

# CTCP: Analisando a Entrega, Latência e Consumo de Energia

Eugênia Giancoli<sup>1,2</sup>, Filipe C. Jabour<sup>1,2</sup>, Aloysio C. P. Pedroza<sup>1</sup>

<sup>1</sup> UFRJ - PEE/COPPE/GTA - DEL/POLI

<sup>2</sup> CEFET-MG - Campus Leopoldina

{eugenia, jabour, aloysio}@gta.ufrj.br

**Abstract**—This work presents Collaborative Transport Control Protocol (CTCP), a new transport protocol for sensor networks. It aims at providing end-to-end reliability and adapts itself to different applications through a two level mechanism of reliability variation. CTCP achieves these properties using hop-by-hop acknowledgments and a storage control algorithm that operates at each node along a flow. It was observed that distributed fault recovery increases 94% the average delivery rate and that duplication of storage responsibility minimizes the message loss. Its congestion control differentiates communication losses from buffer overflow. CTCP is called collaborative because all nodes detect and act on congestion control and also because it includes distributed storage responsibility. It is scalable and independent of the underlying network layer. The protocol energy consumption overhead was calculated and discussed for two reliability levels.

**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 de um mecanismo de variação da confiabilidade. Esta flexibilidade é obtida através de reconhecimentos salto a salto e de um algoritmo distribuído de armazenamento que opera em cada nó pertencente ao fluxo. Observou-se que a recuperação de erros distribuída aumenta para 94%o percentual de mensagens entregues. Além disso, a duplicação da responsabilidade de armazenamento minimiza em aproximadamente 40% a perda de mensagens. 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. O aumento do consumo de energia, referente à introdução do CTCP, foi calculado e discutido para os dois níveis de confiabilidade.

## I. 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. Estas redes são caracterizadas por atrasos longos e variáveis, frequentes desconexões, altas taxas de erros e recursos limitados. 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. Portanto, 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 frequentes [1].

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. Examinou-se vários protocolos de transporte para redes de sensores e propõe-se uma nova solução para aumentar a confiabilidade na entrega dos dados através do Protocolo de Transporte Colaborativo, chamado CTCP (Collaborative Transport Control Protocol).

O restante deste trabalho está organizado da seguinte maneira: na seção II nós discutimos as considerações técnicas que norteiam o projeto do CTCP. Descrevemos os trabalhos relacionados na seção III. O CTCP é especificado na seção IV enquanto na seção V nós relatamos os resultados das simulações. Conclusões e trabalhos futuros podem ser encontrados em VII.

## II. CONSIDERAÇÕES DE PROJETO

Os nós sensores, geralmente, são espalhados por uma região de difícil acesso formando uma rede de sensores sem fio. Cada um dos sensores é capaz de coletar dados e roteá-los para o nó sorvedouro que está fisicamente conectado à estação base. Os dados são roteados para o nó sorvedouro através de uma arquitetura de múltiplos saltos. Este trabalho considera que a estação base e os nós utilizam a pilha de protocolos sugerida por [2]. Esta pilha de protocolos consiste das camadas de aplicação, transporte, rede, enlace de dados e física.

Este trabalho propõe soluções para criar uma entrega de dados confiável nas RSSF e por isso foca seus esforços na camada de transporte cujo principal objetivo é oferecer um serviço confiável, eficiente e econômico a seus usuários que, em geral, são processos presentes na camada de aplicação. A independência da rede física ou das camadas subjacentes é um requisito importante.

Uma importante questão a ser considerada é o *trade-off* entre a implementação da confiabilidade salto-a-salto na camada de enlace ou na de transporte, que, nas redes tradicionais, se concentra em prover confiabilidade fim-a-fim. De acordo com [3], mesmo que o protocolo MAC (da camada de enlace) possa recuperar pacotes perdidos através do

mecanismo correção de erros de bits, ele, normalmente, não possui maneiras de tratar os pacotes descartados por *buffer* cheio. Logo, o protocolo de transporte das RSSF deve possuir mecanismos de reconhecimento (ACK e/ou Selective ACK [4] para recuperar os pacotes perdidos.

Portanto, nas redes de sensores sem fio um dos aspectos a se considerar, com cuidado, é como detectar o congestionamento e como superá-lo, em função dos recursos reduzidos destas redes. Segundo [3], as soluções fim-a-fim são bastante simples e robustas, mas injetam muitos pacotes na rede. Em contrapartida, as soluções salto-a-salto podem rapidamente enfraquecer um congestionamento e injetam menos pacotes na rede. Contudo, nesta última abordagem, o comportamento de cada nó, entre a origem e o destino, precisa ser alterado. Como, menos pacotes na rede podem resultar em economia de energia, existe um *trade-off* entre os mecanismos salto-a-salto e fim-a-fim, que devem ser cuidadosamente considerados ao se projetar algoritmos de controle de congestionamento para as RSSF.

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.

De acordo com [5], [6], [3], os requisitos básicos para uma camada de transporte genérica para as 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 fim-a-fim:** 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.

**Controle de Fluxo:** é preciso usar um mecanismo de controle para evitar que o transmissor rápido sobrecarregue um receptor lento. Protocolos salto-a-salto estão aptos a controlar cada um dos saltos ao longo de uma rota de rede, o que, de acordo com [5], é uma clara vantagem sobre os protocolos fim-a-fim.

**Simplificação da conexão inicial:** Os protocolos de transporte para RSSF devem simplificar o processo de abertura de conexão ou usar protocolos sem conexão a fim de começar a transmissão de dados o mais rápido possível e sobretudo, garantir vazão e atraso mínimo. Algumas aplicações em RSSF são reativas a certos eventos específicos e esperam que estes eventos ocorram para começar a transmitir para a estação base. Estas aplicações possuem somente alguns pacotes para cada ocorrência do evento e, assim, um processo de abertura de conexão rápido seria mais efetivo e eficiente.

A perda de pacotes está sempre presente nas RSSF devido à má qualidade dos canais sem fio, falhas dos sensores, e congestionamento. As RSSF devem garantir certa confiabilidade aos pacotes ou às mensagens da camada de aplicação para conseguir obter dados íntegros da rede. Algumas aplicações críticas necessitam de transmissão confiável para cada pacote e consequentemente uma camada de que aumente a confiabilidade passa a ser necessária. De qualquer maneira, o primeiro passo consiste em detectar a perda de pacotes a fim de poder recupera-los. Métodos utilizados em troca de pacotes nas redes tradicionais podem ser aproveitados nas RSSF. Por exemplo, os mecanismos de reconhecimento (ACK ou NACK) podem ser usados para recuperar pacotes perdidos tanto nas abordagens fim-a-fim quanto nas abordagens salto-a-salto. Intuitivamente, se existem menos pacotes na rede e poucas retransmissões, a economia de energia aumenta. Controles de congestionamento eficientes podem resultar em poucos pacotes trafegando na rede e uma eficiente recuperação das perdas pode resultar em poucas retransmissões. Assim, controle de congestionamento eficiente e garantia de confiabilidade resultam em economia de energia nas redes de sensores.

Este trabalho propõe o CTCP. Ele é um protocolo de transporte colaborativo baseado em mecanismos conhecidos de reconhecimentos (ACK) e temporizadores. Os dois níveis de confiabilidade garantem flexibilidade ao CTCP e possibilitam a sua adaptação a diferentes tipos de aplicação. Este mecanismo objetiva suportar interrupções de conexão sem perda de dados. Mesmo quando um nó recebe os dados a serem transmitidos e falha antes de reencaminhá-los, o protocolo está apto a recuperar esta perda. O CTCP usa reconhecimentos salto-a-salto, considerados eficientes por [7], com liberação imediata dos *buffers*, o que aumenta a capacidade de reencaminhar pacotes e previne o congestionamento. Além disso, o protocolo prevê um mecanismo de controle de congestionamento apto a evitar perdas relativas a *buffer* cheio. O CTCP foi projetado para trabalhar com quaisquer camadas subjacentes.

### III. TRABALHOS RELACIONADOS

O primeiro protocolo analisado foi o TCP [4]. 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 está totalmente adaptado 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) [7] foi projetado para funcionar em conjunto com o Direct Diffusion Protocol [8], 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* e conseqüente congestionamento. 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) [9] 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, solicitando aos nós o aumento ou diminuição da taxa de transmissão. Não garante a 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) [10] 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) [6] 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. Considera-se questionável 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. No CTCP, 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.

#### IV. ESPECIFICAÇÃO DO PROTOCOLO

Para a modelagem e análise formal deste protocolo, foram empregados dois métodos, um formalismo matemático denominado Redes de Petri Predicado Ação [11] e uma linguagem de especificação chamada CCS (*A Calculus of Communicating Systems*), de Robin Milner [12]. 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.

As principais contribuições do CTCP são:

- Garantia de entrega de 98% dos segmentos, à camada de aplicação da estação base, mesmo na presença de falhas de nós e freqüentes desconexões.
- Dois perfis de confiabilidade visando economizar energia.
- Habilidade para 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.

##### A. Abertura e Fechamento da Conexão

Antes de iniciar a transmissão de dados, um pacote de abertura de conexão (ABR) é enviado, salto a salto, 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 uma mensagem de resposta (RSP) à origem da conexão. No cabeçalho do RSP são especificados o nível de confiabilidade requerido pela aplicação à qual o fluxo pertence e o identificador da conexão (ID), que é composto pelo ID do nó e pelo primeiro número de seqüência da conexão. Manter o identificador da conexão unívoco, 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 [13], possuírem fragmentação e agrupamento, existem limites no tamanho dos pacotes que uma entidade consegue fragmentar e garantir a entrega. [7]. 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.

Esta troca inicial de mensagens é feita utilizando-se o nível 1 de confiabilidade até que o pacote RSP chegue ao nó origem com o novo nível de confiabilidade determinado para aquele momento.

Quando a camada de aplicação do nó origem termina seu trabalho, envia um pacote de fechamento de conexão (CLO) à estação base. A estação base, então, libera os *buffers* e variáveis da conexão.

### B. Algoritmo de Confiabilidade Distribuído

Cada aplicação possui requisitos diferentes de confiabilidade. Algumas, por exemplo, suportam perdas de dados enquanto outras precisam de garantir que cada um dos seus pacotes chegará ao destino. Desta forma, para especificar a confiabilidade requerida é preciso que se tenha conhecimento da aplicação e seus objetivos. É preciso ressaltar, que o nível de confiabilidade, configurado durante a fase de abertura de conexão, pode ser alterado a qualquer momento, pelo usuário que interage com a estação base. A necessidade de alteração do nível de confiabilidade pode ser exemplificada como a esgotamento de energia dos nós. Neste caso, pode ser mais interessante diminuir a confiabilidade da rede para aumentar sua vida útil. Quando o nível de confiabilidade é alterado, um pacote RSP é enviado ao nó origem com o novo nível de confiabilidade solicitado.

Uma vez definido o nível de confiabilidade requerido pela aplicação, os nós da rede poderão agir de duas maneiras diferentes descritas a seguir:

#### Nível 1 de Confiabilidade

Este nível de confiabilidade visa economia de energia, através da redução de retransmissões, 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 guarda uma cópia do pacote no seu *buffer*, inicia o temporizador, reencaminha o pacote e envia ao nó A um reconhecimento (ACK) passando 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 1 e formalmente, através das Redes de Petri na figura 2.

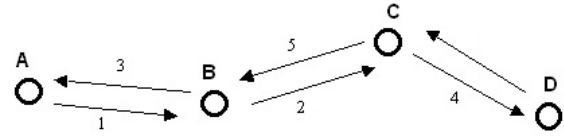


Fig. 1. Ordem da troca de mensagens no nível 1

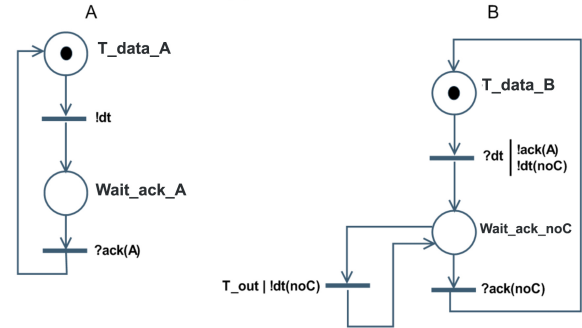


Fig. 2. Comunicação entre dois nós no nível 1

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, todavia, a seguinte situação: o nó A, origem do fluxo, envia dados ao nó B. B recebe os dados, armazena no *buffer*, inicia o temporizador, reencaminha o pacote e envia um ACK ao nó A. Por um erro qualquer o pacote não consegue chegar ao nó consecutivo (C). Logo em seguida, o nó B falha. Deste modo, os dados que estavam sob a responsabilidade do nó B não serão retransmitidos para a estação base e o nó A não perceberá esta falha. Esta situação só poderá ser minimizada com o aumento do nível de confiabilidade.

A seguir, pode-se ver o algoritmo do nível 1 de confiabilidade formalmente especificado em CCS. Considere que  $B!dt$  significa o envio de  $dt$  ao nó B, e  $A?dt$  significa a recepção de  $dt$  enviado por A. Um maior detalhamento sobre CCS poderá ser encontrado em [12].

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

$$\begin{aligned} A &= !dt. Wait\_ack\_A \\ Wait\_ack\_A &= ?ack(A). A \\ B &= ?dt. lack. !dt(C). !starttimer. Wait\_ack\_noC \\ Wait\_ack\_noC &= ?ack(C). !reset\_timer. B + \\ & ?t\_out. !dt(C). !start\_timer. Wait\_ack\_C \\ Timer &= ?start\_timer. (!t\_out. Timer + \\ & ?reset\_timer. Timer) \end{aligned}$$

#### 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 após receber o segundo ACK *A* descartará os dados mantidos em *buffer*. Todos os nós repetirão este processo, sucessivamente, até que os dados cheguem à estação base. A estação base deverá enviar dois ACKs ao nó imediatamente anterior. A figura 3 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.

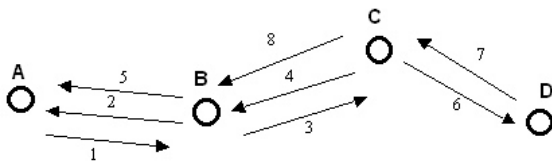


Fig. 3. Ordem da troca de mensagens no nível 2

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 figura 4 ilustra a especificação formal do nível 2.

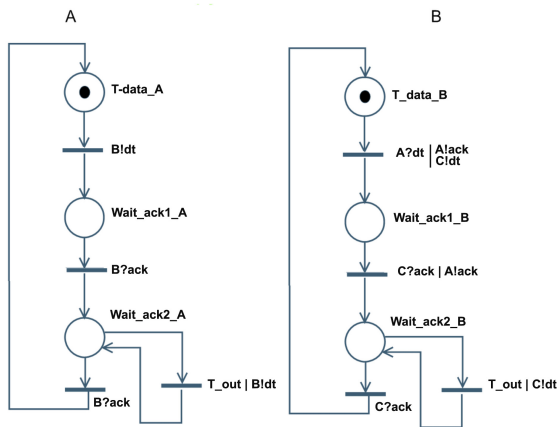


Fig. 4. Comunicação entre dois nós no nível 2

A seguir, pode-se ver o algoritmo do nível 2 de confiabilidade formalmente especificado em CCS. Considere que *B!dt* significa o envio de *dt* ao nó *B*, e *A?dt* significa a recepção de *dt* enviado por *A*.

$$Level\_2 = A || B || Timer$$

$$\begin{aligned} A &= B!dt.Wait\_ack1\_A \\ Wait\_ack1\_A &= B?ack.Timer!start.Wait\_ack2\_A \\ Wait\_ack2\_A &= B?ack.Timer!reset.A+ \\ &\quad Timer?t\_out.B!dt.Timer!start.Wait\_ack2\_A \\ B &= A?dt.C!dt.A!ack.Wait\_ack1\_B \\ Wait\_ack1\_B &= \\ &\quad C?ack.A!ack.Timer!start.Wait\_ack2\_B \\ Wait\_ack2\_B &= C?ack.Timer!reset.B+ \\ &\quad timer?t\_out.C!dt.timer.start.Wait\_ack2\_B \\ Timer &=?start!(t\_out.timer+?reset.timer) \end{aligned}$$

### C. 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 a solução ideal [6]. 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 uma 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. Cada nó da rede mantém uma tabela com o identificador das conexões ativas para fins de multiplexação de fluxos. Desta maneira, é possível saber quais são os vizinhos que estão enviando dados no momento. Caso um destes vizinhos ativos continue a enviar pacotes depois do *broadcast STOP*, um novo pacote *STOP* será enviado diretamente para ele (*unicast*). Contudo, os pacotes enviados neste intervalo não terão sido descartados pois o cálculo do patamar *T* considera esta situação.

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. Novamente, um ou mais vizinhos podem não receber o pacote *START* o que acarretará um travamento

temporário do vizinho. A partir da tabela de conexões ativas é possível identificar quais são os vizinhos temporariamente travados e reenviar diretamente para ele (unicast) um pacote *START*. O patamar  $T$  será calculado, a princípio, empiricamente, através de testes realizados. Desta maneira, consegua-se, através do mecanismo descrito nesta seção, concluir que toda perda de pacotes, será decorrente de erros de transmissão e não de congestionamento.

## V. ANÁLISE DE RESULTADOS

As maiores funções de um protocolo de transporte para redes de sensores são controle de congestionamento, garantia de entrega e conservação de energia que pode ser alcançada através de eficientes controle de congestionamento e garantia de entrega [3].

No trabalho [14], o CTCP foi especificado e analisado probabilisticamente. Aqui, para entender o comportamento do algoritmo de confiabilidade do CTCP, nós fizemos várias simulações usando o TinyOS Simulator (TOSSIM) [15]. A rede do TOSSIM é representada por um grafo direcionado, no qual cada vértice é um nó, e cada aresta possui uma probabilidade de erro de bit. A existência de uma aresta  $(a, b)$ , no grafo, significa que o sinal de  $a$  pode ser ouvido por  $b$ . Cada aresta possui um valor, contido no intervalo  $(0, 1)$ , que representa a probabilidade de não entrega de cada bit transmitido. Por exemplo, um valor de 0.01 significa que cada bit transmitido tem 1% de chance de ser descartado (corrompido), enquanto 1.0 significa que todos os bits transmitidos serão descartados, e 0.0 significa que todos os bits serão transmitidos sem erros. Cada bit é considerado individualmente.

Nós ajustamos os parâmetros do CTCP empiricamente. Simulamos redes com 25, 49 e 100 nós sensores distribuídos em áreas de 50m X 50m, 70m X 70m e 100m X 100m, respectivamente. Nota-se que um espaçamento de 10 pés (3,05 m) foi mantido entre os nós em todas as topologias. Desta maneira, ao aumentarmos o número de nós, estamos aumentando também o número de saltos até a estação base. Durante a simulação um único sensor faz leituras e gera mensagens. Ele envia um pacote a cada 50 segundos do simulador. O raio de transmissão de cada nó é de 50 pés (15,24 m). Combinado com a taxa de erros, isto significa que cada nó transmite seu sinal em um disco de 50 pés de raio, com a taxa de erro aumentando a medida que se afasta do centro. O pacote de dados tem 216 bits de tamanho enquanto o pacote de reconhecimento possui 72 bits.

Para rotear os pacotes até a estação base, nós usamos o protocolo de roteamento padrão do TOSSIM, chamado MintRoute. O MintRoute é um protocolo de roteamento proativo no qual os nós enviam mensagens de roteamento periódicas para informar sobre o seu estado local. O simulador usa o CSMA como protocolo padrão da camada de enlace. As simulações foram executadas durante 1000 segundos do simulador.

Para avaliar o desempenho do CTCP, usou-se as seguintes métricas:

- Fração de pacotes entregues com sucesso.

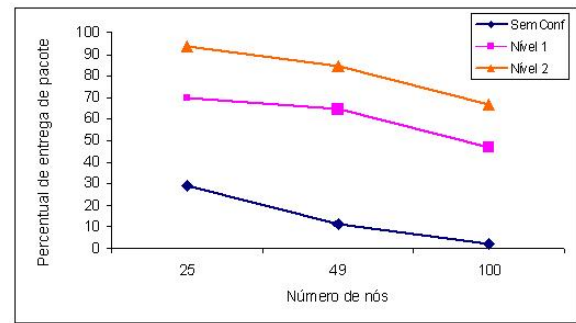


Fig. 5. Probabilidade de entrega de pacotes

- Consumo de Energia.
- Latência.

### A. Fração de Pacotes Entregues com Sucesso

A confiabilidade é medida como a fração de pacotes transmitidos e efetivamente recebidos na presença de erros de bits e falha de nós. A figura 5 ilustra a probabilidade de entrega em três situações diferentes. Na primeira, a aplicação roda diretamente sobre a camada de rede, sem protocolo de transporte. Na segunda situação, o CTCP foi introduzido e configurado para confiabilidade nível 1. E nas últimas séries de simulações, o CTCP foi ajustado para nível 2 de confiabilidade.

A figura 5 demonstra a probabilidade de entrega para três redes diferentes: com 25, 49 e 100 nós. Na rede com 25 nós a entrega é de 28,88% sem nenhum protocolo de transporte. A mesma aplicação, na mesma rede, executando o CTCP com confiabilidade nível 1, entrega 70% dos pacotes enquanto o nível 2 chega a 94%. Note que nestas simulações, o protocolo de transporte é imprescindível, pois mesmo para uma aplicação que possui grande redundância de dados, uma entrega de apenas 28,88% dos pacotes pode ser inviável. A utilização do nível 2 de confiabilidade é apropriado para aplicações onde existe pouca redundância de pacotes.

Nas redes com 49 e 100 nós, podemos notar que existe uma queda na taxa de entrega de pacotes. Esta queda se dá principalmente em função da latência da rede que será analisada na seção VI.

Os arquivos de logs das simulações mostram que o CTCP nível 2 pode alcançar 100% de entrega de pacotes com alguns ajustes na configuração do temporizador. Neste caso, o total de pacotes não entregues está relacionado a uma alta latência e não à perda permanente.

É importante ressaltar que as simulações foram executados com apenas um nó origem, fazendo leituras e consequentemente gerando pacotes a cada 50 segundos do simulador. Se aumentarmos o número de nós que fazem leituras e geram pacotes, passaremos a ter uma grande probabilidade de congestionamento. Neste caso, será muito importante o uso do mecanismo de controle de congestionamento descrito na seção IV-C. O Controle de congestionamento reduz o número de retransmissões devido a *buffer* cheio, consequentemente reduzindo a latência na rede. A implementação deste mecanismo

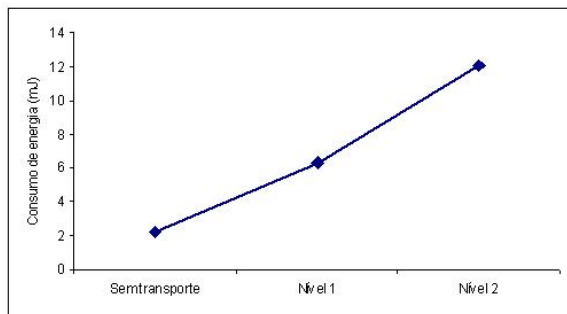


Fig. 6. Consumo de Energia para as redes com 25 nós

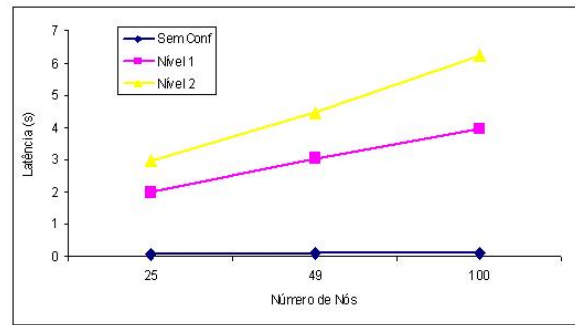


Fig. 7. Latência para as redes com 25, 49 e 100 nós

está em curso.

### B. Consumo de Energia

Na seção V-A, podemos ver que o CTCP é capaz de prover altos níveis de confiabilidade. Como a energia é um recurso escasso nas redes de sensores, é necessário medir o aumento do consumo de energia gerado pela inserção do protocolo de transporte.

Para as simulações descritas acima (sem confiabilidade, nível 1 e nível 2), foram contados o número de total de pacotes transmitidos. Considerou-se duas classes distintas de pacotes: pacotes de dados e pacotes de reconhecimento (quando existiam). O tamanho do pacote de dados é de 216 bits incluindo a mensagem da camada de aplicação (a leitura do sensor), o cabeçalho do CTCP, camada de rede e enlace. O pacote de reconhecimento, que é enviado diretamente da camada de transporte para a camada de enlace (não passa pela camada de rede) é destinado ao endereço de enlace (MAC address) e tem apenas 72 bits.

Assume-se que o rádio dissipa 50 nJ/bit para acionar o circuito de transmissão ou recepção e 0.1nJ/(bit.m2) para amplificar a transmissão de maneira a alcançar um SNR [16] aceitável.

Baseado nos valores acima, calculamos o consumo de energia da camada de aplicação e do CTCP de cada nó. A soma dos consumos individuais é o consumo da rede. A figura 6 mostra o consumo de energia das camadas de aplicação e transporte das três redes diferentes. Todos os valores estão em (mJ).

Como era esperado, a figura 6 nos mostra um consumo maior de energia para prover um maior índice de confiabilidade. Depois de 1000 segundos, o total de energia gasto na rede é de 12,07 mJ no nível 2, 6,29 mJ no nível 1 e 2,19 mJ sem protocolo de transporte. Observe que estes baixos valores (na ordem de milijoules) ocorrem porque existe somente um nó gerando pacotes na rede e somente 15 mensagens são geradas em cada simulação. Isto é aceitável, uma vez que o objetivo principal da simulação é comparar os três níveis de confiabilidade.

A energia gasta para prover a confiabilidade nível 2 é 1,91 vezes maior que a energia gasta para prover o nível 1 e 5,51 vezes o total gasto na simulação sem o protocolo de transporte.

A energia consumida pelas camadas inferiores (rede, enlace e física) é o mesmo em todos os casos estudados. Assim, a sobrecarga proporcional imposta pelo CTCP se torna menos significativa.

## VI. LATÊNCIA

Considerou-se que a latência de um determinado pacote é a diferença entre o instante de chegada da mensagem na estação base e o instante no qual a mensagem foi gerada. A média aritmética simples, destes valores, gerou o que chamamos de latência da rede. Deste modo, a figura 7 ilustra a latência de três redes diferentes: com 25, 49 e 100 nós respectivamente. Pode-se concluir, a partir da figura 7 que a latência da rede aumenta à medida que aumentamos o número de nós. Tal fenômeno ocorre em função do aumento do número de saltos, ou seja, quanto maior o caminho a ser percorrido mais tempo leva-se para completá-lo.

Nós pudemos observar que o valor do temporizador é diretamente proporcional à latência da rede. Fizemos testes com o temporizador configurado para 30 segundos e também para 1 segundo. A latência da rede na primeira situação foi trinta vezes maior que na segunda.

A latência também está diretamente ligada ao número máximo de retransmissões por nó. Neste caso, mantivemos o número de retransmissões em 3 para todas as simulações.

Como era esperado, a latência da rede sem o protocolo de transporte é muito baixa, na ordem de milissegundos. Deve-se considerar que esta latência foi calculada levando-se em consideração somente as poucas mensagens que chegaram.

A latência aumenta à medida que aumentamos o nível de confiabilidade do protocolo. O CTCP nível 2 entrega mais mensagens que o CTCP nível 1, esta diferença na entrega de mensagens acontece principalmente em função de uma maior distribuição de armazenamento das mensagens. O armazenamento distribuído aumenta a possibilidade de uma retransmissão também distribuída.

## VII. CONCLUSÕES E TRABALHOS FUTUROS

Neste trabalho nós apresentamos o protocolo de transporte colaborativo para redes de sensores sem fio (CTCP). Este protocolo se propõe a prover entrega confiável entre o sensor origem e a estação base nas redes de sensores. O CTCP

ajuda a detectar e controlar o congestionamento através da diferenciação entre as perdas relativas a *buffer* cheio e aquelas decorrentes de erros de transmissão. No caso de perda ele oferece recuperação das mensagens através de dois níveis de confiabilidade, e em caso de congestionamento, provê uma sinalização explícita para interromper e/ou continuar a transmissão de dados.

O CTCP, no nível 2, entrega, em média, 65% a mais de mensagens do que em sua ausência. O protocolo apresenta um decréscimo na perda definitiva de mensagens através de seu algoritmo distribuído de armazenamento e retransmissão. Além disso, o armazenamento distribuído aumenta a robustez do protocolo durante grandes períodos de desconexão, anteriores ao recebimento dos ACKs.

Considerando a energia consumida pelas camadas subjacentes (rede, enlace e física), o aumento de consumo imposto pelo CTCP torna o protocolo viável.

O CTCP é um protocolo flexível no sentido de não ter nenhuma exigência quanto às camadas inferiores.

A estação base toma as decisões que dependem do conhecimento global da aplicação (confiabilidade requerida) e controla parâmetros que possuem características globais. Por outro lado, as funções de controle de congestionamento, confiabilidade e abertura de conexão estão distribuídas pela RSSF.

Nos trabalhos futuros investigaremos o efeito de aumentar o número de saltos na geração dos múltiplos ACKs no nível 2 de confiabilidade. Pretendemos avaliar o desempenho do protocolo com vários nós origem (source) e múltiplos fluxos. Implementaremos e testaremos o mecanismo de controle de congestionamento.

## REFERENCES

- [1] S. Kim, R. Fonseca, P. Dutta, A. Tavakoli, D. Culler, P. Levis, S. Shenker, and I. Stoica, "Flush: a reliable bulk transport protocol for multihop wireless networks," in *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, 2007, pp. 351–365.
- [2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.
- [3] C. Wang, K. Sohrawy, B. Li, and W. Tang, "Issues of transport control protocols for wireless sensor networks," in *Communications, Circuits and Systems, 2005. Proceedings. 2005 International Conference on*, vol. 1, Arkansas Univ., Fayetteville, AR, USA, May 2005, pp. 422–426.
- [4] M. Allman, V. Paxson, and W. Stevens, "Tcp congestion control," RFC 2581 (Proposed Standard), apr 1999, updated by RFC 3390. [Online]. Available: <http://www.ietf.org/rfc/rfc2581.txt>
- [5] S. Heimlicher, R. Baumann, M. May, and B. Plattner, "Saft: Reliable transport in mobile networks." Vancouver B.C., Canada: IEEE MASS 2006, Oct. 2006, pp. 477–480.
- [6] Y. G. Iyer, S. Gandham, and S. Venkatesan, "Stcp: A generic transport layer protocol for wireless sensor networks," in *Proceedings of 14th International Conference on Computer Communications and Networks*, Houston, TX, USA, 2005, pp. 449–454.
- [7] F. Stann and J. Heidemann, "Rmst: Reliable data transport in sensor networks," in *Proceedings of the First International Workshop on Sensor Net Protocols and Applications*, Anchorage, Alaska, USA, 2003, pp. 102–112.
- [8] D. E. C. Intanagonwiwat, R. Govindan, "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*, Aug 2000, pp. 56–67.
- [9] Y. Sankarasubramaniam, O. Akan, and I. Akyildiz, "Esrt: Event-to-sink reliable transport in wireless sensor networks," in *In Proceedings of MobiHoc 03, ACM*, Annapolis, Maryland, USA, June 2003. [Online]. Available: [citeseer.ist.psu.edu/641915.html](http://citeseer.ist.psu.edu/641915.html)
- [10] C.-Y. Wan, A. T. Campbell, and L. Krishnamurthy, "Pump-slowly, fetch-quickly (psfq): A reliable transport protocol for sensor networks," in *IEEE Journal on Selected Areas in Communications*, Vol. 23, No. 4, Atlanta, Georgia, USA, 2005, pp. 862–872.
- [11] M. Nielsen, G. Plotkin, , and G. Winskel, *Petri Nets, Event Structures and Domains, Part I, Vol. 13*. Theoretical Computer Science, 1981.
- [12] R. Milner, *A Calculus of Communicating Systems*. Springer Verlag, 1980.
- [13] D. Vassiss, G. Kormentzas, A. N. Rouskas, and I. Maglogiannis, "The ieee 802.11g standard fo high data rate wlangs," *IEEE Network*, vol. 19, no. 3, pp. 21–26, 2005.
- [14] E. Giancoli, F. C. Jabour, and A. C. P. Pedroza, "Collaborative transport control protocol," in *IEEE International Conference on Computer and Eletrical Engineering*. IEEE Computer Society, 2008.
- [15] P. Levis, N. Lee, M. Welsh, and D. Culler, "Abstract tossim: Accurate and scalable simulation of entire tinyos applications," 2003.
- [16] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS '00: Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*. Washington, DC, USA: IEEE Computer Society, 2000, p. 8020.