UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA E INFORMÁTICA INDUSTRIAL

MARCOS COSTA MACIEL

COMPRESSÃO DE DADOS AMBIENTAIS EM REDES DE SENSORES SEM FIO USANDO CÓDIGO DE HUFFMAN

DISSERTAÇÃO

CURITIBA

2013

MARCOS COSTA MACIEL

COMPRESSÃO DE DADOS AMBIENTAIS EM REDES DE SENSORES SEM FIO USANDO CÓDIGO DE HUFFMAN

Dissertação apresentada ao Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do grau de “Mestre em Ciências” – Área de Concentração: Telecomunicações e Redes.

Orientador: Prof. Dr. Richard Demo Souza

Co-orientador: Prof. Dr. Henry Ponti Medeiros

CURITIBA

2013

Dados Internacionais de Catalogação na Publicação

M152 Maciel, Marcos Costa
Compressão de dados ambientais em redes sensores sem fio usando código de Huffman / Marcos Costa Maciel. – 2013.
48 f. : il. ; 30 cm

Orientador: Richard Demo Souza.
Coorientador: Henry Ponti Medeiros.
Dissertação (Mestrado) – Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial. Curitiba, 2013.
Bibliografia: f. 47-48

1. Redes de sensores sem fio. 2. Compressão de dados (Telecomunicação). 3. Teoria da codificação. 4. Algoritmos computacionais. 5. Processamento de sinais – Técnicas digitais. 6. Simulação (Computadores). 7. Engenharia elétrica – Dissertações. I. Souza, Richard Demo, orient. II. Medeiros, Henry Ponti, coorient. III. Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial. IV. Título.

CDD (22. ed.) 621.3

Biblioteca Central da UTFPR, Câmpus Curitiba

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
Câmpus Curitiba

Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial

Título da Dissertação Nº. 628

"Compressão de dados ambientais em redes de sensores sem fio usando código de Huffman"

por

Marcos Costa Maciel

Esta dissertação foi apresentada como requisito parcial à obtenção do grau de MESTRE EM CIÊNCIAS – Área de Concentração: Telecomunicações e Redes, pelo Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial – CPGEI – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Curitiba, às 10h do dia 21 de fevereiro de 2013. O trabalho foi aprovado pela Banca Examinadora, composta pelos doutores:

Prof. Richard Demo Souza, Dr.
(Presidente – UTFPR)

Prof. Marcelo Eduardo Pellenz, Dr.
(PUC-PR)

Prof. Hermes Irineu Del Monego, Dr.
(UTFPR)

Visto da coordenação:

Prof. Ricardo Lüders, Dr.
(Coordenador do CPGEI)

Dedico este trabalho à minha querida família: Paulo Maciel, Márcio, Mylene, Nonata, Marcos Paulo, Ana Paula, e Melina Maciel que esteve e está sempre ao meu lado.

AGRADECIMENTOS

Primeiramente agradeço à Deus, e à minha família pelo apoio irrestrito.

Agradeço à Universidade Tecnológica Federal do Paraná (UTFPR), ao Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial (CPGEI), ao Instituto Federal de Educação, Ciência e Tecnologia do Amazonas (IFAM) pela oportunidade de participar deste programa de Mestrado e a Fundação de Amparo à Pesquisa do Estado do Amazonas (FAPEAM) pelo incentivo e apoio para que eu pudesse me qualificar fora do meu Estado.

Agradeço a meu orientador, Professor Richard Demo Souza, pela grande ajuda na realização deste trabalho e pela dedicação que tem na orientação de seus alunos e ao Professor Henry Ponti Medeiros pelo grande auxílio no decorrer da minha pesquisa.

Finalmente, mas não menos importante, aos meus colegas do Laboratório, Glauber Brante, Marcos Kakitani e Hirley Alves pela convivência durante estes dois anos.

Não há nada que não se consiga com força de vontade, bondade e principalmente amor.
Cícero.

RESUMO

MACIEL, Marcos Costa. COMPRESSÃO DE DADOS AMBIENTAIS EM REDES DE SENSORES SEM FIO USANDO CÓDIGO DE HUFFMAN. 48 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2013.

Nesta dissertação de mestrado é apresentada uma proposta de um método simples de compressão de dados sem perda para Redes de Sensores sem Fio(RSSF). Este método é baseado numa codificação Huffman convencional aplicada a um conjunto de amostras de parâmetros monitorados que possuam uma forte correlação temporal, fazendo com que seja gerado um dicionário Huffman a partir dessas probabilidades e que possam ser utilizadas em outros conjuntos de parâmetros de mesma característica. Os resultados de simulação usando temperatura e umidade relativa mostram que este método supera alguns dos mais populares mecanismos de compressão projetados especificamente para RSSF.

Palavras-chave: Redes de Sensores sem Fio, Compressão de Dados, Codificação Huffman

ABSTRACT

MACIEL, Marcos Costa. ENVIRONMENTAL DATA COMPRESSION IN WIRELESS SENSOR NETWORKS USING HUFFMAN CODE. 48 f. Dissertação – Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial, Universidade Tecnológica Federal do Paraná. Curitiba, 2013.

In this masters thesis we present a lightweight lossless data compression method for wireless sensor networks(WSN). This method is based on a conventional Huffman coding applied to a sample set of monitored parameters that have a strong temporal correlation, so that a Huffman dictionary is generated from these probabilities, and which may be used in other sets of parameters with same characteristic. Simulations results using temperature and relative humidity measurements show that the proposed method outperforms popular compression mechanisms designed specifically for wireless sensor networks.

Keywords: Wireless Sensor Networks, Data Compression, Huffman Coding

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

MEMS	*Micro-Electro-Mechanical Systems*
RSSF	Redes de Sensores sem Fio
JPEG	*Joint Photographic Experts Group*
CMOS	Complementary metal-oxide-semiconductor
LEC	*Lossless Entropy Compression*
ADC	*Analog-to-Digital Converter*
ALFC	*Adaptative Linear Filtering Compression*

LISTA DE SÍMBOLOS

m_i	dado adquirido pelo nó sensor no instante i
χ	alfabeto fonte
L	número médio de bits usados para representar um símbolo da fonte
$H(\chi)$	entropia da fonte
R	resolução do ADC
d_i	diferença entre duas medições consecutivas
$\widetilde{d_i}$	valor não negativo de d_i
$\widehat{\mu_i}$	média dos valores do sinal de entrada
g_i	versão despolarizada das amostras
Q	ordem para os sinais despolarizados
$\widehat{g_i}$	combinação linear dos valores despolarizados
$\mathbf{w}_i$	vetor de coeficientes de pesos
$\widehat{m_i}$	predição das amostras
α	parâmetro de controle
β	parâmetro de controle
e_i	resíduo da predição
S	valor inteiro
$\widehat{G_i}$	variável inteira proveniente de $\widehat{g_i}$
$\widehat{M_i}$	variável inteira proveniente de $\widehat{m_i}$
$\widehat{\Omega_i}$	variável inteira proveniente de $\widehat{\mu_i}$
E_i	variável inteira proveniente de e_i
$\mathbf{W}_i$	vetor de inteiros proveniente de $\mathbf{w}_i$
A	valor inteiro
B	valor inteiro
$\widetilde{e_i}$	valor inteiro de predição residual
f_i	valor inteiro estritamente positivo
k	parâmetro de otimização da palavra código
γ	simbolo com menor probabilidade de ocorrência no alfabeto
λ	simbolo com menor probabilidade de ocorrência no alfabeto
r	*string* de 1s e 0s
p_j	probabilidade de ocorrência de símbolo
φ	*string* binário
δ	marcador especial
H_d	entropia das diferenças
L_u	tamanho médio do símbolo sem compressão
C_r	taxa de compressão
η	eficiência do código

SUMÁRIO

1 INTRODUÇÃO

O avanço da tecnologia dos circuitos integrados, em particular o alto grau de miniaturização, o elevado poder de processamento e a proliferação da tecnologia MEMS (do inglês *Micro-Electro-Mechanical Systems*) que vemos hoje, possibilitou a materialização de dispositivos inteligentes agregados a sensores que permitem o monitoramento de fenômenos físicos nas mais diversas áreas, tais como agricultura, meio ambiente, mapeamento urbano, etc. O agrupamento destes dispositivos com o objetivo de cooperar entre si para superar limitações proporcionou o surgimento de uma rede denominada RSSF (rede de sensores sem fio), definida como um conjunto de dispositivos eletrônicos com comunicação sem fio (nós) que dispersados numa área podem monitorar certo fenômeno de interesse. Os nós realizam medições, fazem o processamento destas e transmitem-nas a um nó líder que faz a agregação destes dados em pacotes enviando-os para uma estação base, responsável por analisá-las e proporcionar conclusões sobre determinada atividade na área de interesse (AKYILDIZ W. SU; CAYIRCI, 2002; CULLER D. ;ESTRIN, 2004; MHATRE V.; ROSENBERG, 2004; YICK J. ; MUKHERJEE, 2008).

Um dos maiores desafios para a massificação das RSSFs é o desenvolvimento de mecanismos que possibilitem as redes operarem em períodos prolongados de tempo (AKYILDIZ W. SU; CAYIRCI, 2002; CULLER D. ;ESTRIN, 2004; YICK J. ; MUKHERJEE, 2008), uma vez que a fonte de energia dos nós é em geral limitada. Sabendo-se que a comunicação de dados é um dos principais fatores responsáveis pelo consumo das reservas de energia da rede, o desenvolvimento de técnicas de compressão para uso das RSSFs com o objetivo de reduzir a quantidade de informação transmitida pelo nó sensor tem gerado grande interesse (MARCELLONI F. ; VECCHIO, 2008, 2009; SCHOELLHAMMER T. ; GREENSTEIN, 2004; SADLER; MARTONOSI, 2006; REINHARDT M. HOLLICK, 2009; REINHARDT A. ; CHRISTIN, 2010; CAPO-CHICHI H. GUYENNET, 2009; KIELY et al., 2010).

1.1 COMPRESSÃO DE DADOS

Existem muitos métodos conhecidos de compressão de dados. Eles são baseados em diferentes ideias, são adequados para diferentes tipos de dados, e produzem diferentes resultados, mas todos são baseados no mesmo princípio, isto é, comprime-se removendo redundâncias dos dados originais (SALOMON, 2004). Uma abordagem eficiente para reduzir a comunicação de dados na rede é fazer o uso da característica de correlação temporal dos dados coletados pelos nós sensores para comprimir a informação localmente antes de ser transmitida.

Embora a área de pesquisa sobre compressão de dados esteja bem estabelecida, existem algoritmos que dificilmente poderiam ser utilizados de forma embarcada no nó sensor devido aos recursos limitados de hardware. Entretanto, vários métodos de compressão de dados especificamente desenvolvidos para RSSFs têm sido proposto nos últimos anos (MARCELLONI F. ; VECCHIO, 2008, 2009; SCHOELLHAMMER T. ; GREENSTEIN, 2004; SADLER; MARTONOSI, 2006; REINHARDT M. HOLLICK, 2009; REINHARDT A. ; CHRISTIN, 2010; CAPO-CHICHI H. GUYENNET, 2009; KIELY et al., 2010). O que muitos desses métodos tem em comum é o fato de que eles fazem uso da correlação dos dados coletados pelos nós sensores para atingir alta taxa de compressão enquanto empregam algoritmos computacionalmente simples.

O algoritmo proposto por Marcelloni e Vecchio (MARCELLONI F. ; VECCHIO, 2008, 2009) é um dos métodos mais eficientes desenvolvidos de acordo com esses princípios. Ele processa as diferenças das medições consecutivas realizadas pelos sensores e divide-as em grupos de tamanho incrementados exponencialmente. Esses grupos são então codificados por entropia usando uma tabela de compressão fixa baseada no algoritmo JPEG (do inglês *Joint Photographic Experts Group*) para compressão de coeficientes DC de imagens. Os autores registraram altas taxas de compressão sobre dados reais coletados por redes de sensores sem fio.

Uma outra abordagem eficiente é proposta por Kiely *et al.* (KIELY et al., 2010) usando um algoritmo de compressão linear adaptativo. Este implementa uma compressão preditiva usando um filtro linear adaptativo para prever valores de amostras obtendo um código de entropia dos resíduos desta predição. A predição adaptativa elimina a necessidade de determinar coeficientes de predição *a priori*, e o mais importante é que permite um ajuste dinâmico na compressão no caso de uma mudança nas características da fonte.

1.2 OBJETIVOS E RESULTADOS

Este trabalho propõe um método simples de compressão de dados para RSSF, considerando suas restrições quanto a energia, tempo de processamento, tamanho de código, etc. Especificamente, será mostrado a construção de um dicionário Huffman fixo a partir das estatísticas provenientes das diferenças entre amostras consecutivas de um conjunto de dados que possua uma forte correlação temporal, tais como, temperatura ambiente e umidade. Tomando o dicionário como referência na compressão de novos conjuntos de dados, as taxas de compressão são muito próximas dos valores teóricos de entropia de cada conjunto. O método proposto é comparado com os algoritmos apresentados em (MARCELLONI F. ; VECCHIO, 2008) e (KIELY et al., 2010) especificamente desenvolvidos para trabalhar com RSSF, e os resultados numéricos mostram a superioridade do esquema proposto.

1.3 ORGANIZAÇÃO DO DOCUMENTO

O restante desta dissertação está organizado da seguinte maneira: no Capítulo 2, são apresentados alguns conceitos básicos de RSSFs, no Capítulo 3 são descritas as técnicas de compressão de dados para RSSFs propostas em (MARCELLONI F. ; VECCHIO, 2008) e (KIELY et al., 2010), bem como alguns conceitos sobre codificação Huffman. No Capítulo 4 é descrito o método proposto de compressão. No Capítulo 5 são mostrados os resultados obtidos com a abordagem proposta comparada com os esquemas descritos no Capítulo 3. No Capítulo 6 apresentam-se os comentários finais.

2 REDES DE SENSORES SEM FIO

2.1 CONCEITOS E CARACTERÍSTICAS

As RSSFs são um conjunto de nós dispersados numa área com o propósito de monitorar certo fenômeno de interesse. Os nós realizam medições, fazem o processamento destas e transmitem-nas a um nó líder que faz a agregação destes dados em pacotes enviando-os para uma estação base, responsável em analisá-las e proporcionar conclusões sobre determinada atividade na área de interesse (MHATRE V.; ROSENBERG, 2004). Estes dispositivos são caracterizados pela limitação de processamento, tempo de vida, capacidade de armazenamento de dados e largura de banda, o que pode melhorar substancialmente quando estes cooperam entre si formando uma RSSF. Esses sensores sem fio possuem também um circuito de rádio frequência que possibilita a troca de informações entre eles ficando geralmente em modo desligado quando não estão em uso, para minimizar o consumo de energia, o qual é em geral fornecida por uma bateria.

A possibilidade de ter em cada nó da rede um conjunto de sensores, que sejam capazes de monitorar diversos tipos de fenômenos, proporcionou a esta tecnologia um grande escopo de utilização. Tais como: monitoramento e rastreamento na área militar (detecção segura), em habitat natural (monitoramento de animais), planta industrial (monitoramento de máquinas e ativos), ambiente hospitalar (monitoramento de dados fisiológicos de pacientes), negócios (controle de ativos), etc (YICK J. ; MUKHERJEE, 2008).

Muitas RSSFs têm sido implantadas para monitoramento ambiental no habitat de aves marinhas ou no estudo do microclima de uma determinada área de floresta. Com o objetivo de coletar dados de temperatura, umidade, pressão atmosférica, condições de luminosidade, níveis de ruído, etc (CULLER D. ;ESTRIN, 2004) . No caso de um microclima, habitualmente os pesquisadores instalam nas copas de árvores equipamentos conectados ao chão através de cabos seriais. Com a utilização de uma RSSF isso pode ser modificado. Um nó usado para monitoramento ambiental do tamanho de um tubo de filme de máquina fotográfica pode ser uma estação meteorológica, é um exemplo do dia a dia, tendo na sua parte superior sensores

de umidade relativa, temperatura, pressão barométrica. Além disso contém dois sensores de incidência de luz um medindo a radiação solar, especificamente luz e o outro radiação fotossinteticamente ativa (banda que a clorofila é sensível), e um outro sensor localizado na parte de baixo para medição de luz radiante na sombra(CULLER D. ;ESTRIN, 2004).

2.1.1 A TECNOLOGIA EMBARCADA NAS RSSFS

Cada nó sensor é um dispositivo minúsculo formado basicamente por três componentes: um subsistema sensor para aquisição de dados do ambiente (sensor e conversores analógico-digitais); um subsistema de processamento para processamento e armazenamento local dos dados (microprocessador e unidade de armazenamento) e um subsistema de comunicação sem fio (micro rádio) para transmissão de dados até um ponto central. Ainda faz-se necessário uma fonte de alimentação para suprir energia do processamento das tarefas (ANASTASI G.;CONTI, 2009).

Os circuitos semicondutores têm se tornado cada vez menores, com menor consumo de energia para uma determinada frequência de trabalho e cobrindo menores áreas num *wafer*. Os micro sensores são materiais que sofrem variação em alguma de sua característica devido a exposição ao ambiente, tais como: termistores, fotocélulas e sistemas MEMS que tem como exemplo de uso comercial o acelerômetro utilizado nos *airbags* dos automóveis. Um micro rádio é necessário para transmissão de informação nos sistemas sem fio, atualmente estes podem ser desenvolvidos usando a tecnologia CMOS (do inglês *Complementary metal-oxide-semiconductor*). A energia necessária para comunicação aumenta rapidamente com a distância entre os nós, por isso a informação é enviada até o destino através de saltos entre os nós da rede. Um sistema operacional que trabalha bem em pequenos dispositivos que possuem recursos limitados é o *tinyOS*, que é baseado em Unix. Algumas plataformas usadas para desenvolvimento de RSSF são a *Berkeley Motes* e a *Intel iMote* (CULLER D. ;ESTRIN, 2004).

Nas RSSFs os algoritmos de roteamento são responsáveis por identificar os caminhos para que os dados sejam enviados desde os pontos de coleta de informação até o ponto de uso. Um dos recursos usados para a comunicação entre os nós é a disseminação de informação, onde protocolos de “inundamento” realizam *broadcast* na rede para enviar comandos, configurações e alarmes, ainda podendo usar este recurso para montagem de uma árvore partindo do nó raiz que pode conter um *gateway* para um outra rede (CULLER D. ;ESTRIN, 2004).

2.1.2 TEMPO DE VIDA DE UMA RSSF

A fonte de alimentação de uma RSSF frequentemente consiste em uma bateria com tempo de vida limitado, pois poderá ser impossível ou inconveniente fazer sua recarga, uma vez que os nós podem ser implantados em ambientes hostis de difícil acesso. Por outro lado a RSSF deverá ter um tempo de vida suficiente para realizar sua aplicação. Em muitos casos o tempo de vida é da ordem de alguns meses, ou até anos. Temos então a questão crucial: "Como prolongar o tempo de vida de uma RSSF? "(ANASTASI G.;CONTI, 2009).

Em alguns casos é possível suprir energia proveniente do ambiente externo (células solares). Entretanto, fontes de energia externa frequentemente tem um comportamento aleatório, logo há necessidade de um *buffer* para armazenamento (baterias). A energia é um recurso crítico, tem que ser usada com moderação. Percebe-se que a conservação de energia é uma peça chave para os projetos de sistemas baseados em RSSF (ANASTASI G.;CONTI, 2009).

Medições experimentais tem mostrado que a transmissão de dados tem um alto custo em termos de consumo de energia, enquanto que o processamento dos dados consome significativamente menos (RAGHUNATHAN V. ; SCHURGHERS, 2002). O custo de energia para transmissão de um bit de informação é aproximadamente o mesmo que é necessário para o processamento de centenas de operações num nó sensor típico (POTTIE G.; KAISER, 2000). O consumo de energia no subsistema sensor depende especificamente do tipo do sensor. Em muitos casos ele é desprezível considerando o consumo do processamento e do subsistema de comunicação. Em outros casos, a energia consumida no sensor pode ser comparada ou mesmo maior que a energia necessária para transmissão de dados (ANASTASI G.;CONTI, 2009) . Em geral as técnicas de redução de energia focam em dois subsistemas: O subsistema de rede (gerenciamento de energia levando em consideração as operações de cada nó sensor, bem como nos projetos dos protocolos de rede), e no subsistema sensor (técnicas são usadas para reduzir a quantidade do consumo de energia por amostra).

O tempo de vida de uma RSSF pode ser prolongado pela aplicação conjunta de diferentes técnicas. Protocolos com eficiência energética visam minimizar o consumo de energia durante as atividades da rede (ANASTASI G.;CONTI, 2009). De uma maneira geral identificam-se três técnicas principais para conservação de energia nas RSSFs: *duty-cycling*, *data-driven* e baseado em mobilidade (ANASTASI G.;CONTI, 2009) conforme observa-se na Figura 1.

O esquema de *duty cycling* pode ser implementado através de duas abordagens

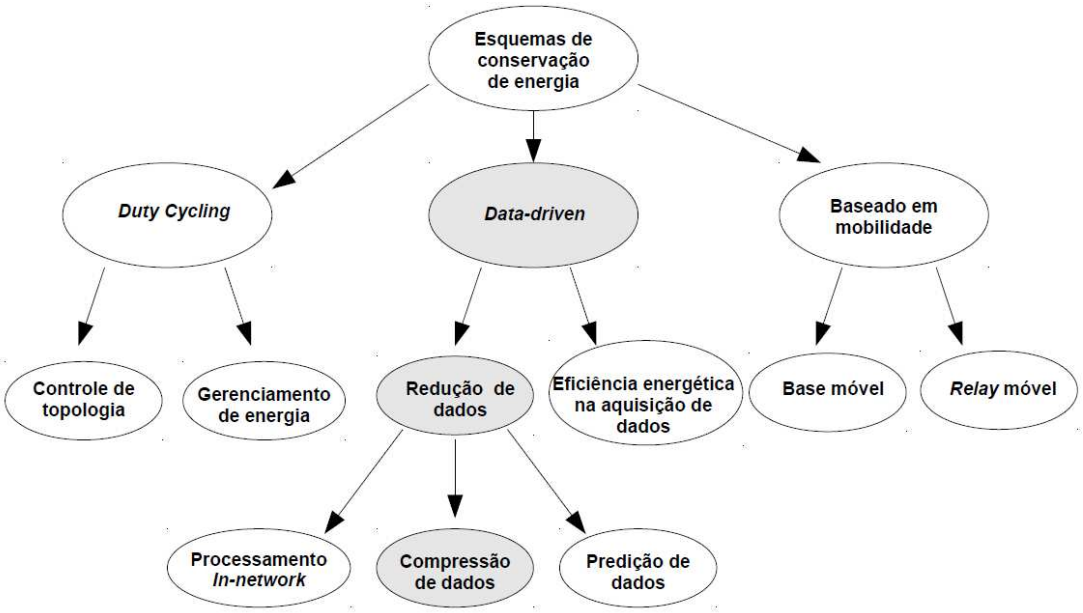

Figura 1: Taxionomia das abordagens para conservação de energia em RSSFs (ANASTASI G.; CONTI,2009)

complementares. Por um lado é possível explorar a redundância dos nós, que é típico nas RSSFs, e adaptativamente selecionar somente um subconjunto mínimo de nós que permaneçam ativos e assegurem a manutenção da conectividade. Os nós que não são necessários para assegurar a conectividade podem ir para o modo *sleep* e assim economizar energia na rede (GANESAN D.; CERPA, 2004; POTTIE G.; KAISER, 2000). Além disso, os nós ativos (selecionados pelo protocolo de controle de topologia) não necessitam manter seus rádios continuamente ligados. Eles podem ser desligados (colocados em modo *sleep*) quando não existe atividade na rede, assim alternando os modos *sleep* e *wakeup*. Esta operação de *duty cycling* sobre os nós ativos denomina-se de gerenciamento de energia. A abordagem de controle de topologia e gerenciamento de energia são técnicas complementares que implementam *duty cycling* com diferentes granularidades.

A abordagem denominada *data-driven* pode ser dividida em dois problemas a solucionar: o primeiro é a redução dos dados em caso de amostras redundantes; e o outro seria a eficiência energética no esquema de aquisição de dados que atua diretamente na redução de energia consumida no subsistema sensor. No primeiro caso, a redução de dados tem o objetivo de diminuir a quantidade de dados que é enviada à estação base. O processo de *in-network* consiste na agregação dos dados (por exemplo processar as médias de alguns valores) nos nós intermediários entre a fonte e a estação base. Desta maneira uma quantidade de dados é reduzida enquanto atravessam a rede até a estação base. A compressão de dados também pode ser aplicada para reduzir a quantidade de informação enviada pelo nó fonte. Esse esquema

envolve codificação de informação nos nós que geram os dados, e decodificação na estação base. A estratégia de predição de dados consiste em construir uma abstração sob um determinado fenômeno, descrevendo um modelo para a evolução dos dados. O modelo pode predizer valores a serem lidos pelos nós sensores com certa limitação de erro, diminuindo a quantidade de informação a ser transmitida (ANASTASI G.;CONTI, 2009).

A abordagem baseado em mobilidade explora a mobilidade dos nós. Porém , neste trabalho supomos que a posição dos nós é fixa e portanto esta abordagem não será discutida.

Enfim, vários esquemas de conservação de energia têm sido propostos. Nesta dissertação foca-se exatamente na abordagem *data-driven* considerando a redução das informações provenientes dos nós através da compressão de dados.

3 TÉCNICAS DE COMPRESSÃO DE DADOS PARA RSSFS

As técnicas de compressão reduzem o tamanho dos dados explorando características de sua estrutura. Os algoritmos de compressão em geral pertencem a duas classes: com perda e sem perda (MARCELLONI F. ; VECCHIO, 2009). Algoritmos sem perda garantem a integridade dos dados durante o processo de compressão e descompressão. Os algoritmos com perda geram perda de informação, porém na maioria das vezes garantem uma alta taxa de compressão.

A escolha do tipo de algoritmo depende do domínio da aplicação. Nas RSSFs, os dados são coletados por sensores, que devido à presença de ruído, produzem leituras diferentes mesmo quando não há mudança no fenômeno amostrado. Por essa razão, os fabricantes de sensores não especificam somente o *range* de operação, mas também sua precisão. Os *datasheets* expressam a precisão fornecendo uma margem de erro. Contudo, algumas aplicações críticas demandam sensores com alta precisão e também não toleram medições corrompidas pelo processo de compressão. Nas redes implantadas no corpo, por exemplo, nós sensores monitoram e registram permanentemente sinais vitais, onde pequenas variações de sinais são capturadas, proporcionando informação crucial para se fazer um diagnóstico. Assim acredita-se que os algoritmos de compressão sem perda escolhidos para uma RSSF devem ser estudados profundamente, pois esquemas clássicos de compressão de dados são geralmente impraticáveis para os minúsculos nós sensores, que são equipados tipicamente com alguns kilobytes de memória e um microprocessador de 4-8 MHz (KIMURA N.; LATIFI, 2005; SADLER; MARTONOSI, 2006; BARR K.C.; ASANOVIÉ, 2006).

A seguir serão descritos alguns conceitos básicos utilizados na abordagem do problema, dois algoritmos usados em RSSFs que serão utilizados na comparação de performance com o método proposto neste trabalho, e também o método de Huffman, utilizado na codificação do esquema proposto nesta dissertação.

3.1 CONCEITOS FUNDAMENTAIS NA ABORDAGEM DO PROBLEMA

Considera-se um nó sensor monitorando dados ambientais. Seja o dado adquirido pelo nó sensor no instante *i* representado depois da conversão analógica digital pelo símbolo $m_i \in \chi$, onde i = 0,1,2,... . O conjunto $\chi = \{x_0\ x_1 \cdots x_{M-1}\}$ é dito alfabeto fonte. Se cada símbolo $x_j \in \chi$ tem uma representação binária com l_j bits de comprimento, então o número médio de bits usados para representar cada símbolo fonte é dado por $L = \sum_{j=0}^{M-1} p_j\ l_j$, onde p_j é a probabilidade de que $m_i = x_j$. Quando não existe um codificador de fonte, a representação dos símbolos da fonte tem geralmente tamanho $L_u = \lceil \log_2 M \rceil$. O limite teórico para o mínimo número de bits por símbolo para uma fonte discreta é dado pela entropia da fonte $H(\chi)$ (SAYOOD, 2000), sendo:

$$H(\chi) = \sum_{j=0}^{M-1} p_j\ I_j = -\sum_{j=0}^{M-1} p_j\ \log_2\left(p_j\right), \tag{1}$$

onde I_j é a medida de informação para cada símbolo x_j definido por $log_2(\frac{1}{p_j})$. A eficiência do algoritmo de compressão pode ser dada pela comparação do tamanho médio do símbolo depois da compressão com a entropia da fonte.

3.2 O ALGORITMO LEC

O algoritmo LEC (do inglês *Lossless Entropy Compression*) proposto por Marcelloni e Vecchio (MARCELLONI F. ; VECCHIO, 2008, 2009) é proveniente de um esquema similar ao que foi baseado o JPEG, algoritmo para compressão de coeficientes DC de uma imagem digital (PENNEBAKER W.; MITCHELL, 1992). Tais coeficientes são caracterizados pela alta correlação, muito similar aos dados coletados pelas RSSFs. O LEC explora uma versão modificada do código Golomb-exponencial (GOLOMB, 1966) de ordem zero, que é uma espécie de código universal. A ideia básica é dividir o alfabeto em grupos de tamanho com crescimento exponencial. Cada palavra código é formada por um código unário e um binário: em particular, o código unário (um código de tamanho variável) especifica o grupo, enquanto o código binário (código de tamanho fixo) representa a indexação do grupo. No LEC, mantem-se a abordagem da divisão do alfabeto em grupos de tamanho incrementado exponencialmente, mas os grupos são codificados por entropia, ao invés de um código unário. Essa modificação introduz a possibilidade de especificar códigos livre de prefixo para os grupos, onde segundo (SRIDHARA, 2006) um código é livre de prefixo se nenhuma palavra código pertencente ao código é prefixo de outra palavra código no mesmo código.

Na unidade sensor do nó, cada medida m_i adquirida pelo sensor é convertida para uma representação binária com $l_i = R$ bits, onde R é a resolução do ADC (do inglês *Analog -to-Digital Converter*). A Figura 2 mostra o esquema de blocos do LEC. Para cada nova aquisição m_i, o LEC processa as diferenças $d_i = m_i - m_{i-1}$, que são entradas para um codificador por entropia [1]. O codificador por entropia executa uma compressão sem perdas das diferenças d_i e codifica-as com base nas suas estatísticas. Cada valor d_i diferente de zero é representado por uma sequência de bits bs_i composta de duas partes $s_i|a_i$, onde s_i codifica o número n_i de bits necessários para representar d_i (que é o grupo ao qual d_i pertence) e a_i que é a representação de d_i (o índice de posição no grupo). Quando d_i é igual a 0, o grupo correspondente tem tamanho igual a 1 e por isso não existe a necessidade de codificar o índice de posição no grupo, isso significa que a_i não é representado.

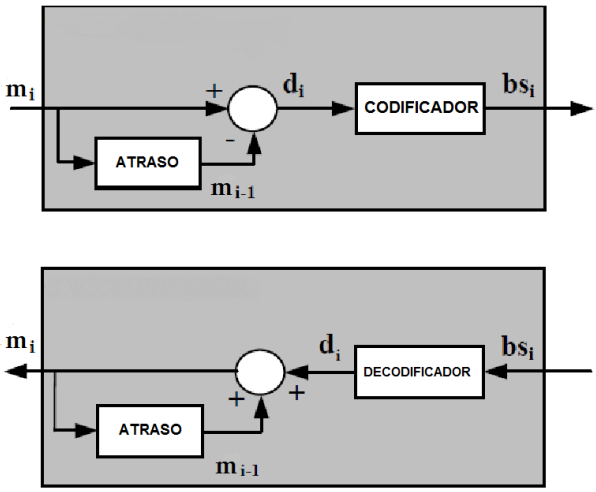

Figura 2: Diagrama de bloco proposto em (MARCELLONI, 2009).

Para d_i diferente de zero, n_i é processado como $\lceil \log_2(|d_i|) \rceil$, sendo n_i no máximo igual a R. Assim, para codificar n_i uma tabela de $R+1$ entradas livres de prefixo é especificada. Essa tabela depende da distribuição das diferenças d_i: diferenças com maior frequência estão associadas a códigos pequenos. Geralmente em dados coletados por RSSFs, verifica-se que as diferenças com maiores frequências estão próximas de 0. Assim, para evitar o custo de processar as frequências dos nós sensores, adota-se a Tabela 1 em que as primeiras 11 linhas coincidem com a tabela usada como base do algoritmo JPEG para compressão de coeficientes DC (PENNEBAKER W.; MITCHELL, 1992). Observa-se que se a resolução do ADC é maior do que R = 14 bits, a tabela tem que ser apropriadamente expandida.

Para gerenciar os possíveis valores negativos de d_i, o LEC mapeia as diferenças de entrada para valores não negativos $\widetilde{d}_i$, usando as seguintes relações:

[1] Para processar a primeira diferença d_0 não existe o valor anterior medido m_{-1}, assume-se então que ele seja igual ao valor central entre os 2^R valores discretos possíveis na conversão analógica digital.

Tabela 1: Dicionário usado no LEC

n_i	s_i	d_i
0	00	0
1	010	-1, +1
2	011	-3,-2,+2,+3
3	100	-7,...,-4,+4,...,+7
4	101	-15,...,-8,+8,...,+15
5	110	-31,...,-16,+16,...,+31
6	1110	-63,...,-32,+32,...,+63
7	11110	-127,...,-64,+64,...+127
8	111110	-255,...,-128,+128,...,+255
9	1111110	-511,...,-256,+256,...,+511
10	11111110	-1023,...,-512,+512,...,+1023
11	111111110	-2047,...,-1024,+1024,...,+2047
12	1111111110	-4095,...,-2048,+2048,...,+4095
13	11111111110	-8191,...,-4096,+4096,...,+8191
14	111111111110	-16383,...,-8192,+8192,...,+16383

$$\widetilde{d_i} = \begin{cases} d_i, & se \quad d_i \geq 0, \\ 2^{n_i} - 1 - |d_i|, & se \quad d_i < 0. \end{cases}$$

Finalmente, s_i é o valor correspondente a n_i na tabela livre de prefixo e a_i é a representação binária de $\widetilde{d_i}$ através de n_i bits. Uma vez que d_i é representado geralmente na notação complemento de dois, quando $d_i < 0$, a_i é igual aos n_i bits de menor grau de $d_i - 1$.

O procedimento usado para gerar a_i garante que todos os possíveis valores tenham códigos diferentes. Sendo m_1 = 30 e m_0 = 27 por exemplo, tem-se d_1 = m_1 - m_0 = 30 - 27 = +3, então: $n_1 = \lceil \log_2(|d_1|) \rceil = 2$. Na Tabela 1 com n_1 = 2 temos s_1 = 011 , e com d_1 = +3 (positivo) tem-se $\widetilde{d_1}$ = +3 , obtêm-se a_1 = 11 que é a representação binária de $\widetilde{d_1}$ através de n_1 bits , logo a codificação $bs_1 = s_1|a_1$ = 011|11. Considera-se agora m_2 = 18, sendo assim d_2 = m_2 - m_1 = 18 - 30 = -12 (negativo), então: $n_2 = \lceil \log_2(|d_2|) \rceil = 4$. Na Tabela 1 com n_2 = 4 temos s_2 = 101, e com d_2 = -12 (negativo) tem-se $\widetilde{d_2} = 2^{n_2} - 1 - |d_2| = 2^4 - 1 - |-12| = 3$, obtêm-se a_2 = 0011 que é a representação binária de $\widetilde{d_2}$ através de n_2 bits , logo a codificação $bs_2 = s_2|a_2$ = 101|0011. Uma vez gerado bs_i, ele é concatenado ao *bitstream* que forma a versão comprimida da sequência de medidas m_i (MARCELLONI F. ; VECCHIO, 2009).

3.3 O ALGORITMO ALFC

O algoritmo ALFC (do inglês *Adaptative Linear Filtering Compression*) proposto por Aaron Kiely *et al* (KIELY et al., 2010) consiste num estágio de predição onde cada valor de

amostra é predito baseado em valores de amostras do passado, e um estágio de codificação, onde um método de codificação por entropia é aplicado para codificar sem perdas os resíduos da predição, que é a diferença entre o valor predito e o valor atual da amostra. A predição adaptativa elimina a necessidade de determinar coeficientes de predição a *priori* e o mais importante é que torna o compressor dinamicamente ajustável para mudanças da fonte. Segundo os autores, isso é particularmente importante para dados sísmicos porque o comportamento da fonte pode variar drasticamente dependendo da atividade sísmica. Os valores de predição das amostras são usados para codificar sem perdas as amostras provenientes da fonte usando um esquema de código de tamanho variável. Encontra-se para cada valor de amostra um inteiro positivo e então codifica-se a sequência resultante usando códigos Golomb (GOLOMB, 1966). Essa estratégia é usada no algoritmo de codificação por entropia Rice (RICE, 1983) e no compressor de imagem LOCO-I (WEINBERGER M. ; SEROUSSI, 2000), entre outras aplicações.

Na definição do ALFC os autores supõem que um nó sensor gera dados unidimensionais que produzem valores inteiros de amostras $m_0, m_1, \ldots$, etc. O instrumento tem um *range* dinâmico de R bits, e sem perda de generalidade, pode-se assumir que cada valor de amostra está no range $[-2^{R-1}, 2^{R-1}-1]$. Cada nó tem um potencial computacional modesto, assim a abordagem de compressão deve ter uma complexidade baixa. Os valores de amostras codificadas são transmitidos usando pacotes de tamanho fixo. Um problema significativo é a perda frequente de pacotes no canal. Por essa razão é imposta uma restrição adicional, tal que os pacotes recebidos não devem depender do conteúdo de outros pacotes. Para isso em cada pacote é incluído um cabeçalho contendo informação necessária para independência entre os pacotes.

3.3.1 PREDIÇÃO

3.3.1.1 PREDIÇÃO LINEAR ADAPTATIVA

Admite-se uma estimativa $\widehat{\mu_i}$ da média dos valores do sinal de entrada. Essa estimativa é usada para processar uma versão despolarizada g_i das amostras m_i, tal que

$$g_i = m_i - \widehat{\mu_i}.$$

Aplica-se a predição linear adaptativa de Q-ésima ordem para os sinais despolarizados g_i. O valor de predição $\widehat{g_i}$ é uma combinação linear dos $Q-1$ valores despolarizados precedentes,

$$\widehat{g}_i = \sum_{j=0}^{Q-1} w_j \cdot g_{i-j} = \mathbf{w}_i^T \mathbf{g}_i, \tag{2}$$

onde $\mathbf{w}_i = [w_0, w_1, \cdots, w_{Q-1}]^T$ é um vetor de coeficientes de pesos que são adaptados para a fonte e $\mathbf{g}_i = [g_i, g_{i-1}, \cdots, g_{i-(Q-1)}]^T$ é o vetor dos $Q-1$ valores precedentes de amostras despolarizados. A predição das amostras é

$$\widehat{m}_i = \widehat{\mu}_i + \widehat{g}_i, \tag{3}$$

o erro estimado, ou resíduo da predição é

$$e_i = m_i - \widehat{m}_i = g_i - \widehat{g}_i, \tag{4}$$

sendo que a predição $\widehat{m}_i$ é usada para codificar sem perda a amostra da fonte m_i usando um esquema de codificação de tamanho variável. O procedimento de codificação por entropia leva em consideração o fato de que a amostra da fonte m_i é um inteiro, enquanto que $\widehat{m}_i$ frequentemente não é.

Após a codificação de m_i, usa-se o algoritmo *sign* (GERSHO, 1984) para atualizar o vetor de pesos, tal que

$$\mathbf{w}_{i+1} = \mathbf{w}_i + \alpha \cdot \mathbf{g}_i \cdot sign(e_i), \tag{5}$$

e a estimativa do valor médio é atualizada por

$$\widehat{\mu}_{i+1} = \widehat{\mu}_i + \beta \cdot (m_i - \widehat{\mu}_i), \tag{6}$$

onde α e β são parâmetros que controlam a adaptação do vetor pesos e a estimativa da média para as estatísticas da fonte.

3.3.1.2 PREDIÇÃO USANDO ARITMÉTICA INTEIRA

Para eliminar as operações de ponto flutuante no algoritmo básico da seção anterior usam-se aproximações racionais para valores de quantidades reais para produzir uma versão do algoritmo que requer somente aritmética inteira. Especificamente, as quantidades de valor real $\widehat{m}_i$, $\widehat{\mu}_i$, $\widehat{g}_i$, e_i e $\mathbf{w}_i$ são aproximadas usando valores racionais tais como,

$$\widehat{g}_i = \widehat{G}_i/2^S, \tag{7}$$

$$\widehat{m}_i = \widehat{M}_i/2^S, \tag{8}$$

$$\widehat{\mu}_i = \widehat{\Omega}_i/2^S, \tag{9}$$

$$e_i = E_i/2^S, \tag{10}$$

$$\mathbf{w}_i = \frac{1}{2^S}\mathbf{W}_i, \tag{11}$$

onde S é algum inteiro fixo, $\widehat{G}_i$, $\widehat{M}_i$, $\widehat{\Omega}_i$ e E_i são variáveis inteiras, e $\mathbf{W}_i$ é um vetor de inteiros. O valor de S controla a resolução dos cálculos executados na predição linear. Nessa versão do algoritmo, implementam-se operações de arredondamento que resultam em valores de g_i inteiro, e, consequentemente, $\mathbf{g}_i$ será um vetor de inteiros. Os parâmetros de adaptação α e β são escolhidos de forma que $\alpha = 2^{-A}$, $\beta = 2^{-B}$ para inteiros A e B tal que uma multiplicação de atualização pode ser implementada com uma operação de deslocamento de bits.

Cada iteração para versão de inteiros do algoritmo consiste de:

(i) Processe

$$\widehat{G}_i = \mathbf{W}_i^T \mathbf{g}_i; \tag{12}$$

(ii) Processe

$$\widehat{M}_i = \widehat{G}_i + \widehat{\Omega}_i; \tag{13}$$

(iii) Codifique a amostra inteira m_i usando o valor de predição racional dado por

$$\widehat{m}_i = \widehat{M}_i/2^S; \tag{14}$$

(iv) Processe o valor despolarizado (inteiro)

$$G_i = m_i - \lfloor(\widehat{\Omega}_i + 2^{S-1} - 1)/2^S\rfloor; \tag{15}$$

(v) Processe o erro de predição

$$E_i = g_i.2^S - \widehat{G}_i; \tag{16}$$

(vi) Atualize o vetor de pesos

$$\mathbf{W}_{i+1} = \mathbf{W}_i + sign(E_i)\lfloor 2^S\mathbf{g}_i + (2^{A-1}-1)\mathbf{I})/2^A\rfloor; \tag{17}$$

onde **I** representa um vetor de 1's.

(vii) Atualize a média do valor estimado

$$\widehat{\Omega}_{i+1} = \widehat{\Omega}_i - \lfloor(\widehat{\Omega}_i - m_i.2^S + 2^{B-1} - 1)/2^B\rfloor. \tag{18}$$

3.3.2 CODIFICAÇÃO POR ENTROPIA

A codificação por entropia é implementada codificando uma sequência de valores inteiros de amostras m_i dado o valor de predição $\widehat{m}_i$. É encontrada para cada amostra um valor inteiro positivo residual f_i que é codificado usando-se um código Golomb.

Para refinar o valor da predição $\widehat{m}_i$, define-se:

$$[\widetilde{m}_i] = min\{max\{truncamento(\widehat{m}_i), m_{min}\}, m_{max}\}, \tag{19}$$

onde $m_{min} = -2^{R-1}$ e $m_{max} = 2^{R-1} - 1$ são os valores de amostras máximo e mínimo possíveis. Usa-se esse refinamento de predição para calcular o valor inteiro de predição residual

$$\widetilde{e}_i = m_i - [\widetilde{m}_i]. \tag{20}$$

Encontra-se para o inteiro $\widetilde{e}_i$ um inteiro estritamente positivo f_i usando uma leve variação no procedimento usado em (RICE, 1983; (CCSDS), 1997), dado por:

$$f_i = \begin{cases} 2|\widetilde{e}_i| - \zeta_i & \text{se} \quad |\widetilde{e}_i| \leq \theta, \\ |\widetilde{e}_i| + \theta & \text{caso contrário,} \end{cases} \tag{21}$$

sendo,

$$\theta = min\{[\widehat{m}_i] - m_{min}, m_{max} - [\widehat{m}_i]\}, \tag{22}$$

e

$$\zeta_i = \begin{cases} 1, & \text{sign}(\widetilde{e_i}) = \text{sign}(\widehat{m}_i - [\widehat{m}_i]), \\ 0 & \text{caso contrário.} \end{cases} \tag{23}$$

Essa metodologia é reversível e garante que $f_i \in [0, 2^R - 1]$, isto é, f_i é um inteiro positivo. A partir do valor de f_i gera-se uma palavra código que consiste na representação unária $\lfloor f_i/2^k \rfloor$ 0´s seguido por um 1 concatenado com os k bits menos significativos do binário representado por f_i, onde k é um parâmetro de otimização da palavra código calculado de acordo com alguns critérios (KIELY et al., 2010). Por exemplo, sejam os parâmetros $A = 15, B = 8, R = 14, Q = 3$ e o vetor de pesos inicializado com $\{w_0, w_1, w_2\} = \{1.5, -1.25, 1.0\}$. Supondo $m = \{m_o, m_1, m_2, m_3, ...\} = \{90, 94, 92, 94, ...\}$, os primeiros Q valores despolarizados são $\{g_0, g_1, g_2\} = \{-4, 2, -2\}$. Aplicando-se o algoritmo ALFC para a amostra $i = 3$, obtêm-se $f_i = 14$ e parâmetro $k = 3$. A representação unária da codificação é $\lfloor f_i/2^k \rfloor = \lfloor 14/8 \rfloor = 1$ zero seguido de 1, ou seja, 01 concatenado com a representação binária dos $k = 3$ bits menos significativos de 14 em binário: 110. Tem-se então a saída codificada com o valor 01110. Nesse caso o valor original a ser codificado é 94, podendo ser representado pelo valor binário 1011110, com 7 bits sem nenhuma codificação. O resultado do ALFC nos leva para o valor 01110 de 5 bits. A implementação do ALFC foi validada processando-se o exemplo descrito em (KIELY et al., 2010), sendo gerada a mesma codificação ao final para os mesmos parâmetros de entrada.

3.4 CÓDIGO DE HUFFMAN: ALGORITMO

Essa técnica foi desenvolvida por David Huffman em 1951 (SAYOOD, 2000), e os códigos gerados usando esta técnica foram chamados de códigos de Huffman. O procedimento é baseado em duas observações:

1) Num código otimizado, símbolos que ocorrem mais frequentemente (tem uma alta probabilidade de ocorrência) terão palavras códigos menores do que símbolos que ocorrem com menos frequência;

2) Num código otimizado, os dois símbolos que ocorrem com menos frequência tem o mesmo tamanho de palavra código.

Os procedimentos de Huffman são obtidos adicionando um requisito simples para essas duas observações. Esse requisito é que as palavras códigos correspondentes para os dois símbolos de menor probabilidade diferem entre si somente do último bit. Sejam γ e λ dois símbolos com menor probabilidade no alfabeto (podem existir mais de dois símbolos empatados com a menor probabilidade, escolhe-se aleatoriamente dois), se a palavra código de γ for r*0,

Tabela 2: Alfabeto inicial com cinco símbolos

Símbolo	Probabilidade	Palavra Código
x_1	0,4	c(x_1)
x_0	0,2	c(x_0)
x_2	0,2	c(x_2)
x_3	0,1	c(x_3)
x_4	0,1	c(x_4)

Tabela 3: Alfabeto reduzido para quatro símbolos

Símbolo	Probabilidade	Palavra Código
x_1	0,4	c(x_1)
x_0	0,2	c(x_0)
x_2	0,2	c(x_2)
x'_3	0,2	φ_1

a palavra código para λ será *r**1. Aqui *r* é um *string* de 1s e 0s, e * denota concatenação. Esse requisito não viola as duas observações e leva a um procedimento simples de codificação.

Exemplo: Construindo o código de Huffman.

Será construído o código de Huffman para uma fonte que produz símbolos de um alfabeto χ com probabilidade p_j. Seja $\chi = \{x_0\ x_1\ x_2\ x_3\ x_4\}$ com $p_0 = p_2 = 0{,}2$, $p_1 = 0{,}4$, e $p_3 = p_4 = 0{,}1$. A entropia, para essa fonte é 2,122 bits/símbolo, dado por: $H = -\sum_{j=0}^{4} p_j . \log_2 p_j$. Para construir o código de Huffman, primeiro ordenam-se os símbolos na ordem decrescente do valor de probabilidade conforme indicado na Tabela 2. Aqui c(x_j) representa a palavra código de x_j. Os dois símbolos com menor probabilidade são x_3 e x_4. Então pode-se escrever suas palavras código como

$$\text{c}(x_3) = \varphi_1 * 0,$$

$$\text{c}(x_4) = \varphi_1 * 1,$$

onde φ_1 é um *string* binário.

Então define-se um novo alfabeto χ' com quatro símbolos, x_0,x_1,x_2, x'_3, onde x'_3 é a composição de x_3 e x_4 e tem uma probabilidade $p_{3'} = p_3 + p_4 = 0{,}2$. Ordena-se esse novo alfabeto em ordem decrescente para obter a Tabela 3. Nesse alfabeto, x_2 e x'_3 são as duas letras do final da lista ordenada. Constroem-se suas palavras códigos como

$$\text{c}(x_2) = \varphi_2 * 0,$$

Tabela 4: Alfabeto reduzido para três símbolos

Símbolo	Probabilidade	Palavra Código
x_1	0,4	c(x_1)
x_2'	0,4	φ_2
x_0	0,2	c(x_0)

$$c(x_3') = \varphi_2 * 1,$$

mas $c(x_3') = \varphi_1$, então,

$$\varphi_1 = \varphi_2 * 1.$$

Consequentemente,

$$c(x_3) = \varphi_2 * 10,$$

$$c(x_4) = \varphi_2 * 11.$$

Nesse estágio, novamente define-se um alfabeto χ'' que consiste em três símbolos x_0, x_1 e x_2', onde x_2' é a composição de x_2 e x_3' e tem uma probabilidade $p_2' = p_2 + p_3' = 0{,}4$. Ordena-se esse novo alfabeto em ordem decrescente obtendo-se a Tabela 4. Nesse caso, os dois símbolos com menor probabilidade são x_0 e x_2', então,

$$c(x_2') = \varphi_3 * 0,$$

$$c(x_0) = \varphi_3 * 1,$$

mas $c(x_2') = \varphi_2$, logo,

$$\varphi_2 = \varphi_3 * 0.$$

Consequentemente,

$$c(x_2) = \varphi_3 * 00,$$

$$c(x_3) = \varphi_3 * 010,$$

$$c(x_4) = \varphi_3 * 011,$$

Tabela 5: Alfabeto reduzido para dois símbolos

Símbolo	Probabilidade	Palavra Código
x_2''	0,6	φ_3
x_1	0,4	c(x_1)

Tabela 6: Código de Huffman para o alfabeto original com cinco símbolos

Símbolo	Probabilidade	Palavra Código
x_1	0,4	1
x_0	0,2	01
x_2	0,2	000
x_3	0,1	0010
x_4	0,1	0011

Novamente define-se um alfabeto, agora somente com dois símbolos x_2'' e x_1, aqui x_2'' é a composição dos símbolos x_2' e x_0 e tem uma probabilidade $p_2'' = p_2' + p_0 = 0{,}6$. Agora tem-se a Tabela 5. Como temos apenas dois símbolos, a palavra código é construída diretamente: c(x_2'') = 0 e c(x_1) = 1, como consequência temos que $\varphi_3 = 0$, o que significa: c(x_0) = 01, c(x_2) = 000, c(x_3) = 0010, c(x_4) = 0011, e o código de Huffman é dado pela Tabela 6.

O tamanho médio desse código é L = 0,4 x 1 + 0,2 x 2 + 0,2 x 3 + 0,1 x 4 + 0,1 x 4 = 2,2 bits/símbolo. Pode-se observar que este valor está muito próximo do valor da entropia da fonte calculada no início desta seção com H = 2,122 bits/símbolo. Uma maneira alternativa de construir-se o código de Huffman é utilizando o fato de que pela virtude de ser um código livre de prefixo, este pode ser representado por uma árvore binária completa com v nós externos e v - 1 nós internos, onde os nós externos são rotulados com as probabilidades de ocorrência de cada símbolo (SAYOOD, 2000; MELLO, 2006).

4 DESCRIÇÃO DO MÉTODO

Neste Capítulo apresenta-se o método de compressão proposto nesta dissertação, baseado num dicionário fixo de Huffman.

4.1 DEFINIÇÃO DO PROBLEMA

Considera-se um nó sensor monitorando dados ambientais. Para o projeto do método proposto considera-se o conjunto de medições de temperatura denominado *Set 1* proveniente de medições realizadas em Hargestown, MD, EUA, no período de 01/01/09 à 08/07/11 com 26.843 amostras conforme descrito na Tabela 7. Truncasse os valores do *Set 1* para obtermos os valores inteiros das medições com variação entre $-16°$C e $+37°$C, sem compressão, necessita-se usar $L_u = 6$ bits/símbolo para representar os 54 diferentes símbolos do alfabeto. A entropia da fonte nesse caso é de $H = 5,29$ bits/símbolo. Implementando um código Huffman para essa fonte em particular, após a compressão obtem-se $L = 5,31$ bits/símbolo. Contudo, nota-se que não obtem-se um bom ganho fazendo-se esta compressão, apenas consegue-se uma redução do tamanho médio do símbolo de 0,69 bits ou seja $11,5\%$ em relação ao caso não codificado. Isso ocorre porque a distribuição de probabilidade dos valores de temperatura é relativamente uniforme, enquanto que um esquema de compressão como o Huffman tem maior impacto no caso de uma distribuição fortemente não uniforme. O desempenho pode melhorar, como em (MARCELLONI F. ; VECCHIO, 2009), se considerarmos as diferenças das medições consecutivas de temperatura. Por exemplo, no caso do *Set 1* a entropia das diferenças de temperatura é somente $H_d = 2,13$ bits/símbolo, tendo neste caso uma redução de $59,7\%$ comparado com a entropia das temperaturas do mesmo *Set 1*. Tal redução é devido ao fato de que a distribuição de probabilidade das diferenças é fortemente não uniforme. Assim, é muito mais promissor considerar a compressão das diferenças consecutivas de temperatura do que a compressão das próprias temperaturas.

Aplicando o LEC (MARCELLONI F. ; VECCHIO, 2009) nas temperaturas do *Set 1* obtem-se $L = 3,24$ bits/símbolo, tendo assim uma redução de $46,0\%$ considerando o valor

Tabela 7: Principais características dos conjuntos de temperaturas

Localização (Latitude , Longitude)	Nome	Range (°C)	Amostras	Data (dd/mm/aa)	Tempo entre amostras
Hagerstown, MD, EUA (39.711,-77.722)	Set 1	-16 a +37	26.843	01/01/09 a 08/07/11	10 min
Manaus, AM, Brasil (-3.145,-59.986)	Set 2	+21 a +36	3.676	01/07/11 a 30/11/11	60 min
Jonesboro, AR, EUA (35.834,-90.649)	Set 3	-1 a +41	3.738	01/07/11 a 20/11/11	60 min
Le Génépi, Suíça (46.025,7.044)	Set 4	-11 a +16	42.141	28/08/07 a 31/10/07	2 min
Morges, Suíça (46.494,6.472)	Set 5	+5 a +29	14.527	06/08/07 a 02/09/07	2 min
Bern, Suíça (46.948,7.444)	Set 6	-14 a +6	4.851	13/03/07 a 15/03/07	0.5 min

original de 6 bits/símbolo necessário para o *Set 1*, mas ainda $18,5\%$ a mais do que o limite teórico dado pela entropia das diferenças de temperatura ($H_d = 2,13$ bits/símbolo). O limite teórico pode ser praticamente alcançado por um código de Huffman, produzindo $L = 2,16$ bits/símbolo, uma redução de $64,0\%$ considerando o valor original de 6 bits/símbolo. Recorde que o LEC usa um dicionário fixo, que pode ser aplicado em qualquer fonte. Uma técnica de codificação por entropia como Huffman tem que ser combinada com a distribuição da fonte para atingir uma performance otimizada (SAYOOD, 2000). No entanto, existe um problema de causalidade, pois não se pode conhecer a distribuição exata *a priori*. A fim de contornar esse problema pode-se fazer uso da técnica de codificação Huffman adaptativa ou dinâmica (VITTER, 1987). A principal desvantagem dessa abordagem é que ela requer uma mudança no dicionário a cada conexão entre os pares de nós vizinhos (REINHARDT A. ; CHRISTIN, 2010). Devido à severa restrição de memória nos nós sensores sem fio, esse método é inteiramente impraticável. Sendo assim, na sequência propomos um método de compressão para RSSF baseado num dicionário Huffman fixo, trabalhando sobre as diferenças das medidas.

4.2 MÉTODO PROPOSTO

Tem-se como objetivo idealizar um método simples de compressão que implemente uma codificação por entropia otimizada baseada num dicionário fixo. Depois de comparar a distribuição de probabilidade das temperaturas e das diferenças consecutivas de temperatura para conjuntos de medidas realizadas em diferentes localidades, notou-se que as distribuições das diferenças eram bastante similares para todos os conjuntos, apesar da distribuição dos valores de temperatura variar significativamente. Isso pode ser observado na Figura 3, que mostra a distribuição de probabilidade das diferenças entre medições consecutivas para cada conjunto de dados da Tabela 7[1]. Como mostra a Figura 3, todas as distribuições são aproximadamente normais com média zero. O mais importante, é notar que para todos os conjuntos a lista dos mais prováveis para os menos prováveis segue a sequência

[1] *Sets* 1 a 3 (UNDERGROUND, 2012) e *Sets* 4 a 6 (SENSORSCOPE, 2012)

$(0, \pm 1, \pm 2, \pm 3, \pm 4, \cdots)$. Assim, a princípio é possível usar um alfabeto Huffman fixo para comprimir os diferentes conjuntos de medidas se considerarmos as diferenças das temperaturas, pois todos os conjuntos tem um comportamento similar, o que nos leva a possibilidade de ter um bom desempenho para todos os cenários. Observa-se também que a taxa de amostragem dos conjuntos tem influência no formato das gaussianas obtidas na representação das distribuições, tendo os *Sets* com menor taxa de amostragem uma menor variância. Em particular, nesse trabalho propomos a construção de um alfabeto Huffman fixo obtido pela aplicação do algoritmo de Huffman para um extenso conjunto de temperaturas medidas. Considerou-se o *Set 1* como nosso conjunto de referência, sem nenhuma razão em particular além do fato de que ambos número de amostras e o *range* de temperaturas medidas são relativamente grandes. Ainda neste trabalho serão mostrados os resultados utilizando os outros *Sets* como referência.

O alfabeto mostrado na Tabela 8 é então usado para comprimir conjuntos diferentes de dados apresentados na Tabela 7. Como o alfabeto é fixo, a complexidade desta abordagem é baixa, sendo não mais complexa do que o esquema apresentado em (MARCELLONI F. ; VECCHIO, 2009). Por exemplo, é válido citar que uma implementação de codificação e decodificação Huffman para um microcontrolador AVR, usado num nó sensor, utiliza somente 468 bytes de memória de programa (AVR, 2012), sendo portanto realmente de baixa complexidade.

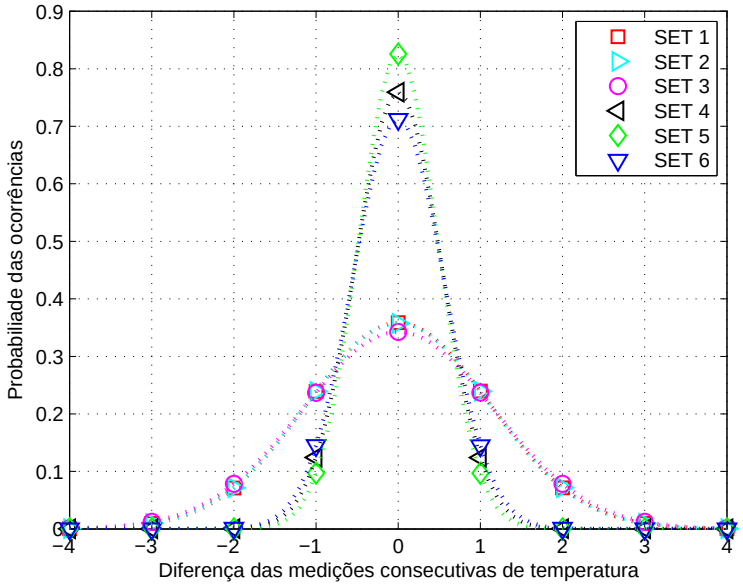

Figura 3: Distribuição das probabilidades de ocorrência das diferenças de temperatura para os conjuntos de dados da Tabela 7.

No esquema de compressão proposto, como em qualquer um baseado em dicionário fixo e na abordagem de compressão de diferenças, dois casos especiais devem ser considerados:

Tabela 8: Alfabeto de Huffman Proposto

d_i	**Códigos**
-10	0010101000101110
-9	00101010001010
-8	001010100010110
-7	001010100011
-6	001010100111
-5	00101010010
-4	001010101
-3	0010100
-2	00100
-1	01
0	1
+1	000
+2	0011
+3	001011
+4	00101011
+5	00101010000
+6	001010100110
+7	00101010001001
+8	00101010001000
δ	0010101000101111

i) No início do pacote de dados a primeira amostra m_0 será transmitida sem compressão, já que não existe valor anterior para computar a diferença d_0. A partir da segunda amostra coletada tem-se as diferenças $d_i = m_i - m_{i-1}$ que poderão ser processadas e comprimidas. O envio desta primeira amostra sem compressão no início do pacote é importante para o processo de descompressão, já que se trata de um caso diferencial; ii) A Tabela 8 possui um alfabeto limitado, pois trata-se de um dicionário fixo, que terá uma quantidade de símbolos dependente da referência usada, neste caso em particular o *Set 1*. Apesar da probabilidade de ocorrência de símbolos fora do dicionário ser baixa, considerando a distribuição apresentada na Figura 3, deve-se considerar esta possibilidade. A solução viável quando da ocorrência de símbolos não presentes no dicionário é tratá-los como uma medição inicial e enviá-lo como um valor sem compressão dentro do pacote de informação. Nesse caso, um símbolo sem compressão pode ser transmitido no pacote de dados usando um marcador especial δ, presente no dicionário de Huffman, antes do valor sem compressão com o objetivo de identificá-lo no meio do pacote de informação.

Na sequência avaliamos o desempenho do método proposto quando usado na compressão dos dados de temperatura mostrados na Tabela 7. Comparações são feitas com os esquemas LEC e ALFC, além do limite teórico. Variações no método são discutidas, assim

como sua extensão para outros tipos de dados além de temperatura.

5 RESULTADOS

5.1 COMPARANDO COM LEC

Nessa seção investigou-se a performance do esquema proposto, quando o alfabeto de Huffman fixo da Tabela 8 é usado para comprimir os conjuntos de temperaturas da Tabela 7. Note que, como mostrado na Tabela 7, os conjuntos de dados em teste, *Set 2* até *Set 6*, foram coletados em diferentes locais e datas do *Set 1*, que foi usado como referência para construir o alfabeto da Tabela 8. A performance do esquema proposto é comparada com o limite teórico dado pela entropia da fonte (considerando ambas temperaturas e diferenças de temperaturas) e com a performance do LEC.

A Figura 4 mostra as entropias (H) das medições e das diferenças consecutivas (H_d), e o tamanho médio do símbolo sem compressão (L_u), para cada conjunto de dados mostrado na Tabela 7. A Figura 5 mostra o tamanho médio por símbolo (L) depois da compressão, a taxa de compressão (C_r) e a eficiência do código (η) usando a abordagem proposta, bem como para o LEC. A taxa de compressão é calculada como :

$$C_r = 100 \times \left(1 - \frac{L}{L_u}\right)\%, \tag{24}$$

enquanto a eficiência do código em relação ao respectivo limite teórico é dada por:

$$\eta = 100 \times \left(\frac{H_d}{L}\right)\%. \tag{25}$$

Como o resultado demonstra, o método proposto não só supera o LEC mas também atinge uma alta taxa de compressão. Além disso, a eficiência do código no método proposto se aproxima de 100 % para alguns conjuntos de dados (*Sets 2, 3 e 6*). Isso significa que apesar de se usar um dicionário fixo, o esquema proposto atingiu um bom desempenho para todos os conjuntos de dados. Ainda pode-se observar que a redução do tamanho médio dos símbolos foi de aproximadamente 50% comparado com o LEC.

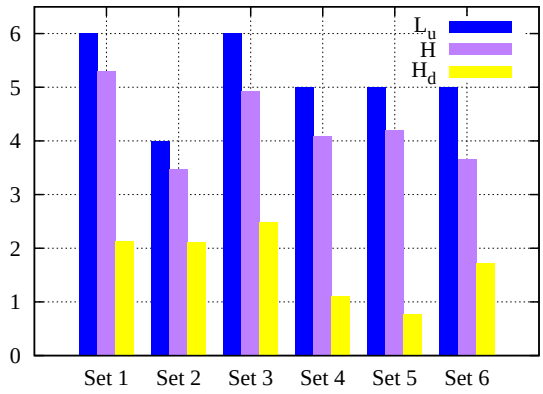

Figura 4: Tamanho médio do símbolo (L_u) sem compressão, entropia das temperaturas medidas (H) e entropia das diferenças consecutivas de temperaturas (H_d), todos expressos em [bits / símbolo]

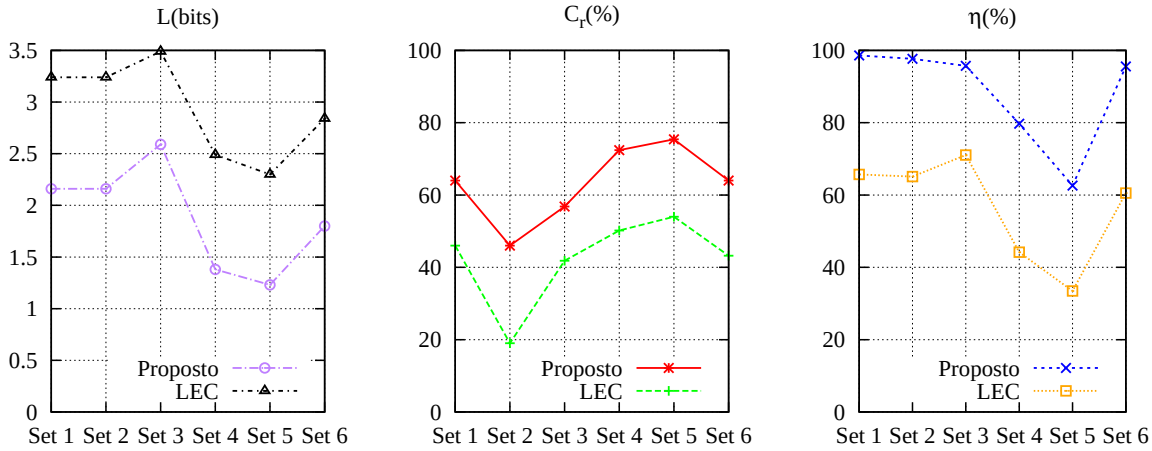

Figura 5: Tamanho médio do símbolo após a compressão (L), taxa de compressão (C_r) e eficiência do código (η) para os diferentes conjuntos de temperatura.

5.2 COMPARANDO COM ALFC

Para avaliar a performance do ALFC, este foi aplicado nos conjuntos de dados *Set 1* a *Set 6* com os parâmetros A = 15, B = 8, R = 14, Q = 3, os mesmos usados na aplicação numérica apresentada em (KIELY et al., 2010), e b = 6 considerando que este é o valor máximo de bits necessário para representar os dados sem codificação. Na Figura 6 são apresentados os resultados do tamanho médio dos símbolos após compressão (L) (bits/símbolo) aplicando o algoritmo ALFC e o método proposto nesta dissertação respectivamente. Nesta investigação os *Sets* foram comprimidos usando seu tamanho máximo num único pacote (Max) e também particionando estes em pequenos pacotes de 50, 100 e 400 amostras, uma vez que em (KIELY et al., 2010) foram usados nas simulações pacotes de 280 amostras. Para cada conjunto de partição foi calculado uma média geral dos tamanhos médios dos símbolos obtidos por partição. Para cada pacote processado usando o ALFC, foi calculado o valor ótimo do parâmetro k, que está

particularmente ligado ao tamanho do código binário gerado resultante da compressão.

Pode-se constatar que no ALFC os valores de *L* tendem a reduzir com o particionamento dos *Sets*, tendo assim uma melhor performance com pacotes de tamanho menor, o que por outro lado não ocorreu com o método proposto nesta dissertação, onde verificou-se uma tendência de aumento nos valores de *L*. Isto de certa forma era esperado, pois a cada particionamento com tamanho de blocos menores mais símbolos sem compressão são incluídos, considerando que o primeiro símbolo é enviado sem compressão.

Constatou-se que todos os valores de *L* são menores para os pacotes comprimidos com a aplicação do método proposto, tendo como os piores casos os pacotes de 50 amostras[1]. Também verificou-se que o ALFC possui seis parâmetros variáveis de otimização, o que o torna a princípio mais complexo que o método proposto.

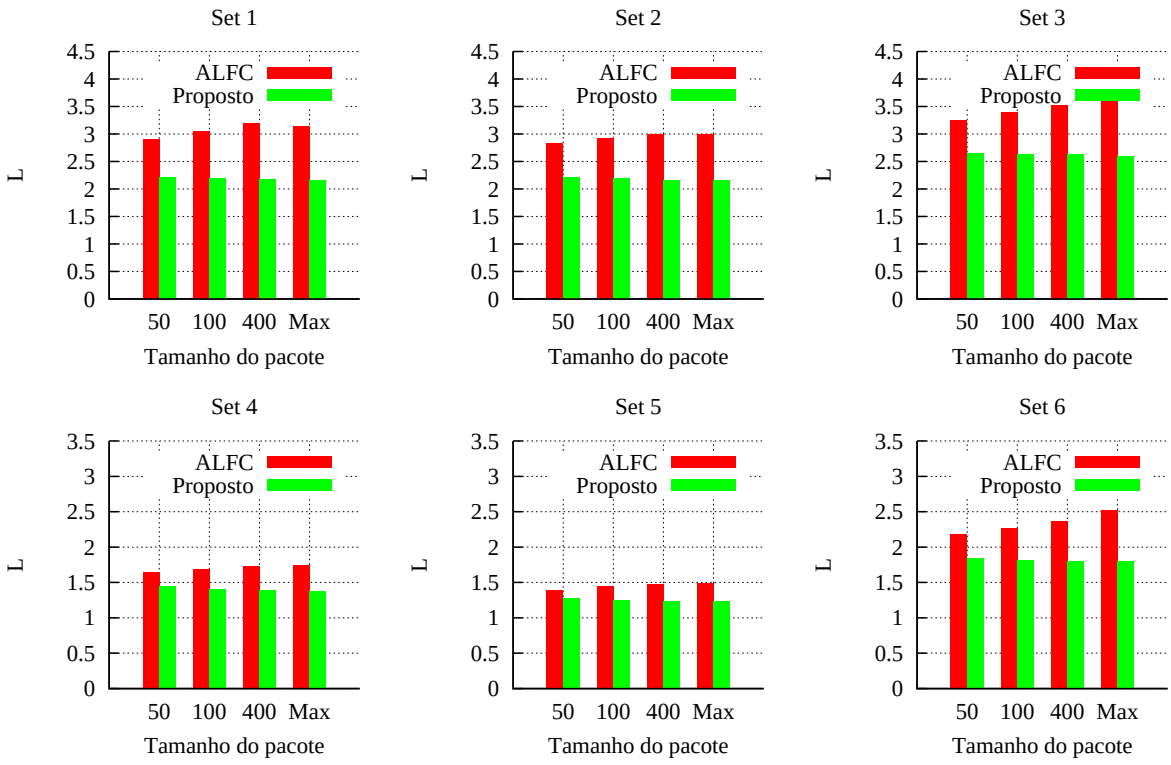

Figura 6: Tamanho médio do símbolo (*L*) (bits/símbolo) depois da compressão usando ALFC e o método proposto para tamanhos diferentes de pacotes

[1]Vale ressaltar que não foi incluído o *overhead* previsto no ALFC nos resultados, o que contribuiria para piorar ainda mais o seu desempenho final.

5.3 VALIDAÇÃO DO MÉTODO USANDO DICIONÁRIOS ALTERNATIVOS

Como mencionado anteriormente na Seção 4.2, foi considerado o *Set 1* como o conjunto de referência para geração do dicionário fixo que seria aplicado na compressão dos demais conjuntos. Investigou-se então qual seria o impacto no uso dos demais *Sets* como referência na geração do dicionário fixo. Na Tabela 9 é apresentado o valor dos tamanhos médios dos símbolos (L) aplicando o método proposto utilizando como referência os *Sets* de cada coluna aplicados aos *Sets* por linha. Na última linha apresenta-se o tamanho médio dos símbolos considerando todos os *Sets* e usando um dicionário baseado no *Set* do topo da coluna. Foi observado que os *Sets 1,2,4,5 e 6* reproduziram valores de tamanho médio dos símbolos com certa uniformidade, apenas o *Set 3* reproduziu valores de tamanho médio dos símbolos com valores 17% acima dos outros, conforme observado na linha que representa as médias das colunas. Isso ocorre pois este *Set* possui o menor valor de probabilidade de ocorrência da média zero, o que gerou maiores códigos para representação dos símbolos. Foi mostrado também o desvio padrão (σ) entre os valores obtidos para cada *Set*, observando maior desvio no *Set 5* quando suas amostras foram comprimidas. Apesar das variações registradas, nenhuma das referências levou a valores maiores de L comparado ao LEC e apenas em duas situações quando o *Set 3* foi tomado como referência e aplicado nos *Sets 4 e 5* geraram valores maiores de L comparado ao ALFC.

Na Tabela 10 é apresentado o impacto da inserção de marcadores especiais proveniente da geração de símbolos não presentes no dicionário Huffman fixo gerado pelo conjunto de dados de referência. Nas colunas temos os conjuntos de referência e nas linhas os impactos correspondentes a cada conjunto comprimido, ou seja, o percentual de marcadores especias que seriam necessários no bloco de dados para transmissão. Pode-se constatar que os piores casos ocorrem quando os *Sets 4 e 5* são as referências na geração do dicionário Huffman, pois estes geram dicionários com variabilidade nas diferenças menor do que os outros (*Set 4* [-3 a +3] e *Set 5* [-6 a +5]), fazendo com que exista uma ocorrência um pouco maior de símbolos fora do dicionário. Mas mesmo assim a ocorrência de símbolos fora do dicionário é pequena, causando pouco impacto na fase de compressão.

Conclui-se que apesar das diferenças existentes considerando-se o uso de cada um dos *Sets* como referência na geração do dicionário Huffman, todos os resultados superam os valores obtidos a partir do LEC e como também o ALFC, com exceção de dois casos , o que não torna fortemente restrita a escolha do conjunto de referência.

Tabela 9: Tamanho médio dos símbolos após compressão (L) no cruzamento entre as bases de dados

	Set 1	***Set 2***	***Set 3***	***Set 4***	***Set 5***	***Set 6***	σ
Set 1	2,16	2,17	2,31	2,19	2,25	2,20	0,06
Set 2	2,16	2,15	2,31	2,17	2,21	2,19	0,06
Set 3	2,59	2,59	2,56	2,58	2,72	2,65	0,06
Set 4	1,38	1,38	2,02	1,38	1,38	1,38	0,26
Set 5	1,23	1,23	2,00	1,22	1,22	1,22	0,32
Set 6	1,80	1,80	2,19	1,78	1,83	1,80	0,16
Média	1,89	1,89	2,31	1,89	1,94	1,91	-

Tabela 10: Impacto da inserção de marcadores especiais proveniente da geração de símbolos não presentes no dicionário Huffman fixo gerado pelo conjunto de dados de referência(%)

	Set 1	***Set 2***	***Set 3***	***Set 4***	***Set 5***	***Set 6***
Set 1	0,00	0,03	0,03	0,79	1,77	0,03
Set 2	0,00	0,00	0,03	0,98	1,17	0,03
Set 3	0,00	0,03	0,00	2,84	3,18	0,03
Set 4	0,00	0,00	0,00	0,00	0,01	0,00
Set 5	0,00	0,00	0,00	0,04	0,00	0,00
Set 6	0,00	0,00	0,00	0,80	1,05	0,00

5.4 AVALIAÇÃO USANDO DADOS DE UMIDADE RELATIVA

O esquema proposto pode ser aplicado em outros tipos de dados ambientais. Para demonstrar isto, considerou-se as medições de umidade relativa listadas na Tabela 11 (SENSORSCOPE, 2012). A Figura 7 lista as entropias das umidades relativas, as entropias das diferenças consecutivas de umidades relativas, bem como os tamanhos médios dos símbolos para cada *Set* de dados sem compressão. Nesse caso, um dicionário Huffman fixo foi gerado com base nas estatísticas do *Set 7* e usado para comprimir os dados provenientes dos *Sets 8 e 9*. Os resultados obtidos a partir da utilização do esquema proposto para compressão e a partir do LEC são mostrados na Figura 8, como pode-se observar os valores obtidos pelo método proposto supera o LEC em todos os *Sets* de umidade. Aplicou-se também nos dados de umidade no algoritmo ALFC, com partições de 50, 100, e 400 amostras, assim como no bloco inteiro (Max). Verificou-se através dos resultados apresentados na Figura 9 que também o esquema proposto supera o ALFC nos três *Sets* de umidade.

As conclusões são similares às obtidas para os dados de temperatura. O esquema proposto tem uma melhor performance, com uma boa aproximação do limite teórico para os conjuntos de dados.

Tabela 11: Principais características dos *Sets* de dados de umidade relativa

Localização (Latitude , Longitude)	Nome	Range(%)	Amostras	Data (dd/mm/aa)	Tempo entre amostras
Morges, Suíça (46.494,6.472)	Set 7	44 a 95	15.362	06/08/07 a 02/09/07	2 min
Lausanne, Suíça (46.521,6.579)	Set 8	23 a 89	3.041	16/04/08 a 20/04/08	2 min
Bern, Suíça (46.948,7.444)	Set 9	25 a 91	4.851	13/03/07 a 15/03/07	0.5 min

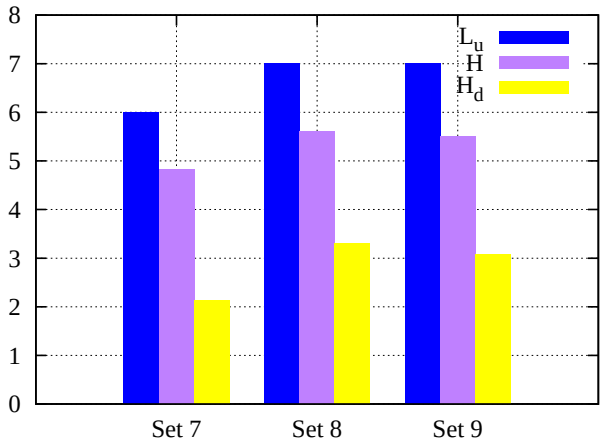

Figura 7: Tamanho médio dos símbolos (L_u) sem compressão, entropia das medições de umidade relativa (H) e entropia das diferenças consecutivas (H_d).

5.5 UMA PROPOSTA DE MELHORIA, O ESQUEMA DOS PARES.

Uma modificação que poderia ser implementada no método proposto na tentativa de melhorar a performance da compressão dos dados seria, por exemplo, comprimir não mais as diferenças consecutivas simples e sim um agrupamento em pares destas diferenças. Sendo assim, os dados a transmitir seriam $t_i = (d_i \quad d_{i-1})$, onde $d_i = m_i - m_{i-1}$ e $d_{i-1} = m_{i-1} - m_{i-2}$. Considerando as características das distribuições das diferenças consecutivas mostradas na Figura 3, os pares próximos da média zero tais como (0 0), (0 1), (1 0), (-1 0), (0 -1)... teriam maior probabilidade de ocorrência. Com o objetivo de testar esta nova abordagem, seguiu-se a metodologia utilizada no esquema proposto. A partir do *Set 1* de dados como referência, gerou-se um alfabeto Huffman fixo para comprimir os demais conjuntos de medidas de temperatura. No método proposto de diferenças simples foi gerada a Tabela 8 que contém 19 códigos representando as diferenças d_i desde - 10 até + 8. No caso dos pares, este alfabeto proveniente do *Set 1* ficou com 118 códigos representando as combinações existentes entre os pares das diferenças. Como no esquema proposto, também teríamos dois casos especiais a serem considerados: i) No início da coleta de dados, a primeira amostra m_0 será transmitida sem compressão já que não existe valor anterior para computar a primeira diferença d_0. A partir da terceira amostra coletada teremos o primeiro par a ser computado e comprimido; ii) O alfabeto fixo gerado a partir do *Set 1* possui uma quantidade limitada de símbolos, assim

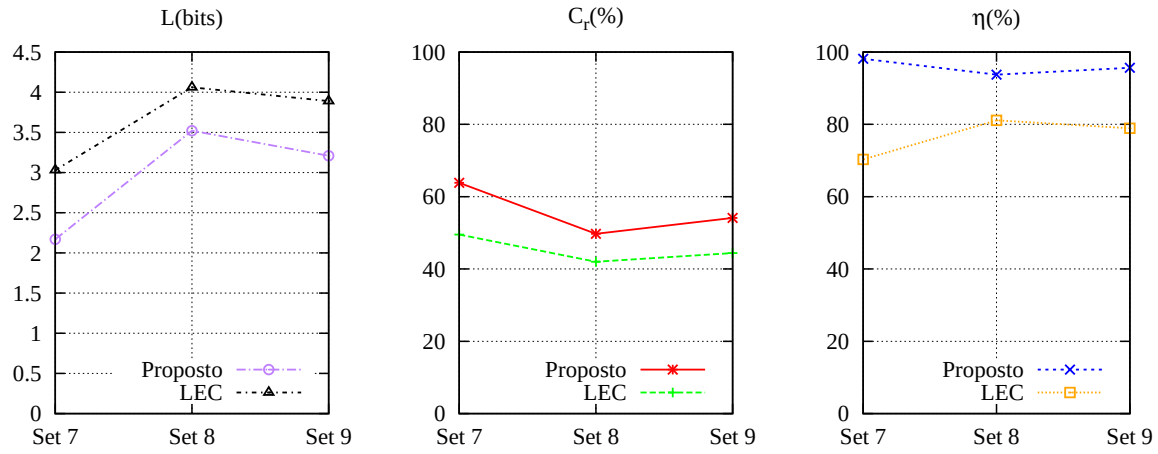

Figura 8: Tamanho médio do símbolo após compressão (L), taxa de compressão (C_r) para os conjuntos de dados de umidade relativa e eficiência do código (η).

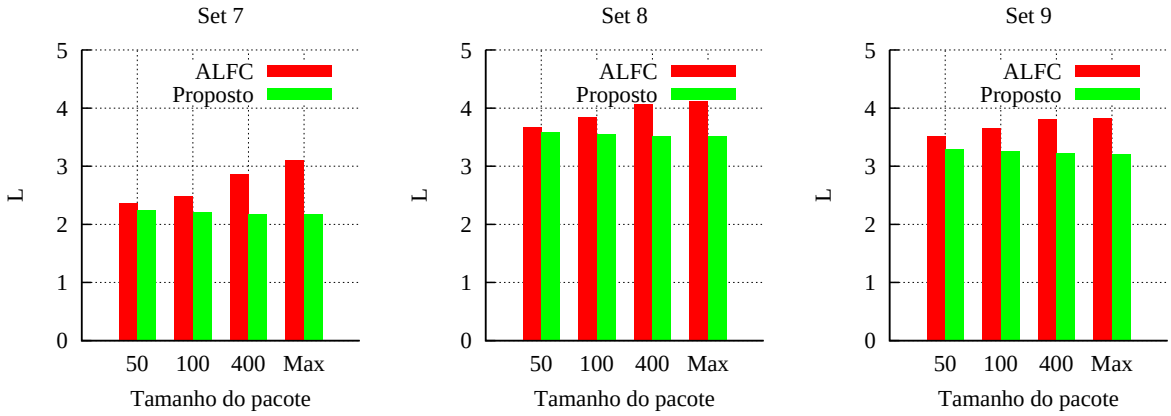

Figura 9: Tamanho médio do símbolo (L_u) (bits/símbolo) depois da compressão usando ALFC e o método proposto para tamanhos diferentes de pacotes

como no esquema proposto anteriormente. Quando ocorrer valores de pares não presentes no dicionário, estes dados seriam enviados sem compressão junto com um marcador especial.

Na Tabela 12 são apresentados os resultados obtidos aplicando nas bases de dados *Set 1* até *Set 6* o esquema de compressão dos pares das diferenças. Para o cálculo correto da taxa de compressão (C_r) do método dos pares deve-se levar em consideração que o tamanho do arquivo a ser processado é reduzido pela metade, assim deve-se dividir por 2 o valor do tamanho médio do símbolo depois da compressão em (24).

Verificou-se que as taxas de compressão (C_r) obtidas para o método proposto apresentado na Tabela 12 e para o caso que considera pares das diferenças são muito próximas, sendo que para os *Sets* 1,2 e 3 o esquema de pares supera o método proposto, e para os *Sets* 4, 5 e 6 ocorre o contrário. No entanto, foi constatado que para o método dos pares gerou-se um dicionário com 118 códigos, que tem valor muito superior comparado ao método proposto para

a mesma base geral *Set 1*, que gerou apenas 19 códigos. Pode-se concluir que apesar das taxas de compressão serem praticamente as mesmas, o esquema de pares gerou um dicionário com uma quantidade bem maior de códigos, o que necessita de maior quantidade de memória para armazenamento e maior tempo de busca no dicionário na descompressão dos dados. Pode-se concluir que o esquema de pares não proporciona ganho significativo em relação ao método de diferenças consecutivas simples proposto nesta dissertação.

Tabela 12: Tamanho médio do símbolo após a compressão (L) e taxa de compressão (C_r) para os diferentes conjuntos de temperatura no esquema dos pares e proposto

		Set 1	***Set 2***	***Set 3***	***Set 4***	***Set 5***	***Set 6***
Esquema dos pares	L	4,19	4,22	4,93	2,77	2,46	3,61
	C_r	65,08	47.25	58,91	72,30	75,40	63,90
Proposto	L	2,16	2,16	2,59	1,38	1,23	1,80
	C_r	64,00	46,00	56,83	72,40	75,40	64,00

6 CONSIDERAÇÕES FINAIS

Esta dissertação apresenta um mecanismo de compressão sem perda que utiliza um dicionário fixo de Huffman, com uma quantidade reduzida de palavras. O esquema proposto tem uma baixa complexidade computacional, o que poderá ser facilmente implementado em nós práticos de sensores de RSSFs. A fim de avaliar o método, foram calculadas as taxas de compressão obtidas de seis conjuntos de dados de temperatura coletados em diferentes localidades e durante períodos de tempo distintos, assim como em três conjuntos de dados de umidade. As taxas de compressão obtidas usando a abordagem proposta variaram entre $45,00\%$ a $75,40\%$ para temperatura e $48,85\%$ a $63,83\%$ para umidade. O esquema proposto, extremamente simples, superou o método LEC proposto em (MARCELLONI F. ; VECCHIO, 2008) para todos os conjuntos de dados considerados e também o método ALFC proposto em (KIELY et al., 2010) aplicado em vários blocos de dados de tamanho variável. Investigou-se também a performance do método usando diferentes *Sets* como referência, o que possibilitou concluirmos que a escolha da referência não é restrita a um determinado *Set*. Como trabalhos futuros pode-se sugerir a execução do algoritmo em redes de sensores reais, com o objetivo de se realizar uma implementação real para que se possa obter dados de campo, desenvolver algoritmo que leve em conta também a correlação espacial dos nós sensores, assim como avaliar os resultados com medições adquiridas por múltiplos sensores, avaliar a robustez do algoritmo considerando erros na transmissão como também sua eficiência energética.

REFERÊNCIAS

AKYILDIZ W. SU, Y. S. I.; CAYIRCI, E. A survey on sensor networks. **Communications Magazine, IEEE**, v. 40, p. 102–114, 2002.

ANASTASI G.;CONTI, M. F. M. P. A. Energy conservation in wireless sensor networks: A survey. **Ad Hoc Networks**, v. 7, p. 537–568, 2009.

AVR. 08 2012. Disponível em: <http://www.das-labor.org/wiki/AVR–Huffman/en>.

BARR K.C.; ASANOVIÉ, K. Energy-aware lossless data compression. **ACM Trans. Comput. Syst.**, v. 24, p. 250–291, 2006.

CAPO-CHICHI H. GUYENNET, J.-M. F. E. P. K-rle: A new data compression algorithm for wireless sensor network. In: . [S.l.: s.n.], 2009. p. 502–507.

(CCSDS), C. C. for S. D. S. **CCSDS RECOMMENDATION FOR LOSSLESS DATA COMPRESSION**. [S.l.], May 1997.

CULLER D. ;ESTRIN, D. . S. M. Guest editors' introduction: Overview of sensor networks. **Computer**, v. 37, p. 41–49, 2004.

GANESAN D.; CERPA, A. Y. W. Y.-Y. Z. J. E. D. Networking issues in wireless sensor networks. **Journal of Parralel an Distributed Computing**, v. 64, p. 799–814, 2004.

GERSHO, A. Adaptative filtering with binary reinforcement. **IEEE Transactions Information Theory**, v. 30, n. 2, p. 191–199, March 1984.

GOLOMB, S. Run-length encodings. **IEEE Transactions Information Theory**, v. 12, p. 399–401, 1966.

KIELY, A. et al. Adaptative linear filtering compression on realtime sensor networks. **The Computer Journal**, v. 53, n. 10, p. 1606–1620, 2010.

KIMURA N.; LATIFI, S. A survey on data compression in wireless sensor networks. **IEEE Computer Society**, p. 8–13, April 2005.

MARCELLONI F. ; VECCHIO, M. A simple algorithm for data compression in wireless sensor networks. **IEEE Communications Letters**, v. 12, p. 411–413, 2008.

MARCELLONI F. ; VECCHIO, M. An efficient lossless compression in wireless sensor networks. **The Computer Journal**, v. 52, p. 969–987, 2009.

MELLO, C. **Codificação livre de prefixo para cripto-compressão**. Tese (Doutorado) — PUC-RJ, 2006.

MHATRE V.; ROSENBERG, C. Design guidelines fo wireless sensor networks: communications, clustering and aggregation. **Ad Hoc Networks**, v. 2, p. 45–63, 2004.

PENNEBAKER W.; MITCHELL, J. **JPEG Still Image Data Compression Standard**. [S.l.]: Kluwer Academic Publishers, 1992.

POTTIE G.; KAISER, W. Wireless integrated network sensors. **Communications of ACM**, v. 43, n. 5, p. 51–58, 2000.

RAGHUNATHAN V. ; SCHURGHERS, C. . P.-S. . S. M. Energy-aware wireless microsensor networks. **IEEE Signal Processing Magazine**, p. 40–50, 2002.

REINHARDT A. ; CHRISTIN, D. . H.-M. . S. J. . M. P. . S. R. . S. J. . K. B. . B. F. Trimming the tree: Tailoring adaptive huffman coding to wireless sensor networks. **Wireless Sensor Networks**, v. 5970, p. 33–48, 2010.

REINHARDT M. HOLLICK, R. S. A. Stream-oriented lossless packet compression in wireless sensor networks. In: . [S.l.: s.n.], 2009.

RICE, R. F. Some practical universal noiseless coding techniques, part iii. **JPL Publication**, v. 83, p. 17, 1983.

SADLER, C. M.; MARTONOSI, M. Data compression algorithms for energy-constrained devices in delay tolerant networks. In: ACM (Ed.). [S.l.]: The 4th ACM Conference on Embedded Networked Sensor Systems, 2006. p. 265–278.

SALOMON, D. **Data Compression**. [S.l.]: Springer, 2004.

SAYOOD, K. **Introduction to Data Compression**. [S.l.]: Morgan Kaufman, 2000.

SCHOELLHAMMER T. ; GREENSTEIN, B. . W.-M. . E. D. . O. E. Lightweight temporal compression of microclimate datasets. In: SOCIETY, C. (Ed.). [S.l.]: 29th Annual IEEE Internacional Conference on Local Computer Networks, 2004. p. 516–524.

SENSORSCOPE. 08 2012. Disponível em: <http://sensorscope.epfl.ch>.

SRIDHARA, D. Efficient coding of information: Huffman coding. **Resonance**, v. 11, n. 2, p. 51–73, February 2006.

UNDERGROUND, W. 08 2012. Disponível em: <http://www.wunderground.com>.

VITTER, J. S. Design and analysis of dynamic huffman codes. **Journal of the ACM**, v. 34, p. 825–845, 1987.

WEINBERGER M. ; SEROUSSI, G. . S.-G. The loco-i lossless image compression algorithm: principles and stardardization into jpeg-ls. **IEEE Trans. Image Process**, v. 9, p. 1309–1324, 2000.

YICK J. ; MUKHERJEE, B. . G.-D. Wireless sensor network survey. **Computer Networks**, v. 52, p. 2292–2330, 2008.