Um Protocolo de Roteamento para Redes de Sensores Sem Fio Adaptável por Regras de Aplicação*

Daniel Fernandes Macedo, José Marcos S. Nogueira (orientador), Antonio Alfredo Ferreira Loureiro (co-orientador)

Departamento de Ciência da Computação
Universidade Federal de Minas Gerais
Av. Antônio Carlos 6627 - Prédio do ICEx - sala 4010 - Pampulha
CEP 31270-010 – Belo Horizonte, Minas Gerais

{damacedo,jmarcos,loureiro}@dcc.ufmg.br
http://www-rp.lip6.fr/~macedo/dissertacao.pdf

Abstract. *Wireless sensor networks are ad hoc networks with severe resource constraints. These constraints preclude the use of traditional ad hoc protocols, and demand optimizations that incur in solutions specific to a class of applications. This master's thesis presents PROC, a protocol that interacts with the application to stablish routes. This protocol allows the application to reconfigure PROC on runtime. The evaluation showed that PROC increases network lifetime around 7% to 12%, and increases the throughput when compared to other routing protocols found in the literature. Also, PROC presents a softer performance degradation when the number of nodes in the network increases. Finally, PROC was tested on real nodes.*

Resumo. *Redes de sensores sem fio são redes ad hoc que possuem restrições severas de recursos. Essas restrições impossibilitam o uso de protocolos ad hoc tradicionais e requerem otimizações, que implicam em soluções específicas para uma classe de aplicações. Esta dissertação apresenta um protocolo chamado PROC, que interage com a aplicação para estabelecer rotas. O protocolo permite que a aplicação reconfigure o roteamento em tempo de execução. Simulações mostram que o PROC, comparado a outros protocolos de roteamento, apresenta maior vazão e aumenta o tempo de vida da rede entre 7% e 12%. O PROC apresenta uma degradação de desempenho suave com o incremento de nós na rede. Finalmente, testamos o protocolo em nós reais.*

1. Introdução

Redes de Sensores Sem Fio (RSSF) são uma subclasse das redes ad hoc sem fio. As RSSF são formadas por elementos de rede chamados de nós sensores, que são dispositivos compactos compostos de sensores, processador, rádio para comunicação, memória e bateria. Esses nós enviam os dados coletados do ambiente para um Ponto de Acesso (PA), que repassa os dados ao usuário final [Akyildiz et al. 2002]. Diferentemente das redes ad hoc tradicionais, em geral não é possível recarregar a fonte de energia dos nós sensores devido à grande quantidade de nós ou às dificuldades impostas pelo ambiente.

*O presente trabalho foi realizado com apoio do CNPq, processo 55.2111/2002-3.

O consumo de energia é um fator crítico em RSSF, o que tem motivado a busca de novos protocolos, que procuram otimizar o consumo de energia. Uma forma frequente de otimização nestas redes é o uso de protocolos *cross-layer*, onde a separação de funções entre as camadas é violada. A aplicação pode, por exemplo, interagir com o protocolo de acesso ao meio para de-sincronizar o envio de dados, evitando colisões.

Em RSSF, o fluxo de dados segue um padrão. Nas *redes dirigidas a eventos*, o envio de dados ocorre somente quando um determinado evento é detectado. Exemplos são RSSF para localização de animais silvestres e detecção de intrusão, entre outros. Nas *redes de disseminação contínua*, os nós enviam mensagens em intervalos constantes para o PA. Exemplos incluem RSSFs para sistemas de tráfego inteligente e monitoração ambiental, entre outros. Devido às restrições de energia em RSSFs, os protocolos de roteamento geralmente satisfazem a apenas uma classe de rede. Assim, caso o desenvolvedor queira um alto desempenho em sua rede, este deve usar protocolos específicos para o cenário utilizado, ou novos protocolos devem ser projetados.

Neste trabalho é proposto um protocolo de roteamento chamado PROC (*Proactive ROuting with Coordination*), específico para redes de disseminação contínua de dados, que utiliza *interfaces* genéricas para que a aplicação personalize o seu funcionamento. A aplicação modifica as rotas construídas pelo protocolo, em tempo de execução, de acordo com as suas necessidades. O PROC ainda provê mecanismos que detectam rotas falhas e enlaces de baixa qualidade.

Comparamos via simulações o desempenho do PROC aos protocolos EAD [Boukerche et al. 2003] e o TinyOS Beaconing [Levis et al. 2004], sendo o último largamente utilizado em redes de sensores em operação. O PROC aumenta em até 12% o tempo de vida da rede, além de recuperar mais rapidamente de falhas de nós. Em seguida, o PROC foi implementado na plataforma Mica2, onde fizemos testes de pequena escala. O código das simulações e da implementação estão disponíveis na Internet[1].

2. O Protocolo PROC

O PROC é um protocolo pró-ativo desenvolvido para redes de disseminação contínua de dados. Esse protocolo considera aplicações onde os nós enviam dados ao ponto de acesso (PA) utilizando um caminho multi-saltos. Nessas aplicações, um nó comunica-se somente com seus nós vizinhos ou com o PA, quando este se encontra no seu raio de comuniçacão. Tendo em vista a escalabilidade, o PROC utiliza algoritmos com complexidade assintótica de $O(v)$ em consumo de memória, onde v é o número médio de nós ao alcance do rádio, e complexidade $O(1)$ em termos do número de mensagens enviadas.

O protocolo constrói uma estrutura de roteamento chamada de *backbone*, que é composta por um conjunto de nós, chamados *coordenadores*. Nós que não fazem parte do *backbone*, chamados de *nós folha*, comunicam-se diretamente com um nó coordenador. O *backbone* é uma árvore, que tem o PA como sua raiz. Assim, cada nó possui um *nó pai*, que repassa os dados para o PA. Esta estrutura possui duas vantagens. Primeiro, os nós folha armazenam apenas o endereço do nó pai no *backbone*, simplificando o roteamento. Segundo, a seleção dos nós do *backbone* é adaptável, permitindo assim que as aplicações adaptem as rotas às suas necessidades, como mostraremos a seguir. O *backbone* é reconstruído periodicamente, em intervalos de tempo chamados de *ciclos*.

[1]Endereço: `http://www-rp.lip6.fr/~macedo/`

2.1. A Construção do Backbone

A criação do *backbone* é um processo de duas fases. Inicialmente, os nós se auto-elegem coordenadores de acordo com as regras de aplicação. Cada nó possui uma probabilidade de tornar-se coordenador, que é definida pelas regras de aplicação a cada ciclo do roteamento. Esta primeira fase é chamada de "Eleição de coordenadores". Como na primeira fase os nós operam sem uma visão global da rede, o *backbone* formado pode estar incompleto. Assim, é necessária a segunda fase, chamada "Complementação do *backbone*", que adiciona mais nós ao *backbone* quando necessário. Devido a limitações de espaço, não mostraremos em detalhes os algoritmos utilizados.

2.2. Regras de Aplicação

As regras de aplicação proporcionam uma interface entre o *software* da aplicação e o protocolo de roteamento, permitindo que a aplicação modifique o comportamento do roteamento para que este reflita as suas necessidades. As regras modificam o conjunto de nós que repassam dados, assim a aplicação modifica as rotas utilizadas. As regras podem ser alteradas em tempo de execução, de acordo com mudanças no ambiente.

O *software* de aplicação pode utilizar qualquer informação disponível como entrada para o algoritmo que implementa as regras, tais como configuração dos nós e de outros protocolos, estado atual da rede, ou mesmo dados de sensores e atuadores. As regras podem ser altamente especializadas, podendo até se aproveitar de características de uma instância específica de uma aplicação. Fica a cargo do projetista definir quais serão as características a serem consideradas na regra da aplicação. O PROC permite que o *software* de aplicação envie dados embutidos nos pacotes de roteamento, que podem ser utilizados para trocar informações úteis ao cálculo das regras.

Caso nenhuma regra seja definida pela aplicação, o PROC utiliza regras próprias, que procuram maximizar o tempo de vida da rede ao homogeneizar o consumo de energia e distribuir o tráfego entre os nós. O algoritmo de roteamento do PROC ainda garante o estabelecimento de rotas mesmo que a aplicação implemente erradamente as regras.

2.3. Tolerância a Falhas

O PROC implementa dois mecanismos de tolerância a falhas, que são a reconstrução periódica de rotas e a monitoração ativa do nó pai.

Reconstrução periódica de rotas: As rotas são completamente reconstruídas ao início de cada ciclo, utilizando somente os nós ativos. O intervalo entre cada recriação de rotas deve ser ajustado de acordo com o grau de tolerância a falhas e o consumo de energia desejados. Por ser um processo que envolve o envio de mensagens para toda a rede, a recriação de rotas envia um grande número de mensagens. Assim, este processo deveria ocorrer com a menor frequência possível.

Monitoração ativa do nó pai: O PROC utiliza um mecanismo de monitoração de atividade para detectar a falha de nós ou enlaces de baixa qualidade. Este mecanismo aumenta a tolerância a falhas entre as reconstruções periódicas das rotas, permitindo assim que os ciclos sejam mais longos. Nesse mecanismo, os quadros de confirmação (ACK) do protocolo MAC são utilizados para monitorar o funcionamento do nó pai. Quando um número de ACKs consecutivos não recebidos (p) ultrapassa um valor limite, o protocolo

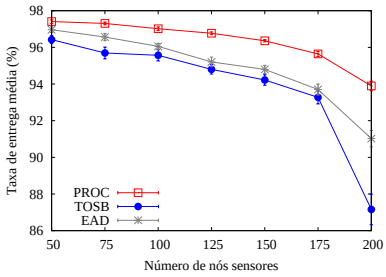

Figura 1. Taxa de entrega média.

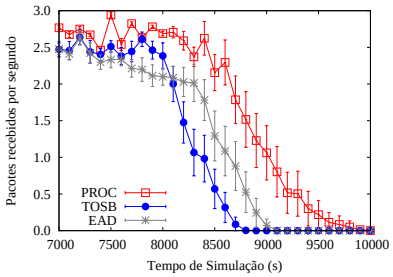

Figura 2. Vazão média ao fim do tempo de vida da rede.

considera que o nó pai está falho, retirando-o da lista de vizinhos e recalculando as rotas. Propomos uma fórmula que calcula o valor de p a partir do grau de certeza esperado do mecanismo de tolerância a falhas e da qualidade do canal, medida pela taxa média de erro de bits do rádio.

3. Avaliação

A avaliação do protocolo foi feita por simulações no simulador NS-2 e por experimentação na plataforma Mica2. As simulações utilizaram o simulador NS-2 pois este é extensivamente usado na literatura. Devido às limitações do simulador ao início do trabalho, realizamos modificações nos módulos MAC e físico, para que as suas características se aproximassem às dos nós sensores Mica2 [Levis et al. 2004].

Simulamos uma aplicação multi-saltos de coleta ambiental, com as características de tráfego similares a uma rede real utilizada para estudos do ecosistema e comportamento de aves [Szewczyk et al. 2004]. Cada sensor envia mensagens de dados de 36 bytes para o PA, em períodos regulares de 70s. As topologias simuladas consistiam de nós estacionários em posições aleatórias seguindo uma distribuição uniforme. Apresentamos a média de 33 simulações com sementes aleatórias e intervalo de confiança de 95%.

A Figura 1 mostra que o PROC possui uma taxa de entrega média de 3% a 6% superior aos outros protocolos avaliados [Macedo et al. 2006]. Ainda, o PROC aumenta o tempo de vida da rede em 12,5% e 7,6% em comparação ao TOSB e ao EAD, respectivamente (Figura 2). Além disto, o PROC apresentou maior vazão durante toda a simulação. A análise de complexidade mostrou que o número de mensagens requeridas para a construção de rotas independe de densidade ou tamanho da rede e que o consumo de memória é proporcional ao número de nós alcançados diretamente, o que torna o protocolo escalável.

Devido à frequência da ocorrência de falhas em RSSF, avaliamos o desempenho do PROC em comparação aos protocolos EAD e TinyOS Beaconing em situações de falhas de nós. Para tornar a avaliação mais fidedigna, realizamos um estudo dos agentes causadores de falhas silenciosas, incluindo ataques de segurança e classificamos as falhas encontradas de acordo com duas características. A *extensão*, que indica quantos nós falham ao mesmo tempo, e a *persistência*, que determina o tempo médio de falha [Macedo et al. 2005]. Este modelo reduziu o número de casos a serem avaliados a quatro, simplificando a avaliação de tolerância a falhas.

Verificamos que a monitoração do nó pai permite que o PROC se recupere mais

rapidamente de falhas curtas que envolvem poucos nós. Este mecanismo ainda maiores intervalos de reconstrução de rotas, diminuindo o consumo de energia. Ainda, o PROC se recupera das falhas em um intervalo inferior ao intervalo de recriação de rotas.

Por fim, implementamos o PROC sobre o TinyOS, e testamos a implementação sobre a plataforma Mica2. A plataforma Mica2 e o sistema TinyOS são amplamente utilizados na pesquisa em RSSF. O TinyOS, por sua vez, é o SO padrão *de facto* nas RSSF. O cenário de teste consistiu de cinco nós, sendo que um deles opera como PA, reconstruindo as rotas a cada dois segundos. Este nó é ligado a um computador pessoal, que registra os pacotes recebidos pelo PA. Os quatro nós restantes transmitem dados para o PA. O cenário avaliado possui dimensões reduzidas, devido às limitações de recursos e tempo. Assim, realizamos apenas testes para validar a implementação do protocolo.

4. Conclusões

Esta dissertação apresentou um protocolo de roteamento pró-ativo, denominado PROC. O protocolo é otimizado para RSSF de disseminação contínua de dados, que são redes onde os nós sensores enviam dados para o ponto de acesso em intervalos regulares de tempo. O PROC é o primeiro protocolo de roteamento para redes de disseminação contínua de dados que interage com a aplicação tendo em vista a otimização do roteamento. Esta interação é feita pelas *regras de aplicação*, que permitem que o PROC se adapte em tempo de execução a mudanças no ambiente.

Experimentos e simulações mostraram que o PROC é um algoritmo escalável, que aumenta em até 12% o tempo de vida da rede em comparação a protocolos de roteamento existentes na literatura. Além disso, propusemos um modelo de falhas que simplifica o estudo de tolerância a falhas para protocolos de roteamento em RSSF. O código das simulações e a implementação na plataforma Mica2 estão disponíveis na Internet.

Referências

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. (2002). A Survey on Sensor Networks. *IEEE Communications*, 40(8):102–114.

Boukerche, A., Cheng, X., and Linus, J. (2003). Energy-aware data-centric routing in microsensor networks. In *Proceedings of the 6th international workshop on Modeling analysis and simulation of wireless and mobile systems*, pages 42–49.

Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., and Culler, D. (2004). TinyOS: An operating system for wireless sensor networks. In *Ambient Intelligence*. Springer-Verlag.

Macedo, D. F., Correia, L. H. A., dos Santos, A. L., Loureiro, A. A., Nogueira, J. M., and Pujolle, G. (2005). Evaluating Fault Tolerance Aspects in Routing Protocols for Wireless Sensor Networks. In *Fourth Annual Mediterranean Ad Hoc Networking Workshop*.

Macedo, D. F., Correia, L. H. A., dos Santos, A. L., Loureiro, A. A. F., and Nogueira, J. M. (2006). A rule-based adaptive routing protocol for continuous data dissemination in WSNs. *Journal of Parallel and Distributed Computing (JPDC)*, 66(4):542–555.

Szewczyk, R., Polastre, J., Mainwaring, A., and Culler, D. (2004). Lessons from a sensor network expedition. In *Proceedings of the First European Workshop on Sensor Networks (EWSN)*, pages 307–322.