um mesmo nó borda em cada rodada na fase de comunicação dos dados.

Figura 1. Fases de configuração e de comunicação.

Cada nó que está na região de interesse (região alvo), ao receber o pacote de interesse, checa se ele está na região a ser monitorada através das coordenadas da região alvo enviadas nesse pacote. Se ele não está, ele computa seu custo para a comunicação com o nó controlador, atualiza esse custo dentro do pacote de interesse e envia-o em direção à região especificada. Todos os nós intermediários encaminham o pacote de interesse somente para os vizinhos que estão mais próximos da região alvo. Mais detalhes sobre como esse custo é calculado pode ser encontrado em [Shah and Rabaey 2002].

Na recepção de um pacote de interesse, calcula-se o custo para se enviar um pacote de volta ao vizinho que enviou o pacote de interesse. Esse custo é computado e é adicionado ao custo total da rota para se alcançar o nó controlador.

Caminhos que têm um alto custo são descartados e não são adicionados à tabela de roteamento, que mantêm múltiplos caminhos dos nós sensores até o nó controlador. Cada nó sensor N_i determina a probabilidade de se escolher uma rota de encaminhamento de dados via um nó vizinho N_j nas tabelas de encaminhamento, onde a probabilidade de se escolher uma rota é inversamente proporcional ao custo dessa rota.

Quando um nó, que está na região alvo, recebe um pacote de interesse de um nó vizinho, que está fora da região alvo, então a fase de inundação direcional é concluída.

O nó percebe que ele está na borda da região e se auto-designa para ser um nó borda, como mostra a Figura 1(a). Cada nó borda computa seu custo de comunicação com o controlador da mesma maneira como foi feita pelos nós que estão fora da região durante a fase de inundação. Um nó borda inunda a região de interesse agora com uma versão modificada do pacote de interesse. O custo para alcançar o nó controlador assume o valor zero e torna-se o custo para alcançar o nó borda. Dentro da região, cada nó mantém uma tabela com o custo de comunicação para se alcançar o nó borda e não para se alcançar o nó controlador.

Dois novos campos são adicionados ao pacote de interesse modificado. Um dos campos mantém o número de saltos entre o nó borda e o nó corrente que está processando o pacote. O outro campo especifica o custo do nó borda para comunicação com o nó controlador, e, este campo, uma vez definido pelo nó borda, não se modifica à medida que o pacote é encaminhado de um nó para outro.

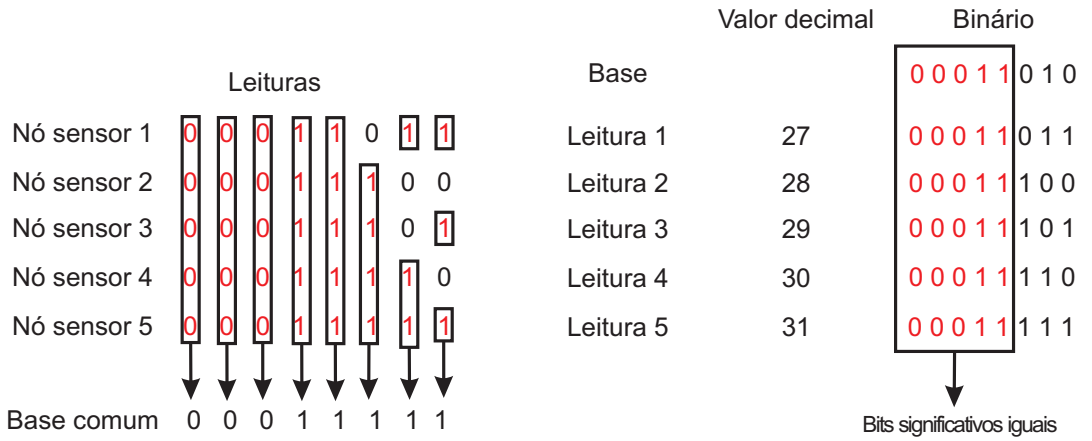
Uma vez que os nós dentro da região alvo recebem o pacote de interesse modificado dos nós bordas, eles passam a rotear suas leituras para o nó controlador via um dos nós bordas da região. Desde que haja muitos nós bordas dentro da região, a máxima agregação das leituras dos nós sensores requer que todos os nós dentro da região concordem em rotear seus dados via o mesmo nó borda durante cada rodada de reporte de leituras para o nó controlador. Isto é alcançado pelos nós sensores da região alvo, pois eles aplicam a mesma função determinística para os custos de todos os nós bordas para se alcançar o nó controlador. Como todos os nós aplicam a mesma função para as mesmas entradas, todos eles irão computar o mesmo escalonamento, elegendo o mesmo nó borda (ver Figura 1(b)). A função usada para computar o escalonamento é similar à função usada para computar as probabilidades para selecionar diferentes rotas no roteamento probabilístico em [Shah and Rabaey 2002].

3.2. Fase de formação de base comum

Esta fase foi adicionada ao protocolo *Data Funneling*. Ela se inicia quando o nó controlador envia uma mensagem em direção à região alvo para que os nós bordas possam dar início à fase de formação de base comum. Os nós bordas da região alvo encaminham esse pedido para os nós sensores da região. Quando um nó sensor recebe esse pedido, ele começa a enviar leituras, em um intervalo de tempo determinado no pacote recebido, para um único nó borda eleito para esse intervalo de tempo.

Na Figura 2(a), mostra-se como uma base comum de bits é formada a partir de várias leituras de nós sensores. Para cada leitura, verifica-se qual é o valor do bit em cada posição do dado. Por exemplo, para o bit mais significativo, o bit zero apresentou-se mais presente. Para o segundo bit mais significativo, o bit zero também é o mais freqüente. E assim, sucessivamente, faz-se a varredura em cada posição dos bits das leituras dos nós sensores para compor a base comum. A motivação para a escolha de uma base comum de bits advém do fato de que em uma rede densa, os nós sensores de uma mesma região sensoriam leituras iguais ou muito próximas. Assim, os bits mais significativos, geralmente, permanecem os mesmos para leituras subseqüentes, como pode ser visto na Figura 2(b).

Para cada rodada de reporte de dados, o nó borda computa uma base comum. Após n rodadas, o nó borda escolhe a base comum mais freqüente da lista de bases formadas até então para enviá-la aos nós sensores. Após receberem a base comum de dados, os



(a) Formação da base comum de bits. Faz-se a varredura em cada posição dos bits das leituras dos nós sensores para se compor a base comum.

(b) Um exemplo onde os bits mais significativos são iguais para valores consecutivos.

Figura 2. Formação da base comum de bits.

sensores irão reportar as suas leituras de acordo com a diferença para a base comum, como pode ser observado na Figura 3.

	Valor decimal	Binário	Diferença
Base		0 0 0 1 1 1 1 1	
Leitura 1	27	0 0 0 1 1 0 1 1	0 1 1
Leitura 2	28	0 0 0 1 1 1 0 0	0 0
Leitura 3	29	0 0 0 1 1 1 0 1	0 1
Leitura 4	30	0 0 0 1 1 1 1 0	0
Leitura 5	31	0 0 0 1 1 1 1 1	-

Figura 3. Diferença de bits das leituras para a base comum de dados.

3.3. Fase de comunicação dos dados

Quando se inicia uma nova rodada para o envio de leituras para o nó controlador, os nós sensores não enviam seus dados diretamente. Eles esperam uma quantidade de tempo inversamente proporcional à sua distância em número de saltos para o nó borda que será usado na rodada corrente. Isso permite que os nós mais distantes do nó borda enviem seus dados primeiro para os nós que estão mais próximos. Desta maneira, nós mais próximos ao nó borda irão receber primeiro as leituras de nós descendentes na árvore de roteamento para processá-las junto com suas próprias leituras. No final, todos os dados enviados pelos nós sensores da região serão coletados em um único nó borda e enviados em um único pacote para o nó controlador.

Nos pontos de agregação, além da própria agregação de pacotes, aplica-se a técnica de codificação diferenciada nas leituras dos sensores. Ao receber um pacote de dados de um sensor diferente, o nó receptor armazena essa leitura diferenciada recebida em um *buffer*. O intervalo de reporte de leituras para o nó borda é determinado no pacote

de formação de base. Um sensor envia suas leituras comprimidas para o nó borda quando esse intervalo expira e, neste momento, todas as leituras diferenciadas armazenadas no *buffer* são agregadas. Os bits diferenciados são concatenados como mostra a Figura 4(a).

Base	0 0 0 1 1 1 1 1	
Decimal	Binário	Diferença
26	0 0 0 1 1 0 1 0	0 1 0
27	0 0 0 1 1 0 1 1	0 1 1
29	0 0 0 1 1 1 0 1	0 1
Concatenação das leituras		0 1 0 0 1 1 0 1
Posição das leituras		0 0 1 0 0 1 0 1

(a) Concatenação de leituras. Os bits de diferença para a base comum de dados são concatenados e enviados em um único pacote para o nó borda.

Concatenação das leituras		0 1 0 0 1 1 0 1
Posição das leituras		0 0 1 0 0 1 0 1
Diferença	Base	Descompressão
0 1 0	0 0 0 1 1 1 1 1	0 0 0 1 1 0 1 0
0 1 1	0 0 0 1 1 1 1 1	0 0 0 1 1 0 1 1
0 1	0 0 0 1 1 1 1 1	0 0 0 1 1 1 0 1

(b) Descompressão de leituras. Cada bit um do byte de posição das leituras indica o término de uma leitura diferenciada.

Figura 4. Compressão e descompressão de leituras diferenciadas.

Se os bits das leituras diferenciadas ultrapassarem um *byte*, um novo *byte* deve ser enviado para conter os bits restantes. Além disso, para cada *byte* enviado de dados, um *byte* a mais será necessário para identificar a posição final da leitura de cada sensor. Em virtude desse *byte* adicional, nem sempre codificação diferenciada pode ser empregada. Por exemplo, sabe-se que um nó folha possui apenas uma leitura para enviar, sendo assim, não há sentido em aplicar a técnica de codificação diferenciada em apenas uma leitura, pois, para cada *byte* de leitura a ser enviada, precisa-se enviar também um *byte* que conterá as posições das leituras diferenciadas. Se redundância temporal for permitida pela aplicação, a técnica de codificação diferenciada poderá ser aplicada nos nós folhas e em nós intermediários que recebam apenas um fluxo de dados além de suas próprias leituras.

A Figura 4(b) apresenta como a descompressão de leituras diferenciadas é feita. No *byte* que indica as posições das leituras dos sensores, cada bit um, significa o término de uma leitura. Os bits iniciais de uma leitura decodificada correspondem aos bits iniciais da base comum de dados.

3.4. Fase de manutenção

Nesta fase, realiza-se inundação localizada para manter atualizada a tabela de rotas dos nós sensores que fora construída durante a fase de propagação de interesses. Este processo pode ser desencadeado por vários elementos da rede, como nós bordas com baixo nível de energia, ou pelo nó controlador, quando ele detecta falhas de alguns nós bordas. Neste último caso, o nó controlador inicia o processo de inundação localizada para redefinir as regiões e encontrar novos nós bordas. A fase de manutenção também pode ser realizada periodicamente pelos nós bordas.

Eventualmente, uma nova base comum de dados pode ser propagada para manter a base comum da região alvo sempre atualizada de acordo com as variações das leituras coletadas na região. Essa nova base pode ser propagada quando a taxa de compressão média, calculada após uma quantidade pré-determinada de rodadas nos nós bordas, cai abaixo de um limiar definido pelo usuário. Se a taxa de compressão média não atingir os

limites determinados, uma nova base para atender as variações de leitura é enviada para a região de monitoramento.

3.5. Codificação por Ordenação

O algoritmo de codificação por ordenação é o algoritmo de compressão de dados lógica do protocolo *Data Funneling*. Esse algoritmo foi substituído pelo algoritmo de codificação diferenciada aqui proposto, porém, ele será detalhado nesta seção para servir de base de comparação na Seção 4.

Quando muitas leituras são transmitidas e a ordem em que essas leituras chegam ao destino não é importante para a aplicação (isto é, o transmissor deve escolher a ordem em que ele deve enviar essas leituras), então a escolha da ordem em que essas leituras deverão ser enviadas pode ser usada para conduzir informações adicionais ao receptor. Através dessa ordem, pode-se suprimir algumas leituras. A quantidade de leituras suprimidas depende do intervalo de valores em que essas leituras podem ser geradas e da quantidade de leituras de diferentes sensores presentes no codificador (nó sensor). Esse é o método de codificação por ordenação, ou *coding by ordering* [Petrovic et al. 2003].

Por exemplo, considere o caso quando há 4 nós sensores com identificadores 1, 2, 3 e 4, na região a ser monitorada, e que esses nós sejam capazes de gerar leituras inteiras entre 0 e 5. O nó borda pode escolher suprimir o valor gerado pelo nó 4, com leitura de valor 3, e escolher uma ordenação apropriada das leituras dos outros 3 nós para indicar o valor gerado pelo nó 4. Pela Tabela 1, pode-se observar que a ordem em que as leituras serão concatenadas é: 2, 3 e 1.

Permutação	Valor inteiro
123	0
132	1
213	2
231	3
312	4
321	5

Tabela 1. Mapeamento de permutação de três identificadores para inteiros.

Se um nó sensor receber um pacote de dados, cujas leituras estão na ordem 231, significa que a leitura que foi suprimida tem valor 3 e, assim, sucessivamente.

Neste exemplo, como as leituras possuem 6 variações de valores, então precisa-se de pelos menos 3 leituras, devidamente ordenadas, para que se possa inferir o valor de uma leitura suprimida. Neste exemplo, a cada 4 leituras, uma leitura pode ser suprimida.

4. Resultados de Simulação

Para avaliar o algoritmo de codificação diferenciada, foram usadas duas versões do protocolo *Data Funneling*: (i) o uso deste protocolo em sua versão original com o uso do algoritmo de codificação por ordenação e (ii) o uso desse protocolo modificado com o algoritmo de codificação diferenciada.

Os dados gerados pelos nós sensores foram simulados usando uma distribuição normal de média μ e desvio-padrão de σ, de valores 29 e 1.0, respectivamente, com 120

valores possíveis para as leituras geradas. A topologia simulada consiste em 360 nós distribuídos em uma área quadrada de tamanho 700m x 700m. O intervalo de tempo para recriação de rotas foi de $10min$ para todos os experimentos realizados. Os testes foram realizados no *network simulator* [VINT 1995].

Os parâmetros comuns usados por todos os experimentos são apresentados na Tabela 2. As métricas usadas nos experimentos são: (i) energia média consumida pela rede e (ii) taxa de compressão média em relação a uma agregação de pacotes comum.

Parâmetro	Valor
Localização do nó sorvedouro	(0,0)
Alcance do rádio	50m
Banda	250 kbps
Tempo de simulação	3600s
Média das leituras	29
Desvio-padrão das leituras	1.0
Intervalo de coleta de leituras	60s
Intervalo de disseminação de leituras	60s
Energia inicial dos nós sensores	100J
Energia consumida na transmissão	0,3W
Energia consumida na recepção	0,4W

Tabela 2. Parâmetros comuns dos experimentos.

Para calcular o número de execuções necessárias a cada teste para se ter um nível de confiança de 95%, foram executadas, inicialmente, 15 execuções (amostras) de cada experimento abaixo. Calculou-se a média de cada amostra para se determinar o número ideal de execuções n, de acordo com $n = \left(\frac{100zs}{rm}\right)^2$, onde z é obtido da tabela de distribuição normal, s é o tamanho da amostra (15), r é a acurácia desejada (5%) e m é a média da amostra.

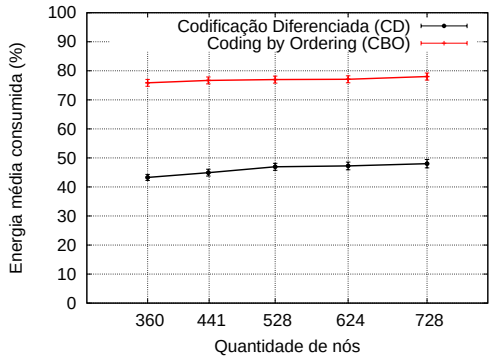
4.1. Escalabilidade

Escalabilidade é uma característica desejável em redes de sensores sem fio, pois esse teste indica a habilidade do uso de algoritmos de compressão quando se aumenta a quantidade de nós sensores e a área de monitoramento de maneira uniforme. Manteve-se constante a densidade da rede em 1 nó por $928m^2$.

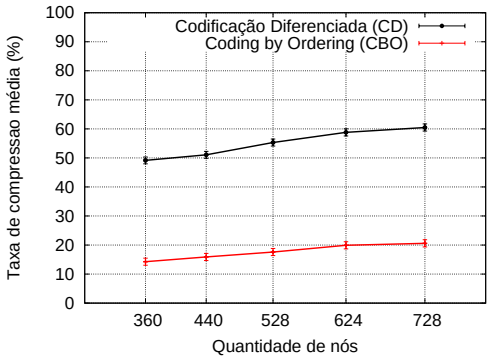
A Figura 5(a) mostra a energia média consumida pelos dois algoritmos de codificação: diferenciada e por ordenação. Embora a energia média consumida pela rede aumente à medida que se aumenta a quantidade de nós fontes, a proporção de aumento cai gradativamente. Isso ocorre em virtude do aumento da taxa de compressão média dos dados, pois a altura da árvore de roteamento tende a aumentar. Quanto mais balanceada for a árvore de roteamento, maior será o balanceamento da energia consumida por cada nó. O algoritmo de codificação diferenciada apresentou uma taxa de compressão média do nó borda entre 50% e 60% em relação a um algoritmo com simples agregação de pacotes, enquanto que o algoritmo *coding by ordering* somente conseguiu atingir uma melhora de 20% em relação à agregação de pacotes comum (ver Figura 5(b)).

A energia média consumida aumenta devido às fases de configuração e de manutenção, esta última fase possui maior custo por realizar inundação em toda a rede para manter atualizadas as rotas.

A taxa de compressão do algoritmo *coding by ordering* neste experimento atinge um baixo rendimento porque somente a cada cinco leituras (considerando-se 120 leituras possíveis) que chegam em um ponto de agregação durante uma mesma rodada, pode-se suprimir uma leitura. Sendo assim, pacotes de dados que chegam até o nó borda sem no mínimo cinco leituras neste pacote, não sofrem qualquer compressão de dados, somente a agregação. Outro fator que explica a diferença das taxas de compressão entre esses dois algoritmos é a de que a codificação diferencida faz melhor uso dos valores das leituras coletadas pelos nós sensores, enquanto que a codificação por ordenação não utiliza as informações coletadas pelos próprios nós sensores para melhorar a taxa de compressão média de dados.



(a) Energia média consumida pela rede.



(b) Taxa de compressão média do nó borda.

Figura 5. Métricas de escalabilidade.

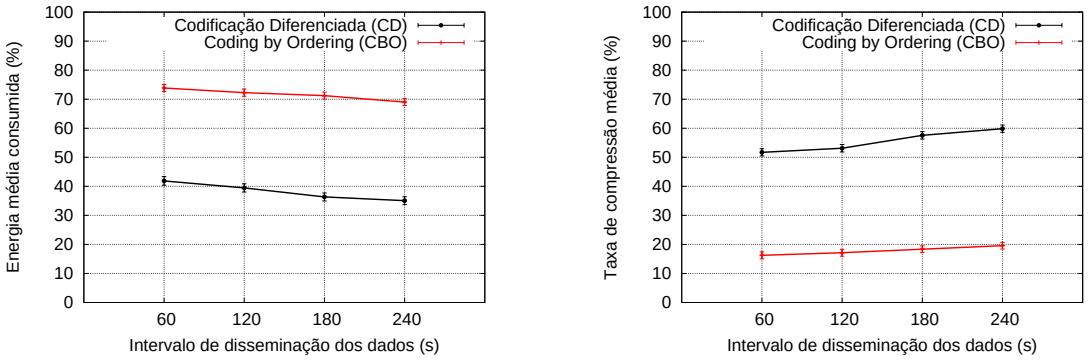
4.2. Intervalo de disseminação dos dados

Neste experimento, avaliou-se o comportamento do algoritmo de codificação diferenciada à medida que se aumenta a latência de disseminação das leituras dos nós sensores em direção ao nó sorvedouro, que variou entre 60s e 240s.

Na Figura 6(a), pode-se observar que a energia média consumida em relação à energia inicial da rede diminui à medida que se aumenta o intervalo de disseminação dos dados. Isso ocorre devido ao aumento da taxa de compressão dos dados da rede, além da menor utilização do rádio para as transmissões.

Na Figura 6(b), pode ser observada a taxa de compressão média do nó borda. A taxa de compressão aumenta principalmente pela aplicação da codificação diferenciada nos nós folhas da árvore de roteamento (a partir do intervalo 180s) e pela melhora da compressão em nós com baixo grau de vizinhos (em todos os intervalos). O algoritmo de codificação por ordenação também apresenta uma melhora em virtude da redundância temporal, mas a sua taxa de compressão média aumenta em proporções bem menores, pois o aumento da quantidade de fluxos de dados em 2, 3 ou 4 vezes, não aumenta, necessariamente, a taxa de compressão. Por exemplo, se a cada 5 leituras, pode-se suprimir 1 leitura, a cada 10 leituras, pode-se suprimir 2 leituras, ou seja, a taxa de compressão é constante. O aumento da taxa de compressão do *coding by ordering* observado ocorre em

decorrência de que a redundância temporal pode favorecer pontos de agregação que estão próximos de suprimir mais uma leitura. Considere um nó sensor que recebe 9 leituras em uma mesma rodada, dobrando-se o intervalo de disseminação dos dados, esse nó irá ter um total de 18 leituras, podendo suprimir 3 leituras, ao invés de apenas 2 leituras.



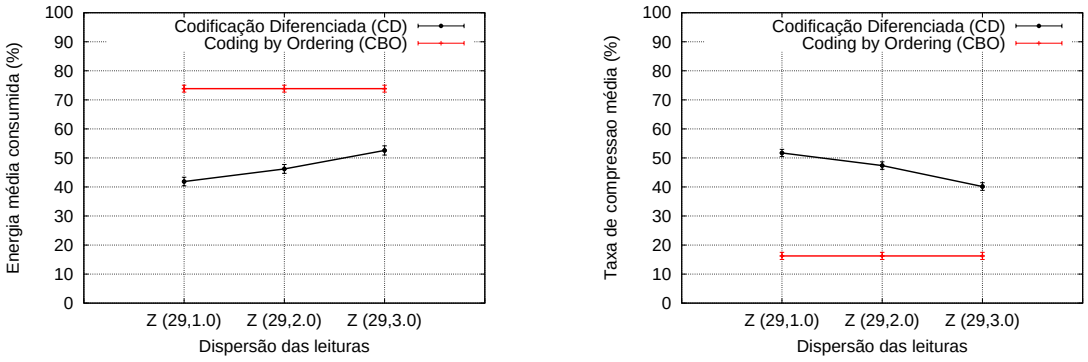
(a) Energia média consumida pela rede. (b) Taxa de compressão média do nó borda.

Figura 6. Métricas do intervalo de disseminação dos dados.

4.3. Dispersão média das leituras

Neste teste avalia-se o desempenho do algoritmo à medida que se aumenta a dispersão dos valores das leituras dos nós sensores de uma dada região. Esse teste é importante para se avaliar o comportamento do algoritmo em regiões com uma maior dispersão dos valores das leituras coletadas. As leituras possuem média 29 e o desvio-padrão dessas leituras varia de 1,0 a 3,0.

A Figura 7(a) mostra a energia média consumida à medida que se aumenta a dispersão das leituras. Observa-se que o consumo médio de energia aumenta, pois algumas leituras passam a ser mais diferentes da base, ocorrendo o envio de mais *bits* de dados para o nó borda. Na Figura 7(b), observa-se que a taxa de compressão média diminui à medida que se aumenta a variação das leituras dos nós sensores de uma mesma região, pois as leituras coletadas são mais diferentes da base, possuindo maior tamanho.



(a) Energia média consumida pela rede. (b) Taxa de compressão média do nó borda.

Figura 7. Métricas de dispersão média das leituras.

Através deste teste, pode-se concluir que o desempenho do algoritmo de codificação diferenciada depende também de como os nós sensores estão organizados

em suas regiões. A divisão em regiões baseadas nas leituras dos nós sensores e do agrupamento de sensores com leituras próximas em uma mesma região pode contribuir para aumentar o desempenho do algoritmo. É importante salientar que a eficiência de um algoritmo de compressão de dados, presente na camada de aplicação, depende do bom gerenciamento em todas as camadas da pilha de protocolos da RSSF. Neste teste, observa-se que o agrupamento de nós sensores baseados na proximidade dos valores de leituras contribui para aumentar o tempo de vida útil da RSSF.

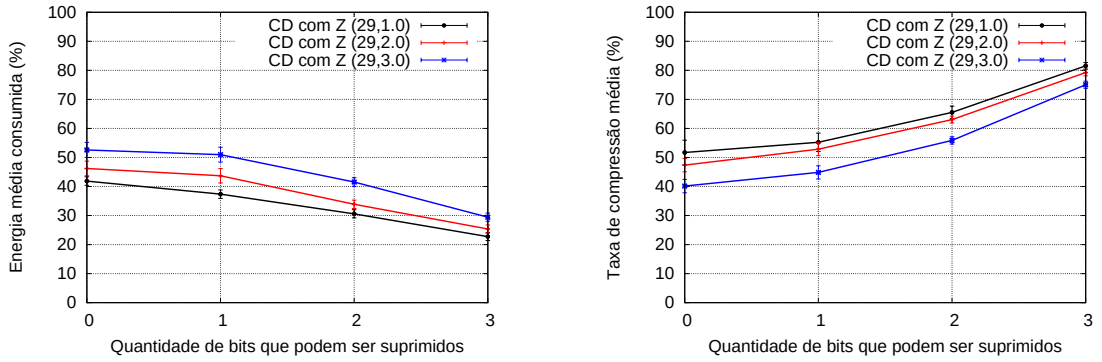
No método *coding by ordering*, não há melhora nos resultados, tendo em vista que esse método de codificação não faz uso das leituras geradas para obter uma melhor compressão.

4.4. Imprecisão das leituras

Pretende-se avaliar o desempenho do algoritmo de codificação diferenciada quando a imprecisão de alguns *bits* nas leituras dos nós sensores for permitida. A perda de alguns *bits* na leitura de um nó sensor pode não ser importante para aplicações que não requerem uma precisão exata do dado sensoriado. As aplicações desse teste permitiram erros variando de 0 a 3 *bits*, ocasionando um erro na precisão das leituras de 0 a 7 unidades.

Observa-se na Figura 8(a) que, à medida que se permite mais perdas de *bits*, a energia média consumida pela rede dimuinui em todos os cenários com diferentes dispersões médias das leituras. Isso ocorre devido à transmissão de uma menor quantidade de *bits* dos nós sensores para os nós bordas. À medida que se aumenta a dispersão média das leituras dos nós sensores, há uma maior variação na economia do consumo médio de energia. Isso acontece porque, para um cenário onde a dispersão média das leituras é maior, o ganho na economia de energia é mais significativo do que para cenários onde as leituras pouco variam, pois esses últimos já teriam menos *bits* para enviar, mesmo se 100% de precisão fosse exigida.

A Figura 8(b) mostra a taxa de compressão média do nó borda à medida que se aumenta a imprecisão de *bits* nas leituras dos nós sensores. A taxa de compressão também aumenta com uma menor quantidade de *bits* a serem transmitidos ao meio, onde se verifica também que a variação da taxa de compressão é maior em cenários cujas leituras possuem dispersão maior.



(a) Energia média consumida pela rede.

(b) Taxa de compressão média do nó borda.

Figura 8. Métricas de imprecisão das leituras.

4.5. Mudança de base

Neste experimento, realizou-se a mudança de *bits* da base. No tempo de simulação de 1000s, alterou-se a média das leituras dos nós bordas de modo que um dos *bits* menos significativos fosse alterado. Este teste tem como objetivo verificar a necessidade de mudança de base quando um dos seus *bits* for modificado.

Ao final de cada rodada, o nó borda calcula a taxa de compressão para as leituras recebidas durante a rodada corrente e uma base para essas leituras. A nova base é comparada à antiga. Se houver mudanças, a nova base é enviada propositalmente para se analisar a necessidade do reenvio de uma nova base.

Pela Tabela 3, observa-se que, para o cenário onde a média das leituras é 29 e o desvio-padrão das leituras é 1, reenviar a base quando há mudança do primeiro ou do segundo *bit* menos significativo, aumenta a energia consumida em relação a um cenário quando não há reenvio de base. Neste cenário, apenas quando há mudança acima do terceiro *bit*, há uma economia de energia reenviando uma nova base.

Através desse teste, concluiu-se que o envio de uma nova base sempre que houver mudança nem sempre é ideal. A determinação do envio de uma nova base irá depender da média das leituras e de quão diferente é a nova base.

	Sem enviar uma nova base		**Enviando uma nova base**	
Bit modificado	Energia média consumida	IC	Energia média consumida	IC
1 bit	41.87	0.47	44.56	0.49
2 bit	41.99	0.55	45.72	0.61
3 bit	**48.87**	**0.83**	**45.32**	**0.75**
4 bit	**53.24**	**1.26**	**47.10**	**0.95**

Tabela 3. Energia média consumida sem atualização e com atualização da base.

5. Conclusões

O algoritmo de codificação diferenciada reduz a redundância dos dados coletados pelos nós sensores, diminuindo a quantidade de bits enviados pelo meio sem fio. Cada nó sensor da região alvo possui uma base comum de bits, onde apenas os bits diferentes para essa base são enviados. A técnica de agregação de pacotes permite unir várias leituras diferenciadas em um único pacote a ser enviado para o nó controlador, diminuindo a probabilidade de colisões no meio sem fio. Os resultados de simulação mostraram que, para um melhor desempenho do algoritmo de codificação diferenciada, é importante que o algoritmo de roteamento utilize informações das leituras reportadas pelos nós sensores para dividir a área a ser monitorada em regiões que agrupam nós sensores com leituras similares. O algoritmo também permite: compressão lógica ainda na fonte, diferentes taxas de compressão para os nós sensores de uma mesma região, reconstrução das leituras codificadas em qualquer ponto de agregação e combinação com outras técnicas para redução de dados.

O algoritmo de codificação diferenciada, na versão deste trabalho, apresenta algumas limitações de implementação, como: no nível que este algoritmo foi implementado no simulador de redes ns-2 (nível de rede e de aplicação), mesmo quando a leitura diferenciada é menor que um *byte*, é necessário completar todo o *byte* para que a leitura possa

ser enviada. À medida que mais de um fluxo de dados ocorre nos pontos de agregação, a codificação diferenciada é aplicada de forma mais eficiente, pois em um mesmo *byte*, pode-se contemplar mais de uma leitura diferenciada.

Outra versão do algoritmo de codificação diferenciada está em andamento para realizar a codificação dos dados baseados na diferença aritmética para a base comum de dados, ao contrário do algoritmo aqui apresentado, que apresenta uma codificação diferenciada lógica. Resultados preliminares indicam que essa nova versão do algoritmo é ideal para aqueles nós sensores que estão reportando leituras distantes logicamente da base, mas próximos aritmeticamente. Como trabalho futuro, pretende-se incorporar um protocolo que use tanto o algoritmo de codificação diferenciada lógica, quanto o algoritmo de codificação diferenciada aritmética. O nó sensor irá escolher um dos dois algoritmos para codificar seus dados, avaliando qual deles codifica os dados com um menor número de bits. Pretende-se também incorporar um modelo que calcula o melhor momento para se reenviar uma nova base para garantir melhores economias de energia.

Referências

Akyildiz, I. F., Su, W., Sankarasubramanian, Y., and Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102 – 114.

Altunbasak, Y., Arici, T., Gedik, B., and Liu, L. (2003). Pinco: a pipelined in-network compression scheme for data collection in wireless sensor networks. *The 12th International Conference on Computer Communications and Networks*, pages 539 – 544.

Ciancio, A. and Ortega, A. (2005). A distributed wavelet compression algorithm for wireless multihop sensor networks using lifting. *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 4:v/825 – iv/828.

Gunopulos, D., Kalogeraki, V., Lin, S., and Lonardi, S. (2005). A data compression technique for sensor networks with dynamic bandwith allocation. *12th International Symposium on Temporal Representation and Reasoning*, pages 186 – 188.

Hoang, A. T. and Motani, M. (2005). Collaborative broadcasting and compression in cluster-based wireless sensor networks. *Proceeedings of the Second European Workshop on Wireless Sensor Networks*, pages 197 – 206.

Ju, H. and Cui, L. (2005). Easipc: a packet compression mechanism for embedded wsn. *11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, pages 294 – 399.

Petrovic, D., Rabaey, J., Ramchandran, K., and Shah, R. (2003). Data funneling: routing with aggregation and compression for wireless sensor networks. *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, pages 156 – 162.

Pradhan, S. S., Kusuma, J., and Ramchandran, K. (2002). Distributed compression in a dense microsensor network. *IEEE Signal Processing Magazine*, 19:51 – 60.

Shah, R. C. and Rabaey, J. M. (2002). Energy aware routing for low energy ad hoc sensor networks. *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, pages 156 – 162.

VINT (1995). *The Network Simulator NS-2: Documentation*.