Protocolo de Transporte Colaborativo para Redes de Sensores sem Fio

Eugênia Giancoli[1,2], **Filippe C. Jabour**[1,2], **Aloysio de Castro Pinto Pedroza**[1]

[1]Programa de Engenharia Elétrica – Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brasil

[2]Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG)
Leopoldina – MG – Brasil

Abstract. *This work presents the Collaborative Transport Control Protocol (CTCP), a new transport protocol for sensor networks. It provides end-to-end reliability and adapts to different type of applications through its mechanism of reliability variation. Its congestion control and detection differentiates communication losses of buffer overflow. It's called collaborative since all nodes detect and act on congestion control and its distributed storage responsibility. It is scalable and independent of the underlying network layer. It was observed that the distributed fault recovery increases the reliability and that the duplication of storage responsibility minimizes the possibility of definitive loss of messages.*

Resumo. *Este trabalho apresenta o CTCP, um novo protocolo de transporte para redes de sensores. Ele provê confiabilidade fim-a-fim e se adapta aos diferentes tipos de aplicação através do mecanismo de variação desta confiabilidade. Seu controle e detecção de congestionamento diferencia as perdas relativas a erro de transmissão daquelas relativas ao esgotamento de buffers. É chamado colaborativo já que todos os nós detectam e atuam sobre o congestionamento e em função da responsabilidade distribuída de armazenamento. É escalável e independe das camadas de rede subjacentes. Observou-se que a recuperação distribuída de falhas aumenta a confiabilidade e que a duplicação desta responsabilidade minimiza a possibilidade de perda definitiva de mensagens.*

1. Introdução

As redes de sensores sem fio (RSSF) fornecem uma solução de sensoriamento amplamente distribuída e econômica para ambientes onde as redes tradicionais não conseguem atuar. Suas principais aplicações estão em monitoramento militar, ambiental, médica, industrial ou infraestruturas domésticas.

Atualmente, o hardware das redes de sensores é escolhido visando maximizar a vida útil da rede para uma aplicação específica. Contudo, acreditamos que o número de aplicações para redes de sensores crescerá e existirá a necessidade de hardwares mais poderosos e genéricos que poderão ser reprogramados durante sua vida útil. Esta reprogramação, é uma das aplicações que requer um protocolo de transporte que garanta confiabilidade na entrega dos dados.

Portanto, cada aplicação possui diferentes características e requisitos de tipos de dados, taxas de transmissão e confiabilidade. A maioria dos protocolos existentes para a camada de transporte das redes de sensores foram adaptados para funcionar com determinados tipos de aplicações, ou assumem que os nós trabalham com determinadas camadas de rede ou enlace. Como resultado, estes protocolos não podem ser aplicados em qualquer tipo de rede de sensores.

Assim, é necessário projetar um protocolo da camada de transporte que possa suportar aplicações múltiplas na mesma rede, provendo controle de confiabilidade variável, controle de congestionamento, redução de perdas e suporte a desconexões freqüentes [Kim et al. 2007]. Por isso, este trabalho explora as decisões de projeto de um protocolo de transporte para suportar uma classe de aplicações que requer entrega de dados confiável nas redes de sensores sem fio.

De acordo com [Iyer et al. 2005], os requisitos básicos para uma camada de transporte genérica para redes de sensores são:

Heterogeneidade: os nós sensores podem possuir múltiplos sensores (luz, temperatura, presença, etc) com características diferentes de transmissão. Os pacotes gerados por cada sensor para uma aplicação constitui seu fluxo de dados, que pode ser contínuo ou baseado em eventos. Nas aplicações de fluxos contínuos, os nós transmitem pacotes periodicamente para a estação base. Nas aplicações baseadas em evento, os nós transmitirão dados somente quando um determinado evento ocorrer. Os dois tipos de fluxo podem existir na mesma rede e o protocolo da camada de transporte deve suportar múltiplas aplicações heterogêneas dentro da mesma rede.

Confiabilidade: as aplicações têm requisitos diferentes de confiabilidade. Por exemplo, em ambiente militar, os dados transmitidos pelos sensores devem chegar sempre à estação base. Na reprogramação de grupo de sensores, os dados também precisam chegar até todos os nós. Entretanto, no monitoramento de temperatura, alguns pacotes podem ser perdidos sem causar grandes danos. O protocolo de transporte deve explorar esta diferença visando economizar energia.

Controle de congestionamento: todos os nós da rede geram pacotes que convergem para os nós localizados ao redor da estação base. Este nós encaminham mais pacotes e consequentemente, aumenta a possibilidade de congestionamento perto da estação base. Altas taxas de dados, rajadas de dados e colisões são as outras razões para os congestionamentos nas redes de sensores.

O protocolo apresentado por este trabalho, chamado Collaborative Transport Control Protocol (CTCP), fornece um transporte confiável, escalável e genérico às RSSF onde cada nó pode ser a origem de vários fluxos com diferentes características. Além disso, suporta redes com aplicações múltiplas e provê funcionalidades adicionais como detecção e controle de congestionamento e variação da confiabilidade.

Na seção 2 serão citados os trabalhos relacionados, na seção 3 o CTCP será descrito, especificado e modelado probabilisticamente. Na seção 4 finalizamos com as conclusões e trabalhos futuros.

2. Trabalhos relacionados

O primeiro protocolo analisado foi o TCP[Allman et al. 1999]. Este protocolo faz parte da pilha de protocolos TCP/IP que foi teoricamente projetada para operar de forma independente das tecnologias das camadas inferiores. Assim, o perfil de protocolos TCP/IP deve operar em redes cabeadas confiáveis, redes sem fio, redes de satélite, redes ópticas etc. No entanto, os atuais mecanismos do TCP se baseiam em suposições típicas de redes cabeadas convencionais, tais como a existência de uma conectividade fim-a-fim entre fonte e destino durante todo o período correspondente à sessão de comunicação, atrasos de comunicação relativamente pequenos (na ordem de milissegundos), baixas taxas de erros, mecanismos de retransmissão efetivos para reparar erros e suporte a taxas de dados bidirecionais relativamente simétricas.

Desta forma, o TCP não se adequa às redes de sensores. Estas redes são caracterizadas por atrasos longos ou variáveis, quebra freqüente de conexões, conectividade intermitente, altas taxas de erro e limitação de recursos. A pilha TCP/IP apresenta um baixo desempenho nestas redes.

Reliable Multi-Segment Transport (RMST) [Stann and Heidemann 2003] foi projetado para funcionar em conjunto com o Direct Diffusion Protocol [C. Intanagonwiwat 2000], ou seja, existe uma dependência da camada de rede. O RMST é um protocolo baseado em NACK e trabalha com ou sem cache. Quando o cache está habilitado, os nós intermediários armazenam os fragmentos de dados, o que pode causar esgotamentos dos *buffers*. RMST não garante a confiabilidade quando um nó falha depois de receber e antes de transmitir os fragmentos de dados. Além disso, o RMST não trata o congestionamento de dados na rede de sensores.

Event-to-Sink Reliable Transport (ESRT)[Sankarasubramaniam et al. 2003] foi projetado para redes centradas em aplicações. É pressuposto que a estação base está interessada em um determinado evento que pode ser detectado por vários nós ao mesmo tempo. No ESRT, é a estação base que faz o controle do congestionamento, socilitanto aos nós o aumento ou diminuição da taxa de transmissão. Não garante entrega fim-a-fim. Em redes reais os nós só transmitem dados quando detectam um evento, o que dificulta o controle da taxa de transmissão pela estação base.

Pump Slowly, Fetch Quick (PSFQ)[Wan et al. 2005] tem como objetivo reprogramar os nós de uma rede de sensores. Possibilita um broadcast de dados confiável da estação base para os nós. Contudo, possui alto consumo de energia, uma vez que a confiabilidade é alcançada através do aumento de retransmissões na rede.

Sensor Transmission Control Protocol (STCP)[Iyer et al. 2005] foi projetado para ser um protocolo genérico. Entende-se por genérico a capacidade de se adaptar a qualquer tipo de aplicação e trabalhar com qualquer camada de rede subjacente. Possui controle de congestionamento, uma vez que os nós ao redor da estação base podem estar sujeitos a esgotamento de *buffers*. Possui nível de confiabilidade adaptável visando economia de energia. Contudo, quase todos os seus controles baseiam-se fortemente na estação base, que possui amplos poderes computacionais e energéticos. Tal comportamento se desvia do desafio maior das redes de sensores que é o auto-gerenciamento. Considera-se questionável, ainda, que o nível de confiabilidade requerido pela aplicação seja controlado pelo nó que origina a informação, pois, o conhecimento global da rede e da aplicação seria

necessário para tomar tal decisão. O sincronismo de tempo da rede de sensor é requerido para economizar energia em aplicações que possuam fluxos de dados contínuos. Contudo, não existem resultados que provem que o overhead causado pela sincronização justifique esta escolha.

Neste trabalho apresentamos o CTCP. Trata-se de um protocolo colaborativo de transporte baseado em mecanismos conhecidos [Allman et al. 1999] de reconhecimento de pacotes (ACK) e temporização. Nosso trabalho utiliza o reconhecimento salto-a-salto, demonstrado eficiente em [Stann and Heidemann 2003] mas prevê a liberação imediata do cache (*buffers*) do nó, aumentando a sua capacidade de reencaminhamento e evitando o congestionamento. O CTCP dispensa ainda a sincronização da rede, utilizada em [Iyer et al. 2005] e é capaz de suportar quebra das conexões sem perda de dados. Mesmo quando um nó recebe os dados e falha antes de reencaminhá-los, o protocolo é capaz de recuperar-se desta perda. Foi projetado para trabalhar com qualquer tipo de camada subjacente e possui um controle de congestionamento capaz de evitar perdas relativas a esgotamento de *buffers*. Os dois níveis de confiabilidade garantem ao CTCP flexibilidade para se adaptar aos diferentes tipos de aplicações.

3. Especificação do protocolo

As principais contribuições do CTCP são:

- Garantir que todos os segmentos sejam entregues, à camada de aplicação da estação base, mesmo na presença de falhas de nós e freqüentes desconexões
- Adaptação ao perfil de confiabilidade da aplicação visando economizar energia
- Diferenciar a perda relativa a congestionamento da perda relativa a erro de transmissão
- Controlar o congestionamento através da interrupção/liberação imediata do encaminhamento
- Independência das camadas subjacentes

Para a modelagem e análise formal deste protocolo, foram empregados dois métodos, um formalismo matemático denominado Redes de Petri Predicado Ação [Nielsen et al. 1981] e uma linguagem de especificação chamada CCS (*A Calculus of Communicating Systems*), de Robin Milner [Milner 1980]. São métodos de especificação formal que permitem o desenvolvimento de sistemas com um mínimo de ambigüidades, através de uma sintaxe e semântica bem definidas. A especificação formal do nosso protocolo possibilita uma análise de comportamento funcional de certas propriedades, como ausência de bloqueios(*deadlocks*), sincronismo e seqüência correta de mensagem, além da verificação da consistência da especificação.

3.1. Abertura e encerramento da conexão fim-a-fim

Antes de iniciar a transmissão de dados, um pacote de abertura de conexão (ABR) é enviado da origem para a estação base. Este pacote informa à estação base o identificador do fluxo de dados e o primeiro número de seqüência. Quando a estação base recebe este pacote, ela reserva os *buffers*, inicializa as variáveis necessárias e envia um reconhecimento (ACK) à origem da conexão. No cabeçalho do ACK são especificados o nível de confiabilidade requerido pela aplicação à qual o fluxo pertence e o identificador da conexão (ID), que é controlado pela estação base para evitar que existam conexões diferentes com

o mesmo ID. Controlar o identificador da conexão provê flexibilidade ao protocolo, uma vez que, este dado será necessário para implementar a multiplexação de diferentes fluxos da rede. Esta implementação será tratada em trabalhos futuros.

Logo a seguir, o nó origem estará apto a iniciar a transmissão de dados para a estação base. Para que este protocolo fosse o mais genérico possível, durante seu projeto, preocupou-se em estabelecer formatos de pacotes compatíveis com a maioria das redes subjacentes. Desta forma, foi preciso considerar que a comunicação sem fio, e principalmente as redes de sensores, possuem dificuldades de transferência de pacotes que sejam maiores do que o quadro da camada de enlace. Apesar de alguns protocolos, como o 802.11 [Vassis et al. 2005], possuirem fragmentação e agrupamento, existem limites no tamanho dos pacotes que uma entidade consegue fragmentar e garantir a entrega. [Stann and Heidemann 2003]. Por isso, os sensores que utilizarão o CTCP serão pré-configurados com o MSS (Maximum Segment Size) permitido pelas camadas de rede subjacente. Tal variável não precisa ser negociada durante a abertura da conexão.

Neste protocolo, ao contrário do STCP, a confiabilidade não é definida pelo nó origem, uma vez que o próprio nó não possui ”inteligência“ suficiente para identificar o tipo de confiabilidade requerida por cada aplicação.

Esta conexão fim-a-fim não expira por tempo nem por ausência de tráfego podendo atender a requisitos de desconexões por longos períodos de tempo sem perda de dados. A figura 1 ilustra a abertura e o fechamento das conexões.

Quando a camada de aplicação do nó origem termina seu trabalho envia um pacote de fechamento de conexão à estação base. A estação base, então, libera os *buffers* e variáveis da conexão especificada e responde com um ACK. A origem aguarda o recebimento do ACK para efetuar a interrupção da conexão. Resumidamente, o fechamento da conexão consiste em informar à estação base que a origem terminou de gerar dados e não vai transmitir mais nenhum pacote, ou se o desejar fazer abrirá uma nova conexão.

3.1.1. Abertura e Encerramento de conexões no CTCP descritas em CCS

$$Abertura\_Encerramento = Origem || Base$$

$$Origem = !abr.Esp\_ack$$
$$Esp\_ack = ?ack.T\_dados\_or$$
$$T\_Dados\_or = !disc.Esp\_ack\_final$$
$$Esp\_ack\_final = ?ack.Origem$$
$$Base = ?abr.!ack.T\_Dados\_Base$$
$$T\_Dados\_Base = ?disc.!ack.Base$$

3.2. Controle da variação da confiabilidade

Cada aplicação possui requisitos diferentes de confiabilidade. Algumas, por exemplo, suportam perdas de dados enquanto outras precisam de banda passante mínima. Desta forma, para especificar a confiabilidade requerida é preciso que se tenha conhecimento da aplicação e seus objetivos. Ao contrário do que sugere o trabalho [Iyer et al. 2005], nesta proposta, esta decisão não será tomada pelos nós, uma vez que eles não possuem uma visão global da aplicação.

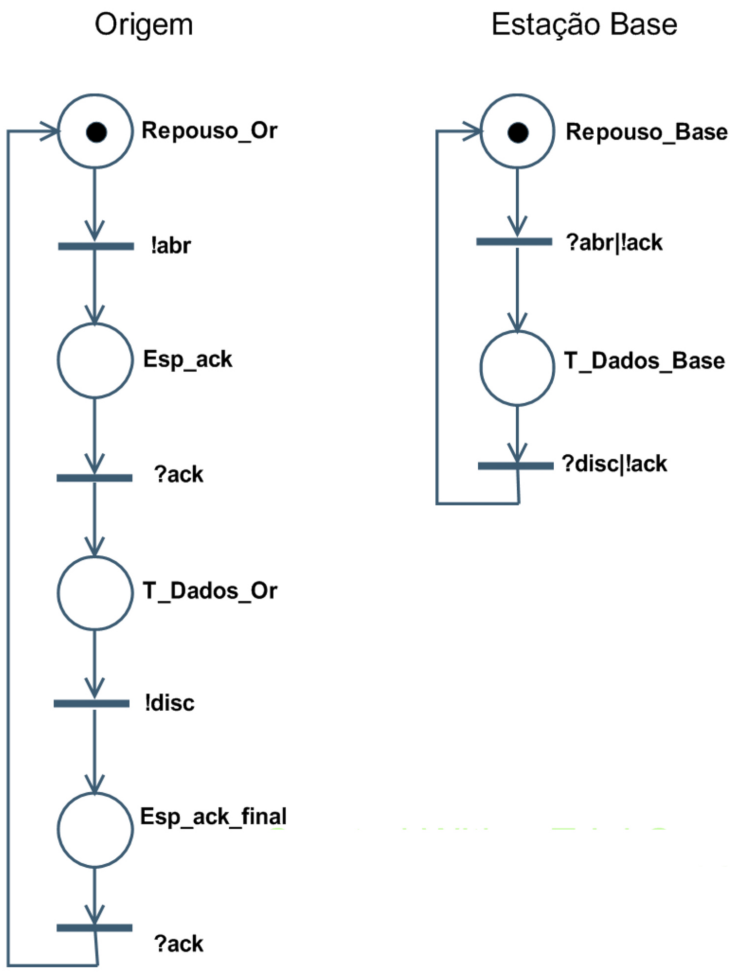

Figura 1. Abertura e Fechamento das conexões

Neste trabalho, a estação base, é responsável por estipular o nível de confiabilidade requerido pela aplicação. Além disso, o nível de confiabilidade, que é definido na abertura da conexão, pode ser alterado dinamicamente sempre que houver necessidade. Esta necessidade pode ser representada, por exemplo, pelo esgotamento de energia dos nós. Nessa situação, pode ser mais interessante trabalhar com menos confiabilidade para maximizar a vida útil da rede.

A confiabilidade será medida pela fração de pacotes recebidos pela estação base com sucesso.

É preciso ressaltar que cada pacote pode, ou não, tomar diferentes caminhos para chegar até o destino. Isto depende do algoritmo de roteamento e é tarefa da camada de rede. Este trabalho parte do pressuposto que a rota já foi definida pela camada de rede e que há reconfiguração de rotas no nível de rede em caso de falha ou desligamento dos nós.

Uma vez definido o nível de confiabilidade requerido pela aplicação, os nós da rede poderão agir de duas maneiras diferentes descritas nas seções 3.2.1 e 3.2.3.

3.2.1. Nível 1 de confiabilidade

Este nível de confiabilidade visa economia de energia, através da redução de transmissões, possui baixo custo de *buffers* e aplica-se principalmente a aplicações que possuem alguma redundância de dados ou que possam tolerar perdas.

Depois de receber um pacote de um nó A, o nó B envia ao nó A um reconhecimento (ACK) e passa a ser temporariamente responsável pela entrega do pacote à estação base. Este processo acontecerá repetidamente, através da rota estipulada pela camada de rede, até que a estação base receba o pacote de dados, e envie um ACK ao nó imediatamente anterior. Qualquer um dos nós, ao receber um ACK poderá descartar o pacote enviado, poupando espaço em seus *buffers* conhecidamente reduzidos. Esta situação está representada graficamente na figura 2 e formalmente, através das Redes de Petri na figura 3

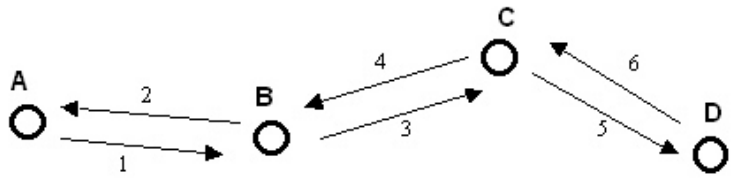

Figura 2. Ordem da troca de mensagens do CTCP para nível 1 de confiabilidade

Repare que, ao assumir a obrigação de entregar o pacote, o nó intermediário deverá manter uma cópia deste pacote em buffer até receber o ACK. A ausência de recebimento do ACK gera um esgotamento de temporização do nó origem e a retransmissão do pacote não reconhecido.

Considere, entretanto, a seguinte situação: o nó A, origem, envia dados ao nó B. Este, por sua vez, recebe os dados, armazena e envia o ACK ao nó A. Neste exato momento o nó B falha e deixa de fazer parte da rede. Logo, os dados que estavam sob responsabilidade do nó B não serão entregues à estação base e o nó A não perceberá esta falha até que a conexão seja fechada. Esta situação pode ser resolvida através do aumento do nível de confiabilidade.

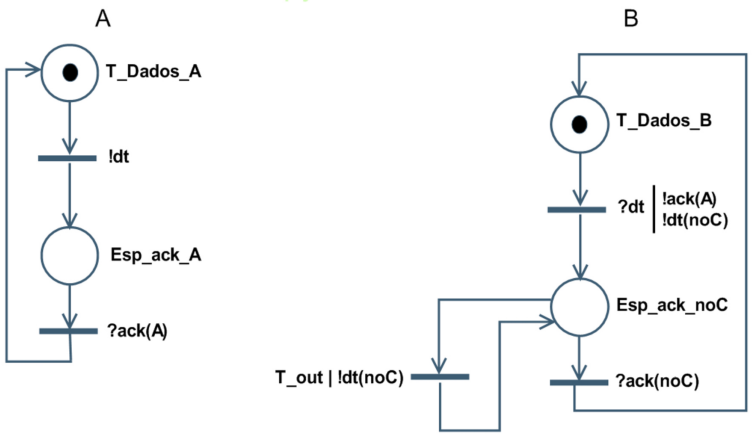

Figura 3. Comunicação entre dois nós da rede

Para escolher entre os dois níveis de confiabilidade, o usuário que interage com a estação base, deverá considerar as necessidades da aplicação.

3.2.2. Confiabilidade Nível 1 descrita em CCS

$Nivel\_1 = A|||B|||Timer$

$A = !dt.Esp\_ack\_A$
$Esp\_ack\_A = ?ack(A).A$
$B = ?dt.!ack.!dt(noC).!starttimer.Esp\_ack\_noC$
$Esp\_ack\_noC = ?ack(noC).!reset\_timer.B + ?t\_out.!dt(noC).!start\_timer.Esp\_ack\_noC$
$Timer = ?start\_timer.(!t\_out.Timer + ?reset\_timer.Timer)$

3.2.3. Nível 2 de Confiabilidade

O nó A envia os dados para o nó B e espera receber o duplo ACK. O duplo ACK é gerado da seguinte maneira: B recebe os dados de A e devolve para A o primeiro ACK. O nó B envia os dados para C que devolve para B o primeiro ACK. Quando B receber o primeiro ACK de C enviará para A o segundo ACK. Somente neste momento A descartará os dados mantidos em buffer. Todos os nós repetirão este processo, sucessivamente, até que os dados cheguem à estação base. A figura 4 representa graficamente esta troca de mensagens.

Se o nó B falhar antes de entregar os dados ao nó C, o nó A não receberá o segundo ACK e retransmitirá o pacote. Observe que partimos do pressuposto de que as falhas dos nós são monitoradas pelo algoritmo de roteamento e este será responsável por refazer a rota quando da falha de um determinado nó, ou conjunto deles.

O protocolo descrito acima possibilita que a mensagem chegue ao seu destino com uma probabilidade maior, uma vez que a falha de um nó no caminho não interrompe a entrega dos dados. A especificação formal do nível 2 é feita de forma análoga à do nível 1.

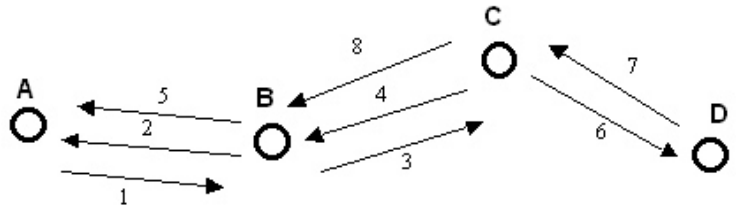

Figura 4. Ordem da troca de mensagens do CTCP para nível 2 de confiabilidade

3.3. Avaliação probabilística da confiabilidade

As redes tradicionais garantem confiabilidade na camada de transporte através de mecanismos de recuperação de erros fim-a-fim. Esta abordagem não é ideal para as redes de sensores pois ao contrário das redes tradicionais onde os nós intermediários são apenas

roteadores, nas redes de sensores eles possuem a camada de transporte o que nos permite distribuir a tarefa de recuperação de erros. Esta colaboração entre os nós torna-se factível porque todos os nós pertencem à mesma entidade administrativa e desejam atingir um mesmo objetivo.

Além de todas as diferenças no modelo de comunicação e serviços das redes de sensores, o maior problema com a recuperação de erros fim-a-fim é a baixa qualidade dos enlaces, pois, os sensores, normalmente trabalham com rádios de baixo alcance, em ambientes com muitos obstáculos e usam técnicas de encaminhamento de mensagens de múltiplos saltos. Desta forma, os erros se acumulam exponencialmente através dos múltiplos saltos.

3.3.1. Probabilidade de entrega fim-a-fim de uma mensagem

Considere que as chances de trocar uma mensagem com sucesso através de um único salto seja p. Assim, a taxa de erros no canal de comunicação é dada por $(1 - p)$. Seja q a probabilidade de sucesso no envio de um ACK por um único salto. Seja R o número máximo de retransmissões que podem ser feitas no nível de transporte.

Observe que a utilização de um único ACK fim-a-fim para a recuperação de uma eventual perda, estará sujeita às vulnerabilidades acumuladas de toda a rota de ida e volta. Assim, para o sucesso na transmissão fim-a-fim, a mensagem terá que atravessar os h saltos da origem (nó fonte) ao destino (nó sorvedouro) e o ACK terá que atravessar os mesmos h saltos de volta (supondo que não houve alteração de rota). Em caso de falha, qualquer uma das R retransmissões (novas tentativas) ocorrerão mais uma vez fim-a-fim. Assim, a probabilidade de sucesso na entrega de uma mensagem, encapsulada em um único pacote, à estação base, h saltos distante do nó origem, será:

p^h : sucesso em todo o caminho de ida (MSG)

q^h : sucesso em todo o caminho de volta (ACK)

$p^h\, q^h$: sucesso ida e volta

$P(f) = (1 - p^h\, q^h)$: probabilidade de alguma falha ao longo de todo o trajeto

$P(f)_R = (1 - p^h\, q^h)^{R+1}$: probabilidade de falha em todas as transmissões

Assim, a equação 1 nos dá a probabilidade de sucesso com confirmação (ACK) fim-a-fim e com até R retransmissões

$$P(s)_{ACK\ fim-a-fim} = 1 - (1 - p^h\, q^h)^{R+1} \tag{1}$$

Conforme descrito na seção 3.2, o protocolo proposto propõe a recuperação de falhas feita salto-a-salto, com ACKs enviados por cada vizinho ao nó anterior. Deste

modo, a análise probabilística passa a ser a seguinte:

Probabilidade de sucesso $P(s)$ para um único salto:

$$P(s) = \sum_{i=0}^{R-1} pq\,(1-pq)^i \tag{2}$$

A expressão correspondente, para a probabilidade de não falhar nas R retransmissões, é:

$$P(s) = 1 - (1-pq)^R \tag{3}$$

Assim, a probabilidade de sucesso na entrega de uma mensagem, encapsulada em um único pacote, à estação base, h saltos distante do nó origem, será:

$$P(s)_{ACK\ salto-a-salto} = (P(s))^h \tag{4}$$

Observamos pelas equações 1 e 4 e pela figura 5, que a recuperação salto-a-salto oferece uma garantia de entrega muito maior que o esquema com recuperação fim-a-fim. As equações foram resolvidas para $R = 3$ baseado em [Stann and Heidemann 2003].

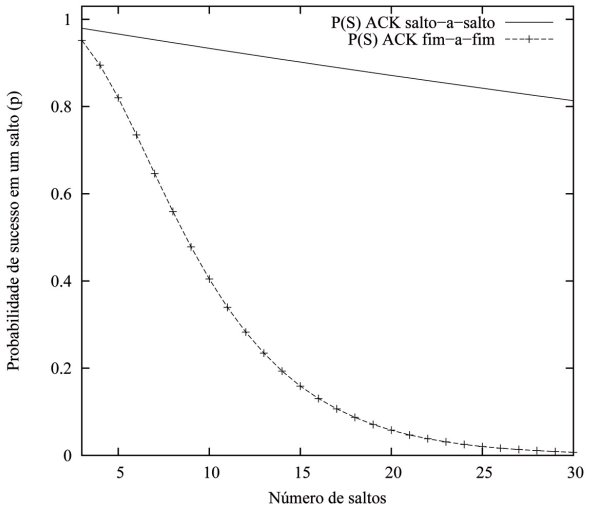

Figura 5. Probabilidade de entrega fim-a-fim, com ACK fim-a-fim e ACK salto-a-salto. ($R = 3$)

3.3.2. Análise da falha dos nós

Como vimos na seção 3.3.1, as falhas de comunicação podem ser tratadas através do envio de ACKs e posterior retransmissão corretiva de dados. Um problema adicional reside na possível falha do nó que contém a informação a ser transmitida.

No caso da transmissão com ACK fim-a-fim, a informação se mantém armazenada apenas no nó origem, até que um ACK informe que a mesma chegou ao destino. Um

defeito, dano ou esgotamento de energia neste nó antes que a mensagem chegue ao destino ocasionará sua perda definitiva.

No transporte com nível 1 de confiabilidade, a mensagem se mantém armazenada sucessivamente nos nós intermediários, até que o nó subseqüentes envie o ACK, assumindo a responsabilidade de mantê-la até que o próximo salto seja concluído. Mais uma vez, uma falha no nó detentor da informação causará a perda definitiva da mesma.

No transporte com nível 2 de confiabilidade, o armazenamento é feito por 2 nós adjacentes ao mesmo tempo, o que reduz a probabilidade de perda da mensagem. Apenas a falha destes dois nós causa a perda da informação.

Consideremos que a probabilidade de um nó **não** falhar é dada por f. A perda total de uma mensagem se dará caso as falhas de nós citadas acima ocorram em intervalos críticos.
Considere os nós A, B e C representados nas figuras 2 e 4.

Os intervalos críticos da confiabilidade no nível 1 são:

- B recebe a MSG de A com sucesso, envia o ACK para A e falha antes de enviar a MSG para C.
- B recebe a MSG de A com sucesso, envia o ACK para A e envia a MSG para C. A MSG falha e não chega em C. Antes de retransmitir B falha.

Observe ainda que a seqüência B_FALHOU E ACK_BA_FALHOU não implica na perda definitiva de MSG, já que A ainda não descartou MSG.

Os intervalos críticos da confiabilidade nível 2, são:

- Considere a figura 4. A perda definitiva da mensagem se dará se e somente se os nós A, B e C falharem, juntos, antes que C tenha enviado a MSG ao nó D. Ou seja, $ACK2\_BA\_OK\ E\ FALHA$ de **todos** os nós que já receberam MSG antes do último nó na cadeia enviar a MSG ou após este envio, mas com falha do mesmo.

A rigor, basta uma cópia de MSG ainda não reconhecida pela segunda vez (ACK2), ou seja, basta existir um nó que tenha MSG e que ainda não tenha recebido o ACK2 correspondente, para que MSG ainda não tenha sido perdida em definitivo.

Aqui identificamos o conceito de elasticidade do protocolo. Uma vez enviada pela origem, MSG se propaga indefinidamente (até o destino no máximo). Dependendo do sucesso de envio e recebimento das mensagens ACK2, haverá um maior ou menor consumo distribuído de buffer para uma mesma MSG. A elasticidade é boa, na medida em que pode gerar mais e mais cópias de MSG (mais uma vez dependendo do sucesso de envio e recebimento das mensagens ACK2), o que reduz a possibilidade de perda definitiva de MSG. O mais provável é que, de um modo geral, o protocolo se comporte da maneira esperada, ou seja, com 2 cópias de MSG na rede a cada instante.

Análise das falhas de nós na confiabilidade 1

Seja $f(t)$ a propabilidade de um nó falhar em um intervalo de tempo t

Seja $t1 =$ Intervalo crítico na confiabilidade 1

Seja $P_1(F)$ a probabilidade de perda definitiva de MSG na confiabilidade 1

Temos então: $P_1(F) = f(t1)$

Análise das falhas de nós na confiabilidade 2

Seja k o grau de elasticidade do protocolo.

k = número de nós subseqüentes ao mais antigo detentor de MSG que já receberam MSG.
(k varia com todas as caracteríticas momentâneas da rede, acesso ao meio, contenção, condições climáticas, densidade de nós, tráfego etc).

Seja $t2$ = Intervalo crítico na confiabilidade 2

Seja $P_2(F)$ a probabilidade de perda definitiva de MSG na confiabilidade 2

Temos então: $P_2(F) = (f(t2))^k$

Como $t1$ e $t2$ se referem a um instante entre um envio de um ACK e o envio com sucesso de uma MSG, pode-se considerar $t1 = t2$. Sendo a chance de sucesso no envio de um ACK aproximadamente a mesma que o envio de uma MSG, tem-se $k = 2$.

Logo: $P_2(F) = (f(t2))^2 = (f(t1))^2 = (P_1(F))^2$

Assim, a probabilidade de perda definitiva da mensagem é consideravelmente menor no esquema com confiabilidade 2, como vemos na figura 6.

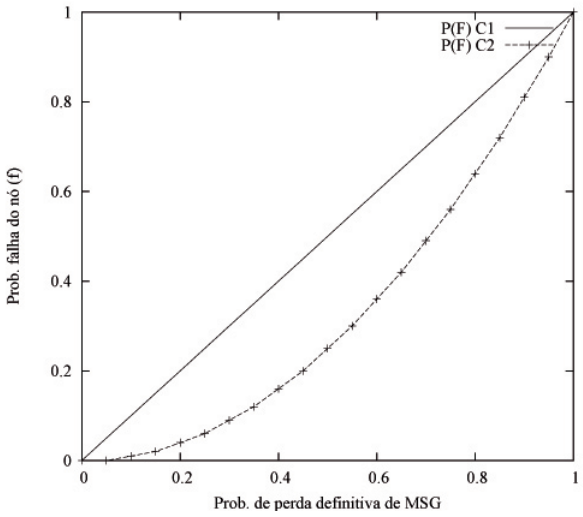

Figura 6. Probabilidade de perda definitiva da mensagem nos dois níveis de confiabilidade

3.4. Detecção e controle de congestionamento

Detecção e controle de congestionamento em redes de sensores, é um assunto importante. O mecanismo de detecção antecipada e randômico usado em redes tradicionais propõe que um nó intermediário descarte um pacote quando existe congestionamento na rede. A origem perceberá o descarte através de um ACK ou NACK. Contudo, descartar pacotes em redes de sensores não é uma solução aceitável.[Iyer et al. 2005]

Portanto, foi preciso considerar outras opções. Nas redes de sensores, as perdas de pacotes normalmente se referem a erros de transmissão e não a congestionamentos. Por isso, qualquer perda de pacote inicializaria o mecanismo de controle de congestionamento e reduziria a taxa de transmissão sem necessidade. Assim, faz-se necessária a implementação de um mecanismo de controle de congestionamento que saiba diferenciar uma perda de pacote relativa a esgotamento de *buffers* de um perda de pacote relativa a erro de transmissão.

Nesta proposta o controle de congestionamento é implementado através da participação de todos os nós intermediários na conexão de transporte. Estes nós gerenciam o congestionamento utilizando mensagens de sinalização. Desta forma, um nó se recusa a receber mais pacotes, caso seus *buffers* estejam ocupados. Logo, quando os *buffers* do nó B atingem o patamar T, este nó envia um pacote sinalizador (*STOP*), em *broadcast*, para todos os seus vizinhos, avisando que pacotes não podem mais ser enviados para ele pois seus *buffers* estão sem espaço de armazenamento. Tal atitude diminuirá a taxa de transmissão de seus vizinhos e conseqüentemente dos vizinhos dos vizinhos. Essa redução na taxa de transmissão poderá se propagar por toda a rede, dependendo do nível de congestionamento existente no momento.

Quando o nó conseguir esvaziar os seus *buffers*, ou seja, quando eles estiverem abaixo do patamar T, um novo pacote sinalizador (*START*) é enviado para liberar o encaminhamento de novos pacotes. Este patamar será calculado, a princípio, empiricamente, através de testes realizados. Portanto, consegue-se, através do mecanismo descrito nesta seção, concluir que toda perda de pacotes, será decorrente de erros de transmissão e não de congestionamento.

4. Conclusões e trabalhos futuros

Neste trabalho, nós apresentamos o Protocolo de Transporte Colaborativo para Rede de Sensores (CTCP). Este protocolo promove entrega confiável de mensagens entre um nó e sua respectiva estação base. Ele é capaz de detectar e controlar o congestionamento através da diferenciação entre as perdas relativas a erro de transmissão e aquelas geradas por esgotamento de *buffers*. Em caso de perdas, ele oferece recuperação em dois níveis de confiabilidade e, em caso de congestionamento, oferece sinalização explícita de interrupção e retorno do envio de dados.

Analisamos probabilisticamente o mecanismo de confiabilidade proposto e concluímos que a recuperação distribuída de erros (salto-a-salto) aumenta significativamente a probabilidade de entrega de mensagens fim-a-fim. Observamos ainda que, dividindo a responsabilidade de armazenamento temporário da mensagem entre dois nós adjacentes (nível 2 de confiabilidade), temos uma queda importante da probabilidade de perda definitiva de uma mensagem. Além disso, o armazenamento distribuído de mensagens até que

um ACK confirme que a mesma já se encontra sob responsabilidade do próximo (ou dos 2 próximos) nó(s) demonstra a robustez do protocolo diante de períodos de desconexão.

O CTCP não faz qualquer restrição quanto aos protocolos de níveis inferiores que serão usados.

A estação base assume as decisões que dependem do conhecimento dos requisitos da aplicação (confiabilidade requerida) e controla parâmetros com características centrais (ID da conexão). Por outro lado, as funções de controle de congestionamento, implementação da confiabilidade e abertura de conexões estão distribuídas na RSSF.

Em trabalhos futuros pretendemos investigar o efeito do aumento do número de saltos na geração de ACKs múltiplos no modelo de nível 2 de confiabilidade. Utilizaremos como métrica o aumento do custo de transmissão e de consumo de *buffers* versus aumento na probabilidade de entrega.

Através da implementação deste protocolo, em um ambiente de simulação, pretendemos analisar a multiplexação de fluxos (ou conexões) de transporte.

Referências

Allman, M., Paxson, V., and Stevens, W. (1999). Tcp congestion control. RFC 2581 (Proposed Standard). Updated by RFC 3390.

C. Intanagonwiwat, RC. Govindan, D. E. (2000). Direct diffusion: A scalable and robust communication paradigm for sensor networks. In *In Proceedings of the Sixth Annual ACM International Conference on Mobile Computing and Networking*, pages 56–67.

Iyer, Y. G., Gandham, S., and Venkatesan, S. (2005). Stcp: A generic transport layer protocol for wireless sensor networks. In *Proceedings of 14th International Conference on Computer Communications and Networks*, pages 449–454, Houston, TX, USA.

Kim, S., Fonseca, R., Dutta, P., Tavakoli, A., Culler, D., Levis, P., Shenker, S., and Stoica, I. (2007). Flush: a reliable bulk transport protocol for multihop wireless networks. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 351–365, New York, NY, USA. ACM.

Milner, R. (1980). *A Calculus of Communicating Systems*. Springer Verlag.

Nielsen, M., Plotkin, G., , and Winskel, G. (1981). *Petri Nets, Event Structures and Domains, Part I, Vol. 13*. Theoretical Computer Science.

Sankarasubramaniam, Y., Akan, O., and Akyildiz, I. (2003). Esrt: Event-to-sink reliable transport in wireless sensor networks. In *In Proceedings of MobiHoc 03, ACM*, Annapolis, Maryland, USA.

Stann, F. and Heidemann, J. (2003). Rmst: Reliable data transport in sensor networks. In *Proceedings of the First International Workshop on Sensor Net Protocols and Applications*, pages 102–112, Anchorage, Alaska, USA.

Vassis, D., Kormentzas, G., Rouskas, A. N., and Maglogiannis, I. (2005). The ieee 802.11g standard fo high data rate wlans. *IEEE Network*, 19(3):21–26.

Wan, C.-Y., Campbell, A. T., and Krishnamurthy, L. (2005). Pump-slowly, fetch-quickly (psfq): A reliable transport protocol for sensor networks. In *IEEE Journal on Selected Areas in Communications, Vol. 23, No. 4*, pages 862–872, Atlanta, Georgia, USA.