Collaborative Transport Control Protocol for Sensor Networks

Eugenia Giancoli
GTA - PEE - COPPE - UFRJ
CEFET/MG - Brasil
eugenia@gta.ufrj.br

Filippe Jabour
GTA - PEE - COPPE - UFRJ
CEFET/MG - Brasil
jabour@gta.ufrj.br

Aloysio Pedroza
GTA - PEE - COPPE - UFRJ
Brasil
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 mechanism of reliability variation. Its congestion control and detection 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. It was observed that distributed fault recovery increases reliability and that duplication of storage responsibility minimizes the message loss.

1. Introduction

Sensor networks are deployed for a wide range of applications in the military, environmental, health, industrial, and office domains. These networks are characterized by long or variable delays, frequent breakdown of connectivity, high error rates and limited resources. Each application has different characteristics and requirements of data types, transmission rates and reliability. Existing transport layer protocols for sensor networks are either tailored for certain applications, or assume that nodes employ a particular network layer or MAC layer protocol. As a result, these protocols cannot be applied across many sensor network deployments. Thus, it is necessary to design a transport protocol that can support multiple applications on the same network, provide controlled variable reliability, address congestion, reduce losses and support frequent disconnections [4]. This paper explores design decisions regarding one transport protocol that supports reliable data delivery in wireless sensor networks. We examined various transport protocols for sensor networks and propose a new solution to increase the reliability of data delivery by developing a Collaborative Transport Control Protocol (CTCP).

In what follows, we discuss technical considerations that underpin the design of CTCP, section 2. CTCP will be specified in section 3 and it is modeled probabilistically in section 4. In the section 5, we mention conclusions and further work.

2. Desing Considerations

The sensor nodes are usually deployed over a difficult access area, forming a wireless sensor network. Each sensor is capable of collecting and routing data to sink, who is connected by cable to base station. The data is routed through a multiple hop architecture. This paper considers that base station and nodes use the protocol stack suggested by [1]. The protocol stack consists of the application layer, transport layer, network layer, data link layer, physical layer.

This paper proposes solutions to create a reliable data transfer on WSNs and therefore focuses its efforts on transport layer, whose primary goal is to provide a reliable, efficient, and economical service for application layer processes. The independence of physical network and underlying layers is a feature of the protocol presented here.

An important issue to be considered is the trade-off between the implementation of hop-by-hop reliability into transport or link layer. According to [8], even if the MAC protocol (the link layer) can recover lost packets through bit-error mechanism, it does not usually have ways to recover the packets discarded by buffer overflow. Therefore, WSN transport protocol must have mechanisms to recover packets loss.

Moreover, [8] cites that end-to-end solutions are quite simple and robust, but inject many packets on the network. However, the hop-by-hop solutions can quickly weaken congestion and bring fewer on-going packets in networks, while it needs to change the behavior of each node on its way from source to destination. Therefore, having fewer on-going packets can result in saved energy.

According to [2, 3, 8], the basic requirements for a generic transport layer of sensor networks are: Heterogeneity, reliability, congestion control, flow control and initial connection simplification.

Packet loss is usual under WSN due to bad quality of wireless channels, sensors failure, and congestion. WSNs must guarantee certain reliability in packet- or application-level through loss recovery in order to abstract correct information. Some critical applications need reliability. Other applications need only a proportionally reliable transmission of total packets and thus application reliability is needed. Anyway, we first need to detect packet loss in order to correctly recover missing packets. After detecting packet loss, ACK and/or NACK (and their variant) can be used to recover missing packets based on an end-to-end or hop-by-hop approach. Likewise in congestion control, there is still a trade-off between end-to-end and hop-by-hop approach, which should be thought over. When designing transport control protocols for wireless sensor networks, we must consider energy conservation as well. Intuitively, if there are few on-going packets and few re-transmissions, energy can be saved. Effective congestion control can result in fewer on-going packets and effective loss recovery approach can result in fewer re-transmissions. So, congestion control and reliability guarantee can additionally save energy in a wireless sensor network. In short, the problem of transport control protocols for sensor networks is how to effectively control congestion and how to guarantee reliability while conserving as more energy as possible simultaneously.

This paper presents CTCP. It is a collaborative transport control protocol based on known mechanisms of packets acknowledgments (ACK) and timeout. Our work uses hop-by-hop acknowledgments, proved efficient in [7], but provides immediate node buffer release. Buffer release increases forwarding capacity and avoids congestion. In CTCP, the network clock synchronization is not needed unlike [3]. CTCP aims to support connection interrupt without data loss. Even when a node receives the data and fails before forwarding them, the protocol is able to recover the loss. It was designed to work with any underlying network layer. It has a control congestion service able to avoid losses related to buffer overflow. The two levels of reliability ensure CTCP flexibility and adapt to different applications types.

3. Protocol Specification

The new contributions of CTCP are: Delivery of all segments to base station application layer, even in the presence of nodes failures and frequent disconnections. Two reliability profiles to save energy. Ability to differentiate congestion loss from transmission error loss. Congestion control through the interruption of packets forwards (or immediate release of packets forwards) and independence from underlying layers.

Two methods were used for protocol modeling and formal analysis. A mathematical formalism called Predicate-action net [6] and a specifying language called CCS (*The Calculus of Communicating Systems*), of Robin Milner [5]. These are formal specification methods that allow systems development with a minimum ambiguity, through a well-defined syntax and semantics. A formal specification of our protocol provides an analysis of certain properties, such as lack of deadlocks blocks, correct timing and message sequence, and the verification of specification consistency.

3.1. Hop-by-Hop connection open and close

Before starting data transmission, a packet (ABR) is sent from source to base station. This packet informs the data flow identifier and the first sequence number to the base station. When the base station receives this packet, it reserves the buffers, initializes required variables and sends a response (RSP) message to source node. RSP header specifies the reliability level required by the application to which the flow belongs. It also specifies the connection identifier (ID), controlled by the base station, in order to prevent different connections with the same ID. Controlling the connection identifier provides flexibility to the protocol, since this data will be used to implement the multiplexing of different network flows. This implementation will be addressed in future works.

The source node will be able to initiate the data transmission after it receives the response (RSP). The protocol was designed to be as generic as possible. So, CTCP was concerned to establish packet formats compatible with the majority of underlying networks. Thus, it was necessary to consider that wireless communication and especially the sensor networks have a particular difficulty to transmit packets larger than the network MTU. Despite some protocols, such as 802.11, have fragmentation and reassembly, there are limits for packets size that an entity can fragment and guarantee delivery [7]. Therefore, the sensors that use the CTCP will be pre-configured with the MSS (Maximum Segment Size) permitted by underlying network layers. This variable does not need to be negotiated during connection opening phase.

In this protocol, reliability is not defined by source node, once global knowledge of an application and a network would be needed to make such a decision. This hop-by-hop connection uses readability level 1 until the RSP reaches source node with the real readability determined by base station.

When source node application layer ends its work, it sends a packet (CLO) to base station requesting connection closing. The base station then releases buffers and variables.

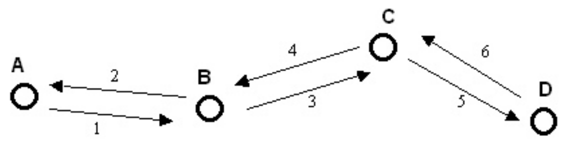

Figure 1. Level 1 messages exchange order

3.2. Controllably Reliable Delivery

Each application has different requirements of reliability. Thus, in order to specify the required reliability, it must know the application and its goals. Unlike the work [3], in CTCP, this decision will not be taken by nodes, since they do not have an application overview.

In our proposal, base station is responsible for stipulating the required application reliability level. Moreover, the reliability level, which is set at the connection opening phase, may be dynamically changed, when it is necessary, by the user who interacts with the base station. This need can be represented, for example, by the nodes energy exhaustion. In this case it may be more interesting to work with less reliability to maximize useful network life. When the user changes readability level, an RSP packet is sent to source node with a new level.

Once the user sets the required application reliability level, network nodes can act in two different ways, described below.

Reliability Level 1

This reliability level is intended to save energy by reducing transmissions. It has low cost of buffers and applies mainly to applications that have some data redundancy or that can tolerate losses. After receiving a packet from node A, node B stores a copy in its buffer and sends an acknowledgment (ACK) to node A. Node B is temporarily responsible for packet delivery to base station. This process happens repeatedly, through the network layer stipulated route, until the base station receives data packet and sends an ACK to the preceding node. Any node that receives an ACK may discard the sent packet, saving space in its buffer. This is represented in Figure 1.

Note that when the intermediary node assumes the responsibility to deliver the packet, it must keep a copy of this packet in buffer until it receives the ACK. The absence of an ACK generates a forwarding node clock fired and retransmission of the unrecognized packet.

Consider, however, the following situation: node A, source, sends data to node B. B, receives data, stores and sends ACK to node A. At this moment, node B fails. Therefore, the data that were under node B responsibility will not be forwarded to base station and node A will not notice this flaw. This situation can only be resolved by increasing the reliability level.

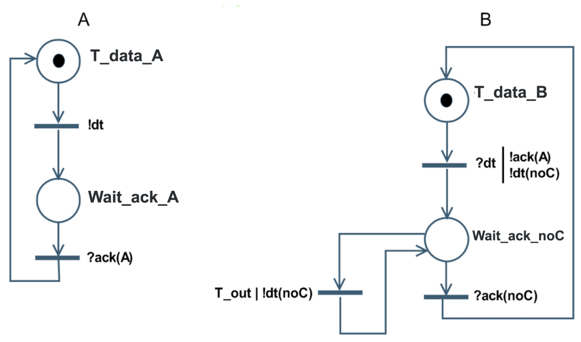

Figure 2. Two nodes level 1 communication

In what follows, we can see reliability level 1 formally specified in CCS. Consider that $B!dt$ means the sent of dt to B, and $A?dt$ means the reception of dt sent by A. More details in [5].

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

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

Reliability Level 2

Node A sends data to node B and waits to receive the *double ACK*. The double ACK is generated as follows: B receives data from A and sends A the first ACK. B sends the data to C that sends B the first ACK. When B receives the first ACK from C, it forwards the second ACK to A. Immediately, A discards the data kept in buffer. All nodes repeat this process, successively, until the data reaches base station. The base station should send two ACKs for the ultimate none. Figure 3 represents this message exchange.

If node B fails before delivering data to node C, node A will not receive the second ACK (double ACK) and will retransmit the packet. Note that starting from the assumption that failures of nodes are monitored by routing algorithms, these algorithms will be responsible for reconstructing the route in the presence of one node failure or a set of them.

The protocol described above enables the message to reach its destination with a greater probability, since one node failure in the path does not interrupt data delivery. A formal specification of level 2 is very similar to level 1.

Below, reliability level 2 formally specified in CCS.

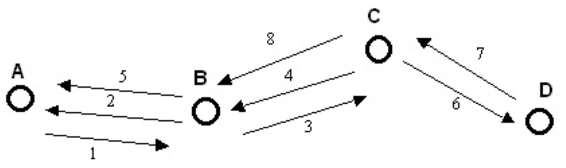

Figure 3. Level 2 messages exchange order

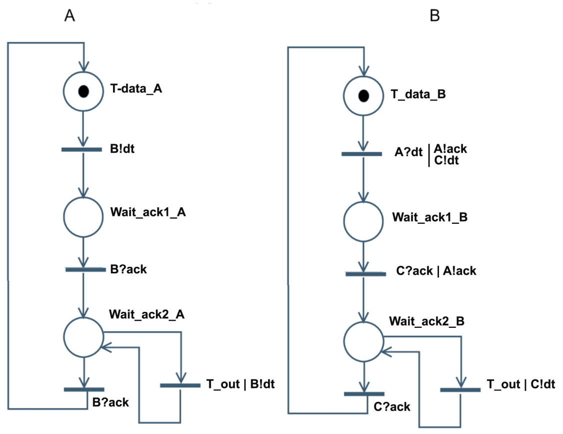

Figure 4. Two nodes level 2 communication

Again, consider that $B!dt$ means the sent of dt to B, and $A?dt$ means the reception of dt sent by A. More details in [5].

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

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

3.3. Detection and congestion control

Discading packets on sensor networks is not an ideal solution. Therefore, it was necessary to consider other options. In sensor networks, the packet loss usually refers to transmission errors and not to the network congestion. Any packet loss triggers the controlling congestion mechanism and transmission rate reduces without necessity. Thus, it is necessary to implement a control congestion mechanism that considers the difference between a transmission error packet loss and the buffer overflow.

In this proposal, congestion control is implemented through the participation of all nodes. These nodes manage congestion using signaling messages. Thus, a node refuses to receive more packets, if their buffer is up the threshold. So when buffer nodes reach threshold T, the node broadcasts a packet flag *(STOP)* for all its neighbours. This signaling packet warns that packets can no longer be sent to it as its buffer is overflowing. That would reduce the neighbours' transmission rate. This reduction in transmission rate may be flooded across the network, depending on the congestion level.

When nodes achieve their empty buffer, or when they are below threshold T, a new packet flag *(START)* is sent to release the forwarding of new packets. Every node in the WSN maintains a table with its active neighbours ID and the corresponding connection identifiers. After sending the START package, these neighbours start transmitting their data packages again. If this does not happen, it is possible to identify those that did not receive the START package. Thus, a new START package will be sent in unicast for those neighbors that did not resumed its transmissions. The computation of threshold T must consider that, when a node i sends the STOP package in broadcast, one or more neighbours may not receive it due to the increased probability of transmission collisions during congestion. Noting that one or more neighbours are still sending packages, node i sends the STOP package in unicast only for these nodes. So, it is possible, through the mechanism described in this section, to conclude that all packet losses will be due to errors in transmission rather than congestion.

4. Reliability probabilistic analysis

Traditional networks provide end-to-end reliability on transport layer through error recovery mechanisms. This approach is not ideal for sensor networks. In traditional networks, the intermediate nodes are just level 3 routers. In sensor networks all nodes have a transport layer which makes it possible to distribute the error recovery task. This collaboration between nodes becomes feasible because all nodes belong to the same administrative entity and want to reach the same goal.

Message delivery in end-to-end model:

Consider that p is the successfully exchanged message through a single hop. Thus, the error rate in the communication channel is $(1-p)$. q is the probability of success in sending an ACK for a single hop. Let R be the maximum number of retransmitions that can be done at the transport level.

Note that the use of a N end-to-end single ACK to loss recovery is subject to the vulnerabilities accumulated throughout all the route. Thus, for success in the end-to-end transmission, the message will have to cross the h hops between source and destination node and an ACK will have to cross the same h hops back (assuming that there has been no route change). In case of failure, any of the R retransmits occur once more end-to-end.

Message delivery in hop-by-hop level 1 model:

As described in section 3.2, the CTCP protocol proposes the hop-by-hop error recovery, with ACKs sent by each neighbor to the previous node. Thus, a probabilistic analysis becomes the following:

Probability of success $P(s)$ for a single hop:

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

Message delivery probability of success, encapsulated in a single packet, to base station, h hops away from source node, will be:

$$P(s)_{ACK\ hop-by-hop} = (P(s))^h \tag{2}$$

Throught the equations 1 and 2 we could see that the hop-by-hop recovery model provides a much greater delivery guarantee than the end-to-end recovery scheme. See figure 5

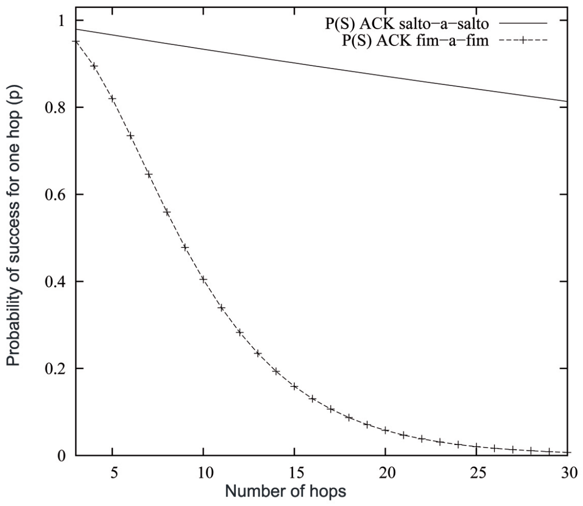

Figure 5. ACK Delivery Probability (R = 3)

5. Conclusions and future work

In this paper, we present the Collaborative Transport Protocol for Sensors Network. This protocol porposes reliable messages delivery between a source node and its respective base station. It is aims to detect and control congestion through the differentiation between transmission error losses and buffer overflow. In case of losses, it offers recovery in two reliability levels and, in case of congestion, provides explicit signaling for break and return on data transmission.

We have a significant decrease in definitive loss message probability through distributed responsibility of temporary message storage between two adjacent nodes (reliability level 2). Moreover, distributed message storage demonstrates protocol robustness during periods of disconnection, prior to an ACK confirmation.

CTCP makes no restriction on the protocol lower levels to be used.

The base station takes the decisions that depend on the application requirements knowledge (reliability required) and control parameters with central characteristics (ID connection). Furthermore, the functions of controlling congestion, implementation of the reliability and connections openness are distributed in WSN.

In future works, we will investigate the effect of increasing the number of hops in the generation of multiple ACKs in reliability model level 2. We will use metrics as the increase in the transmission and consumption buffer cost versus increased delivery probability.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: A survey. *Computer Networks*, 38:393–422, 2002.

[2] S. Heimlicher, R. Baumann, M. May, and B. Plattner. Saft: Reliable transport in mobile networks. pages 477–480, Vancouver B.C., Canada, Oct. 2006. IEEE MASS 2006.

[3] 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*, pages 449–454, Houston, TX, USA, 2005.

[4] 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*, pages 351–365. ACM, 2007.

[5] R. Milner. *A Calculus of Communicating Systems*. Springer Verlag, 1980.

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

[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*, pages 102–112, Anchorage, Alaska, USA, 2003.

[8] C. Wang, K. Sohraby, 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*, volume 1, pages 422–426, Arkansas Univ., Fayetteville, AR, USA, May 2005.