ARTICLE IN PRESS

Performance Evaluation ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at SciVerse ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

An analytical model for transport layer caching in wireless sensor networks

Nestor Michael C. Tiglao [a,c,*], António M. Grilo [a,b]

[a] INESC-ID/INOV, Rua Alves Redol, No. 9, 1000-029 Lisboa, Portugal
[b] Instituto Superior Técnico, Avenida Rovisco Pais, 1, 1049-001 Lisboa, Portugal
[c] EEEI, University of the Philippines, Diliman, Quezon City, 1101, Philippines

ARTICLE INFO

Article history:
Received 31 March 2011
Received in revised form 19 September 2011
Accepted 24 December 2011
Available online xxxx

Keywords:
Wireless sensor networks
Reliable transport protocol
Medium access control
Intermediate caching
IEEE 802.15.4

ABSTRACT

Reliable transport protocols have traditionally been designed to perform end-to-end error control transparently to the intermediate nodes (e.g., TCP). However, the resource constraints featured by Wireless Sensor Network (WSN) require a different paradigm where intermediate nodes are able to cache packets, retransmitting them on-demand in order to avoid incurring on costly end-to-end retransmissions. This paper presents an analytical model of end-to-end delivery cost for WSN reliable transport with intermediate caching. The model calculates the cost as the total number of physical layer transmissions using a probabilistic formulation that has been validated through network simulation. Although the model is based on a specific transport protocol (DTSN), the addressed mechanisms are more generic, allowing it to be easily adapted to other WSN transport protocols that also feature intermediate caching. Numerical results confirm the improved efficiency introduced by a transport layer with intermediate caching in comparison with end-to-end approaches that are based exclusively on MAC layer reliability. Different cache partitioning policies were tested, and it is shown that cache partitioning policies should take into account the network conditions experienced by concurrent flows, namely the status of the radio links and the flow lengths.

1. Introduction

Wireless Sensor Networks (WSNs) are infrastructureless networks that rely on large numbers of autonomous sensor nodes to establish routing paths to the systems that must be fed with the sensorial data. One or more sink nodes typically act as gateways to the external systems. The energy efficiency requirements of WSNs demand the use of low power radio technologies, usually leading to multihop data transmission from the sensor nodes to the sink nodes and vice versa. Since most WSN standards such as IEEE 802.15.4 [1] specify operation in unlicensed industrial, scientific, and medical (ISM) bands, radio links become usually more error-prone than in other wireless networks (e.g., WLANs), which use higher transmission power. The end-to-end error rate increases even more with the number of hops, usually requiring error recovery mechanisms to keep it within acceptable bounds. The transport layer is usually responsible for end-to-end error recovery, being widely accepted that transport protocols which rely purely on end-to-end retransmission (e.g., TCP) are energy-inefficient. Transport protocols such as DTC [2] and DTSN [3,4] employ intermediate caching in order to decrease the cost of end-to-end delivery. In these protocols, intermediate nodes are allowed to store and retransmit packets that are missing at the destination.

* Correspondence to: INESC-ID, Room 535, Rua Alves Redol, No. 9, 1000-029 Lisboa, Portugal. Tel.: +351 21 3100226; fax: +351 21 3145843.
E-mail addresses: nestor.tiglao@inesc-id.pt, nmctiglao@yahoo.com (N.M.C. Tiglao), antonio.grilo@inov.pt (A.M. Grilo).

0166-5316/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2011.12.002

Distributed Transport for Sensor Networks (DTSN) is a simple protocol developed with the objective of providing a more efficient transport-layer service in WSNs compared with TCP. It has the following main features: (1) reliable transmission of block-oriented data, with selective repeat ARQ; (2) signaling traffic controlled by the sender (the sender explicitly issues packet confirmation requests); (3) WSN nodes are able to cache packets at intermediate nodes so that end-to-end retransmissions are minimized.

This paper presents an analytical model of WSN transport with intermediate caching, allowing the performance to be predicted in terms of the number of physical transmissions required to guarantee the end-to-end delivery of data packets. Although the details of the model are inspired by the DTSN protocol, its scope spans WSN transport with intermediate caching in general. The conclusions that are drawn from the numerical results regarding the design of cache partitioning policies will still hold in other similar protocols, for example DTC. As far as the authors know, this is the first time that a comprehensive and integrated model is proposed to analyze the effects of caching on the performance of the WSN transport layer, considering the impact of cache size, hop length, packet error rates (for both data and control packets) and concurrent flows.

The rest of this paper is organized as follows. Section 2 presents the relevant related work. Section 3 provides an overview of the DTSN protocol. This is followed by Section 4, where the proposed analytical model is presented. Section 5 presents some numerical and simulation results. Finally, Section 6 concludes the paper.

2. Related work

Since the list of reliable transport protocols proposed for WSNs is very extensive, this section shall only cover the ones that are most directly related to the scope of the paper (i.e., protocols that employ intermediate caching to minimize end-to-end retransmissions), as well as the main analytical models of WSN end-to-end delivery cost proposed in the literature. The interested reader is referred to [5] for a more extensive survey on WSN transport protocols.

2.1. Reliable WSN transport protocols with intermediate caching

Reliable Multi-Segment Transport (RMST) [6] offers two simple services: data segmentation/reassembly and guaranteed delivery using a NACK-based ARQ mechanism. RMST can operate end-to-end or in a store-and-forward mode where intermediate nodes recover all the fragments of a block before relaying them forward. Since in this case the data blocks are reconstructed at each hop, RMST requires significant memory resources to be available at individual nodes and is not designed to cache packets in the intermediate nodes probabilistically.

Pump Slowly Fetch Quickly (PSFQ) [7] is a protocol primarily designed for downstream multicast dynamic code update, though it can also be configured for unicast communication. Regarding intermediate caching, data is reconstructed at each hop, just like in RMST (see above). While this makes sense for dynamic code update (i.e., each node must get all the executable code fragments), that can be very limiting for other applications (e.g., audio or image transmission) since it poses significant requirements on node storage capabilities.

GARUDA [8,9] is focused on sink-to-sensor reliability like PSFQ. It uses a core architecture, where every third node is a core node, serving as caching node and loss recovery server. Nodes use implicit multiple NACKs to recollect missing fragments, where the NACK is the sequence number of the last message ID the node has received thus far. Error recovery is performed in two phases. During the first phase, a core node detecting missing packets requests them from its upstream core node. After recovering all the missing packets, the second phase starts, in which the core node acts as recovery server for the non-core nodes in its vicinity. To protect against lost full messages, Garuda uses a special Wait-for-First-Packet (WFP) pulse, which is a small finite series of short duration pulses repeated periodically. Sensor nodes along the path to the intended recipients, upon reception of the pulses also start pulsing and eventually all sensor nodes along the path are informed that a packet is to be received. After pulsing for a finite duration, the sink transmits the first fragment. If a node receives the first fragment, it stops pulsing the WFP and broadcasts the first fragment. Therefore, WFP serves as an implicit NACK for the first fragment, while termination of WFP pulsing corresponds to an ACK. As first messages can store the size of the data that is going to be transferred, reliable transfer of the first fragment can solve the lost last fragment problem. GARUDA supports out-of-order packet delivery, which allows it to reduce the delivery delay.

Distributed TCP Caching (DTC) [2] enhances TCP in order to make it more efficient in WSNs. Its guaranteed delivery service is similar to that of DTSN. DTC improves the transmission efficiency by compressing the headers and by using cache at selected intermediate nodes. DTC is fully compatible with TCP, leaving the endpoints of communication unchanged—it only requires changes in the logic of intermediate nodes. Although the paper only considers caching a single segment, this significantly improves the efficiency of end-to-end delivery, minimizing the energy spent with retransmissions. The caching mechanism may also be easily extended to use more than one segment per node. The authors also propose to use the TCP SACK option in order to optimize the use of the cache. In this proposal, the SACK block is used to carry information about segments in cache, allowing nodes farther from the destination to free their cache entries in case a node that is closer already has the segment in cache. Some improvements have been proposed to the original DTC protocol. In [10], a multipath scheme between sender and receiver allows the TCP session to be rerouted when intermediate nodes fail. In [11] the focus is rather the MAC-transport cross-layer optimization, assuming a CSMA/CA MAC with ARQ. While DTC always tries to cache more

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

recent segments, the new scheme allows to maintain an older segment already in cache with 50% probability. The paper also proposes an alternative method to estimate the round-trip-time.

Wisden [12] is a sensor-to-sink wireless structural data acquisition system that incorporates data synchronization and data compression algorithms besides reliable transport. The latter can be performed hop-by-hop, in which case intermediate nodes keep lists of missing packets and cache recently forwarded packets. Missing packets are requested from the previous node using a NACK mechanism. Since the amount of cache at each node is limited, there may not be enough space to store all the packets in cache. Hence, an end-to-end recovery scheme is used to ensure that all packets are delivered. The sink node keeps track of missing packets and requests them using the same NACK scheme used for hop-by-hop. It is assumed that the sink node has no memory constraints, being able to store all out-of-order packets until the gaps are filled.

Rate-Controlled Reliable Transport (RCRT) [13], developed later by the same research lab, is another reliable sensor-to-sink transport protocol, which includes congestion control and explicit rate adaptation functions. Although RCRT defines a purely end-to-end NACK-based retransmission mechanism, [14] proposes an hop-by-hop variant, where each node along the path keeps a packet cache and a list of missing packets per-flow. Whenever missing packets are detected, the respective sequence numbers are included in a NACK packet, which is sent to the previous node. In case an intermediate node receives a NACK and has any of the requested packets in cache, it retransmits them toward the sink and forwards the updated NACK toward the source. Cached packets are deleted once they are confirmed by a positive ACK.

Reliable Transport with Memory Consideration (RTMC) [15] integrates hop-by-hop error recovery with flow control to prevent cache overflow at intermediate nodes. A connection-oriented approach is followed since cache is explicitly reserved per-flow upon reception of a special control packet in the beginning of the session, being also explicitly freed at the end of the session. Each node processes data packets coming from both the previous hop and the next hop. Packets from the previous hop are stored in cache and transmitted to the next hop as soon as possible. Packets relayed by the next hop are used as implicit ACKs to free space in the local cache, as well as to know the amount of cache that is currently free at the next hop. A node will only relay data packets when there is free room at the next hop. Packets are explicitly requested from the previous hop when detected as missing or when the cache is empty. No end-to-end reliability scheme is considered.

Actor-Actor Reliable Transport Protocol (A^2RT) [16] seeks to provide real time and reliable data delivery between partitioned resource-rich actor nodes whose connectivity is bridged by resource-constrained sensor nodes. Caching of packets at actor nodes improves the performance, compared with end-to-end reliability. Cross-layer interaction with the routing protocol allows a transport wrapper entity to divide the end-to-end path into multiple segments between partitioned actor nodes, minimizing the number of sensor nodes that are used to bridge two consecutive actor nodes. In each segment, a reliable transport protocol is used to guarantee actor-to-actor delivery until the destination is reached. An end-to-end wrapper session is maintained between the source and the final destination in order to assure end-to-end delivery.

The Distributed Caching for Sensor Networks (DTSN) protocol is more directly related to the present paper and shall be described separately in Section 3.

2.2. Analytical models of end-to-end reliability in WSNs

In [17], four algorithms are proposed, which improve the end-to-end delivery ratio of single packets based on hop-by-hop retransmissions. The Hop-by-Hop Reliability (HHR) and Hop-by-Hop Reliability with Acknowledgments (HHRA) algorithms use single paths, where HHR always sends the packet a fixed number times, while HHRA bases its retransmission decisions on the reception of acknowledgments. The other two algorithms, Hop-by-Hop Broadcast (HHB) and Hop-by-Hop Broadcast with Acknowledgments (HHBA) are similarly related, but employ braided paths that are built by means of packet broadcasting at each hop. An analytical model is provided, allowing the comparison between the proposed protocols. A multipath reliable delivery mechanism designated Reliable Information Forwarding Using Multiple Paths (ReInForM) is proposed by the same authors in [18], together with an analytical model. In both papers the reliability model is purely end-to-end and intermediate caching is not considered.

In [2], the authors of DTC propose a simple analytical model in order to allow a rough comparison between DTC with a fully-reliable hop-by-hop retransmission scheme. However, this analytical model does not allow the calculation of total MAC overhead and assumes that the packet is always cached at the last node before a transmission failure takes place. Concurrent flows are also not considered in the model.

In [16] simple calculations are included to show that dividing an end-to-end path into segments between caching nodes is more efficient than performing pure end-to-end delivery. However, no model is presented for multi-flow scenarios or even memory-constrained caching nodes.

Differently from the above models, the one presented in this paper addresses scenarios with concurrent flows and it is also more generic, focusing on the caching mechanism. In fact it can be easily adapted to map the DTC functionality, namely when the SACK option is used (see above).

3. DTSN overview

The basic DTSN specification [3,4] was thought for critical data transfer requiring end-to-end full reliability, in the fashion of TCP. However, for sake of improving energy efficiency in WSNs, DTSN employs a Selective Repeat Automatic

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

Repeat reQuest (SR-ARQ) using negative acknowledgments (NACK). Positive acknowledgment packets (ACK) are also used to prevent the situations where the complete message or its last packet is lost (which cannot be detected solely based on NACKs). Both NACKs and ACKs are to be sent by the receiver only upon request by the sender (Explicit Acknowledgment Request-EAR), which can be piggy-backed in data packets.

In DTSN, a session is a source/destination relationship univocally identified by the tuple ⟨*source address, destination address, application identifier, session number*⟩, designated the session identifier. Within a session, packets are sequentially numbered. The Acknowledgment Window (AW) is defined as the number of packets that the source transmits before generating an EAR. The output buffer at the sender works as a sliding window, which can span more than one AW. Its size depends on the specific scenario, namely on the memory constraints of individual nodes.

In order to minimize end-to-end retransmissions, the intermediate nodes are able to cache a number of packets. Upon interception of a NACK, in case they find any of the requested packets in cache, they retransmit those packets to the receiver. After deleting the respective sequence numbers from the NACK, the intermediate node allows the modified NACK packet to resume its walk toward the source.

4. Analytical model

Let $G = (V, E)$ be a directed graph representing the network logical topology, with $V = \{1, \ldots, N\}$ being its set of vertices/nodes (with $N = |V|$), E its set of edges/links. Let P be the link reliability matrix of G, where each entry $p^{\lambda}_{i,j}$ is the function that calculates the physical error probability of a packet λ in link $(i, j) \in E$, or 1 if $(i, j) \notin E$. The variable λ represents a packet (e.g., DATA, ACK, or NACK) and abstracts all intrinsic factors that may affect the packet error probability (e.g., coding, bit rate, preamble, header and payload lengths, etc.), while the $p^{\lambda}_{i,j}$ take into account all extrinsic phenomena that might be a cause of packet error (e.g., interference, attenuation, fading, collisions, etc.). A Binary Symmetric Channel (BSC) is assumed, according to which all errors are considered independent. This is a relaxation of the model, since the error patterns in real networks may be bursty.

Let Φ represent the set of flows,[1] where each $\varphi_i \in \Phi$, $i = \{1, \ldots, F\}$, is a linear sub-graph of G with hop length l_i. The proposed analytical model is focused on the overhead introduced by the end-to-end reliable transport protocol, leaving aside other transport-related aspects such as congestion control, as well as accurate throughput and delay calculation. It is assumed that the flow paths are previously set-up by a routing protocol operating transparently beneath the transport layer. All intermediate nodes in these paths are potential caching nodes provided that their cache size is greater than 0. The route decision logic and overhead associated with the routing protocol are out-of-scope of this model. The model was designed to allow the computation of the following technology-independent metrics for a given transport flow φ_i:

- $\mathbf{PoD_i(t)}$: Probability of end-to-end packet delivery after t transmission iterations. This metric is related to the packet delivery delay and throughput.
- $\mathbf{c_i}$: Average end-to-end delivery cost, defined as the average number of physical layer transmissions required to deliver the packet from the source to the destination considering no limit for the number of transport retransmission attempts. This metric can be provided as input to a technology-dependent energy consumption model, allowing the computation of an equivalent energy cost. The metric is also indirectly related to throughput and interference.

It was an objective of this model to be as generic as possible so that the results can be applied to other WSN transport protocols. As such, the model captures only the following DTSN features[2]: (1) SR-ARQ based on NACK message feedback; (2) caching at the intermediate nodes, with the cache at a given node being shared among concurrent flows that cross that node; and (3) windowed transmission with feedback provided at the end of each AW.

Besides probabilistic caching protocols like DTSN and DTC, store-and-forward schemes like the one used in RMST can also be captured by the proposed model when suitably parameterized. In this case, each flow should be considered as a sequence of shorter sub-flows, with the end-to-end delivery cost being given by the sum of the delivery costs of the sub-flows.

The proposed model can be divided into two components: a link-layer component and a transport layer component. A summary of the mathematical notations is provided in Table 1.

4.1. Link-layer component

Assuming a BSC and assuming that the MAC layer uses an ARQ scheme to improve link reliability, the effective packet error probability of link (i, j) can be computed according to the following equation:

$$\pi^{\lambda}_{i,j} = (p^{\lambda}_{i,j})^{r+1} \tag{1}$$

[1] The terms *transport session* and *transport flow* shall henceforth be used interchangeably in this paper.

[2] The Explicit Acknowledgment Request (EAR) and positive acknowledgment (ACK) packets were left out of the presented model for space reasons and in order to make the model more generic. In fact, a timeout at the receiver can be used instead of EAR notifications, while in steady-state operation the ACK notification for a given AW can be replaced by the first NACK belonging to the ensuing AW, decreasing the overhead impact of the ACK as time progresses. However, the authors have also developed a complete DTSN model that considers EAR and ACK packets, which are implemented as options in the developed software tool.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

Table 1
Summary of mathematical notations.

Notation	Meaning
N	Total number of nodes in the network.
F	Total number of flows in the network.
F_n	Total number of flows that cross node n.
λ	Represents a packet (i.e., DATA, ACK, or NACK)
$p_{i,j}^{\lambda}$	Physical error probability associated with the transmission of packet λ over link (i, j).
$\pi_{i,j}^{\lambda}$	Effective error probability associated with the transmission of packet λ over link (i, j), taking into account the maximum number of MAC retries.
$\varphi_i(j)$	Node that is located j hops away from the source of flow φ_i. In accordance with this definition, $\varphi_i(0)$ and $\varphi_i(l_i)$ are the source and the destination nodes, respectively.
l_i	Hop length of the end-to-end path of flow φ_i.
ω_i^n	Cache partitioning weight assigned to flow φ_i at node n.
ρ_i^n	Fraction of cache assigned to flow φ_i at node n.
χ_i^n	Actual probability that a packet from flow φ_i is found in the cache of node n.
$\gamma_{i,j}^{S,\lambda}$	Average cost of the successful transmission of packet λ over link (i, j).
$\gamma_{i,j}^{F,\lambda}$	Average cost of the failed transmission of packet λ over link (i, j).
$PeP_i(t, h)$	Probability of effective progress of a packet from flow φ_i to $\varphi_i(h)$ in iteration $t \in [0 \cdots + \infty[$.
$PoD_i(t)$	Probability of delivery of a packet from flow φ_i to $\varphi_i(l_i)$ in iteration $t \in [0 \cdots + \infty[$. This is equivalent to $PeP_i(t, l_i)$.
$c_i(t)$	Expected accumulated cost of transmission of a packet from flow φ_i up to iteration t.
c_i	Expected cost of end-to-end delivery of a packet from flow φ_i. This is equivalent to $\lim_{t\to\infty} c_i(t)$.

where r is the MAC retry limit. In other words, the link-layer transmission can only fail if all the physical transmission attempts for the data packet fail. It should be noted that the transmission of the MAC layer acknowledgments (MACK) is not relevant to calculate this probability, since even if no MACKs are received by the sender, the data packet may have been correctly received during one of the attempts. Nevertheless, it must be taken into account when calculating the transmission cost.

The link-layer transmission of a packet incurs expected success and failure costs $\gamma_{i,j}^{S,\lambda}$ and $\gamma_{i,j}^{F,\lambda}$, which are defined as the average number of physical layer transmissions (including MACK transmissions) made in case of a successful or a failed link transmission, respectively. These costs are calculated as follows[3]:

$$C_1 = \sum_{k=1}^{r}\sum_{m=0}^{k}\binom{k-1}{m}\left[\left(1-p_{i,j}^{DATA}\right)p_{i,j}^{MACK}\right]^m\left(p_{i,j}^{DATA}\right)^{k-1-m}\left[\left(1-p_{i,j}^{DATA}\right)\left(1-p_{i,j}^{MACK}\right)\right](m+k+1)$$

$$C_2 = \sum_{m=0}^{r}\binom{r}{m}\left[\left(1-p_{i,j}^{DATA}\right)p_{i,j}^{MACK}\right]^m\left(p_{i,j}^{DATA}\right)^{r-m}\left(1-p_{i,j}^{DATA}\right)(m+r+2)$$

$$C_3 = \sum_{m=1}^{r}\binom{r}{m}\left[\left(1-p_{i,j}^{DATA}\right)p_{i,j}^{MACK}\right]^m\left(p_{i,j}^{DATA}\right)^{r-m}p_{i,j}^{DATA}(m+r+1)$$

$$\gamma_{i,j}^{S,DATA} = \frac{C_1 + C_2 + C_3}{1-\pi_{i,j}^{DATA}} \tag{2}$$

$$\gamma_{i,j}^{F,DATA} = r+1. \tag{3}$$

The numerator of $\gamma_{i,j}^{S,DATA}$ comprises the following terms:

1. The cost of link-layer success when less than $r+1$ data transmission attempts are needed, i.e., the last attempt is confirmed by a successful MACK, which ends the MAC ARQ procedure before the maximum number of attempts is exceeded. As to the other attempts, either the data packet fails or the respective MACK fails.
2. The cost of link-layer success when $r+1$ data transmission attempts are made, with the last one being successful (the fate of the last MACK is not relevant). As to the other attempts, either the data packet fails or the respective MACK fails.
3. The cost of link-layer success when $r+1$ data transmission attempts are made but the last one fails. As to the other attempts, the data transmission is successful at least once, but the MACK transmissions always fail.

Regarding $\gamma_{i,j}^{F,DATA}$, it suffices to say that link-layer transmission failure can only occur when all the $r+1$ data packet transmission attempts are unsuccessful, hence leading to the absence of MACK transmissions.

[3] In the following formulation it is assumed that each DATA, NACK and MACK physical transmission has a cost equal to 1. However, the model can be easily adapted to consider different costs (e.g., packet length, etc.).

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

4.2. Transport-layer component

The objective of the transport-layer model is to calculate $PoD_i(t)$ and c_i. As already explained, the role of the transport protocol is to guarantee that each data packet generated at the source of a flow arrives sooner or later at the destination. In case the packet is lost in transit at some point along the path φ_i, the destination detects this occurrence and sends a NACK back to the source following the reverse path φ_i^{-1}.

In case the NACK reaches the source, it retransmits the packet. However, since there is caching at intermediate nodes, if the NACK is intercepted by an intermediate node which has the packet in cache, it immediately retransmits the requested packet and deletes the respective sequence number from the NACK. The latter is then allowed to travel again toward the source, carrying the remainder of the list of missing sequence numbers. If the NACK is lost in transit, sooner or later another NACK packet is issued by the destination.

For each node $n \in V$, the total cache size C_n is divided among the different flows that cross the node. Let $\omega_i^n \geq 0$ be a weight related to the fraction of cache at node n that is assigned to flow φ_i and let $\Omega = \{\omega_n^i\}$. The actual fraction is given by the normalized weight $\rho_i^n = \frac{\omega_i^n}{\sum_{j=1}^{F_n} \omega_j^n}$. It is assumed that the source and destination of a flow always keep all packets from that flow in a separate memory, not interfering with the caching mechanism. AW_i is the size of the transmission window of the source of flow i.

A packet that belongs to flow i can only be cached in the fraction of cache assigned to flow i, whose size is equal to $\rho_i^n \times C_n$. Consequently, the caching probability for a packet that belongs to flow i at node n is given by[4]:

$$\chi_i^n = \min\left(1, \rho_i^n \times \frac{C_n}{AW_i}\right). \tag{4}$$

After each retransmission request, assuming that the NACK is successful, the data packet is retransmitted from the node that is closest to the destination and has the packet in its memory (in the worst case this is the source node). Each transport-layer transmission attempt constitutes an iteration, and the position of the transmitting node (i.e., its hop distance from the source) in iteration $t+1$ constitutes the state of the system after iteration t, being designated the effective progress of the packet up to iteration t. It should be noted that the latter means not only that the packet was successfully transmitted to that node, but also that it is stored in its memory and none of the nodes closer to the destination (including the destination itself) is currently storing it.

The set of possible system states is equal to the number of nodes along the path of the flow, including the source and the destination. In each iteration, the transition probabilities only depend on the present state, on the packet error probability matrix $\left(\prod = [\pi_{i,j}]\right)$ and on the set of caching probabilities $\left(X = \{\chi_i^n\}\right)$. From this follows that the progress of the packet along the path of a flow φ_i can be modeled as an absorbing Markov chain whose initial state is $\varphi(0)$ and the final or absorbing state is $\varphi_i(l_i)$. In order to simplify the model, the following assumptions are made:

1. Once an intermediate node decides to cache or not a packet, that decision remains unchanged until the AW is completely received by the destination.
2. For each flow, the transmission window spans only one AW.
3. For each flow the forward and reverse paths are stable and symmetric.

Assumption 1 above means that, from one iteration to the next, the system either maintains its state or goes to a state where the packet is closer to the destination, which greatly simplifies the model. Assumption 2 further simplifies the model putting aside the situation where NACK packets carry sequence numbers belonging to different AWs, which would cause the NACK overhead to be divided by more than one AW. Assumption 3 means that the NACKs are forwarded through the reverse path of the data packets, so that the system gets effective benefit of intermediate node caching.

For each iteration t there is a probability that the packet has progressed to node $\varphi_i(h)$, $0 \leq h \leq l_i$, the probability of effective progress of the packet, $PeP_i(t, h)$, which is calculated by solving for the effective probabilities resulting from all the possible outcomes of a given iteration. $PeP_i(t, h)$ is calculated according to Eq. (5). The reasoning behind the expression can be explained by considering the most complex situation, i.e., $t > 1, 0 < h < l_i$. In this case, $PeP_i(t, h)$ is the sum of the probabilities associated with the following possible situations:

i. The current status of the system is $j < h$. In this case, $PeP_i(t, h)$ is equal to the product of the following probabilities (these probabilities are represented by the respective numbers in Fig. 1):
   1. The packet is in node j at the end of the previous iteration: $PeP_i(t-1, j)$.
   2. The NACK packet that resulted from iteration $t-1$ was able to reach node j: $\prod_{d=j}^{l_i-1}\left(1 - \pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right)$.
   3. The data packet was able to reach node h: $\prod_{e=j}^{l_i-1}\left(1 - \pi^{DATA}_{\varphi_i(e),\varphi_i(e+1)}\right)$.

[4] This expression assumes an optimal use of the cache. The software tool developed by the authors can also consider the possibility that each packet is subject to an independent Bernoulli trial.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

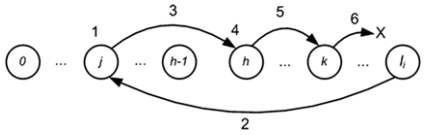


Fig. 1. Example of $PeP_i(t, h)$ with $t > 1$, $0 < h < l_i$.

4. The data packet was cached at node h: χ_i^h.
5. The data packet may have been successfully transmitted to some node $k < l_i$, but was not cached along the way: $\prod_{f=h}^{k-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)$.
6. Then, the ensuing transmission from node k has failed: $\pi^{DATA}_{\varphi_i(k),\varphi_i(k+1)}$.

ii. The current status of the system is already h and the NACK packet was successfully received. The reasoning is similar to the previous case, but the packet is assumed to be already cached at h.

iii. The current status of the system is already h but the NACK packet was lost along the way.

$$
\begin{aligned}
&PeP_i(t,h) = && (5)\\
&t=0, h=0 \quad 1\\
&t=0, h\neq 0 \quad 0\\
&t=1, h=l_i \quad \prod_{j=0}^{l_i-1}\left(1-\pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\right)\\
&t=1, h=0 \quad \sum_{j=0}^{l_i-1}\left[\left(\prod_{f=0}^{j-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)\right)\cdot\pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\right]\\
&t=1, 0<h<l_i \quad \prod_{e=0}^{h-1}\left(1-\pi^{DATA}_{\varphi_i(e),\varphi_i(e+1)}\right)\cdot\chi_i^h\cdot\sum_{j=h}^{l_i-1}\left[\prod_{f=h}^{j-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)\right]\cdot\pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\\
&t>1, h=l_i \quad PeP_i(t-1,l_i)+\sum_{j=0}^{l_i-1}\left[PeP_i(t-1,j)\cdot\left(\prod_{d=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right)\right)\cdot\left(\prod_{e=j}^{l_i-1}\left(1-\pi^{DATA}_{\varphi_i(e),\varphi_i(e+1)}\right)\right)\right]\\
&t>1, h=0 \quad PeP_i(t-1,0)\cdot\sum_{j=0}^{l_i-1}\left[\left(\prod_{f=0}^{j-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)\right)\cdot\pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\right]\\
&t>1, 0<h<l_i \quad \sum_{j=0}^{h-1}\left[\begin{array}{l}PeP_i(t-1,j)\cdot\left(\prod_{d=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right)\right)\cdot\left(\prod_{e=j}^{l_i-1}\left(1-\pi^{DATA}_{\varphi_i(e),\varphi_i(e+1)}\right)\right)\cdot\chi_i^h\cdot\\ \quad\sum_{k=h}^{l_i-1}\left[\left(\prod_{f=h}^{k-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)\right)\cdot\pi^{DATA}_{\varphi_i(k),\varphi_i(k+1)}\right]\end{array}\right]\\
&\quad + PeP_i(t-1,h)\cdot\left(\prod_{d=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right)\right)\\
&\quad\cdot\sum_{k=h}^{l_i-1}\left[\left(\prod_{f=h}^{k-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\left(1-\chi_i^{f+1}\right)\right)\cdot\pi^{DATA}_{\varphi_i(k),\varphi_i(k+1)}\right]\\
&\quad + PeP_i(t-1,h)\cdot\left(1-\prod_{d=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right)\right).
\end{aligned}
$$

The $PeP_i(t, h)$ function has the following properties:

1. $\left[0<\pi_{\varphi_i(j),\varphi_i(j+1)}<1, \forall j\in[0,\ldots,l_i-1]\right]\wedge(t_2>t_1)\Leftrightarrow PeP_i(t_2,h)>PeP_i(t_1,h)$
2. $\left[\pi_{\varphi_i(j),\varphi_i(j+1)}<1, \forall j\in[0,\ldots,l_i-1]\right]\Leftrightarrow\lim_{t\to+\infty}PeP_i(t,l_i)=1$
3. $\exists j\in[0,\ldots,l_i-1]:\pi_{\varphi_i(j),\varphi_i(j+1)}=1\Leftrightarrow PeP_i(t,l_i)=0$
4. $\sum_{h=0}^{l_i}PeP_i(t,h)=1, \forall t\geq 0$.

The cost of progress of a packet in each iteration can be calculated based on $PeP_i(t, h)$. In order to simplify the cost expression, the auxiliary function $CDP_i(x, y)$ is defined, representing the expected cost of progress of a packet from node $\varphi_i(x)$

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

to node $\varphi_i(y)$ but not beyond $\varphi_i(y)$. This assumes that the packet coming from $\varphi_i(x)$ reaches $\varphi_i(y)$ but is lost somewhere between $\varphi_i(y)$ and $\varphi_i(l_i)$, not being cached anywhere between $\varphi_i(y)$ and $\varphi_i(l_i)$.

$$CDP_i(x,y) = \sum_{k=y}^{l_i-1} \left[\begin{array}{c} \left(\prod_{f=y}^{k-1} \left(1 - \pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right) \left(1 - \chi_i^{f+1}\right) \right) \cdot \pi^{DATA}_{\varphi_i(k),\varphi_i(k+1)} \cdot \\ \left(\sum_{f=x}^{k-1} \gamma^{S}_{\varphi_i(f),\varphi_i(f+1)} + \gamma^{F}_{\varphi_i(k),\varphi_i(k+1)} \right) \end{array} \right]. \tag{6}$$

The cost calculated by $CDP_i(x,y)$ is equal to the sum of the costs of successful link transmissions from $\varphi_i(x)$ to $\varphi_i(y)$, plus the cost of successful link transmissions beyond $\varphi_i(y)$, plus the cost of a failed link transmission somewhere between $\varphi_i(y)$ and $\varphi_i(l_i)$. Since this failed attempt can occur anywhere between $\varphi_i(y)$ and $\varphi_i(l_i)$, the cost of the possible situations is weighted by the respective probability.

Furthermore, two auxiliary functions $CNS_i(h)$ and $CNF_i(h)$ are defined, which respectively represent the average cost of successful and failed NACK delivery to node $\varphi_i(h)$:

$$CNS_i(h) = \frac{\sum_{j=1}^{h} \left[\prod_{m=j}^{h-1} \left(1 - \pi^{NACK}_{\varphi_i(m+1),\varphi_i(m)}\right) \cdot \pi^{NACK}_{\varphi_i(j),\varphi_i(j-1)} \cdot \left(\sum_{m=j}^{l_i-1} \gamma^{S,NACK}_{\varphi_i(m+1),\varphi_i(m)} + \gamma^{F,NACK}_{\varphi_i(j),\varphi_i(j-1)} \right) \right] + \prod_{j=0}^{h-1} \left(1 - \pi^{NACK}_{\varphi_i(d+1),\varphi_i(d)}\right) \cdot \sum_{j=0}^{l_i-1} \gamma^{S,NACK}_{\varphi_i(j+1),\varphi_i(j)}}{AW_i} \tag{7}$$

$$CNF_i(h) = \frac{\sum_{j=h+1}^{l_i} \left[\prod_{m=j}^{l_i-1} \left(1 - \pi^{NACK}_{\varphi_i(m+1),\varphi_i(m)}\right) \cdot \pi^{NACK}_{\varphi_i(j),\varphi_i(j-1)} \cdot \left(\sum_{m=j}^{l_i-1} \gamma^{S,NACK}_{\varphi_i(j+1),\varphi_i(j)} + \gamma^{F,NACK}_{\varphi_i(j),\varphi_i(j-1)} \right) \right]}{AW_i}. \tag{8}$$

The $CNS_i(h)$ function assumes that the NACK was able to reach node $\varphi_i(h)$, which then forwarded it in the direction of the sender $\varphi_i(0)$. Between node $\varphi_i(h)$ and the sender, the NACK may have reached the sender or may have been lost in transit at some node $\varphi_i(j)$. $CNS(i,h)$ provides the average NACK cost weighted by the probability associated with the possible outcomes of the NACK packet. In any case, the cost counts the successful transmissions from the destination node $\varphi_i(l_i)$ to $\varphi_i(h)$. Finally, the cost is divided by AW_i, since each NACK is sent for each AW_i. The $CNF_i(h)$ assumes that the NACK was lost somewhere between $\varphi_i(l_i)$ and $\varphi_i(h)$, bearing some similarity with the first term of the numerator in Eq. (7). The cost of NACK failure is also divided by AW_i.

Based on the $PeP_i(t,h)$, $CDP_i(x,y)$, $CNS_i(h)$ and $CNF_i(h)$, the function $g_i(t,h)$ is defined as shown in Eq. (9), representing the average cost of progress to some node $\varphi_i(h)$ in iteration t multiplied by $PeP_i(t,h)$. Similar to the latter, the last case ($t > 0, 0 < h < l_i$) will be explained in greater detail. This equation has three terms, which represent the following situations:

i. The packet has previously progressed to node $\varphi_i(h)$ and the NACK does not reach this node. The cost is only related to the NACK transmissions at the MAC layer: $CNF(i,h)$.
ii. The packet has previously progressed to node $\varphi_i(h)$ — it is already cached there — and the NACK is able to reach this node. However, the retransmitted data packet does not reach the destination and the packet is not cached at nodes closer to the destination. The considered costs are the following:
   (a) Cost of the successful NACK: $CNS_i(h)$.
   (b) Cost of the data packet while it is successfully transmitted by the MAC layer beyond $\varphi_i(h)$, plus the cost of a failed attempt between $\varphi_i(h)$ and $\varphi_i(l_i)$: $CDP(h,h)$.
iii. The packet has previously progressed to node $\varphi_i(j)$, $0 < j < h$, and the NACK successfully reaches node $\varphi_i(j)$. The latter retransmits the data packet, which successfully reaches and is cached at node $\varphi_i(h)$—the conditional probability of this occurrence is equal to $\left(\prod_{f=j}^{h-1}(1 - \pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)})\right) \cdot \chi_i^h$. No nodes closer to the destination than $\varphi_i(h)$ cache the packet and the latter is lost before reaching the destination. The considered costs are the following:
   (a) Cost of the successful NACK: $CNS_i(j)$.
   (b) Cost of the data packet while it is successfully transmitted by the MAC layer up to node $\varphi_i(h)$ and from then on, until the failed attempt at some node between $\varphi_i(h)$ and $\varphi_i(l_i)$: $CDP(j,h)$.

$$g_i(t,h) = \begin{array}{ll} t = 0 & 0 \\ t = 1, h = l_i & \left[\prod_{j=0}^{l_i-1} \left(1 - \pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\right) \right] \sum_{j=0}^{l_i-1} \gamma^{S}_{\varphi_i(j),\varphi_i(j+1)} \\ t = 1, h = 0 & CDP(0,0) \end{array} \tag{9}$$

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

$$t = 1, 0 < h < l_i \quad \left(\prod_{j=0}^{h-1}\left(1-\pi^{DATA}_{\varphi_i(j),\varphi_i(j+1)}\right)\right)\cdot \chi_i^h \cdot CDP(0,h)$$

$$t > 1, h = l_i \quad \sum_{j=0}^{l_i-1}\left[PeP_i(t-1,j)\cdot\left(\prod_{f=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(f+1),\varphi_i(f)}\right)\right)\cdot\left(\prod_{f=j}^{l_i-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\right)\cdot \left(CNS_i(j)+\sum_{f=j}^{l_i-1}\gamma^{S}_{\varphi_i(f),\varphi_i(f+1)}\right)\right]$$

$$t > 1, h = 0 \quad PeP_i(t-1,0)\cdot CNF_i(0) + PeP_i(t-1,0)\cdot\left(\prod_{j=0}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(j+1),\varphi_i(j)}\right)\right)\cdot[CNS_i(0)+CDP(0,0)]$$

$$t > 1, 0 < h < l_i \quad PeP_i(t-1,h)\cdot CNF(i,h) + PeP_i(t-1,h)\cdot\left(\prod_{f=h}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(f+1),\varphi_i(f)}\right)\right)\cdot[CNS_i(h)+CDP(h,h)] + \sum_{j=0}^{h-1}\left[PeP_i(t-1,j)\cdot\left(\prod_{f=j}^{l_i-1}\left(1-\pi^{NACK}_{\varphi_i(f+1),\varphi_i(f)}\right)\right)\cdot\left(\prod_{f=j}^{h-1}\left(1-\pi^{DATA}_{\varphi_i(f),\varphi_i(f+1)}\right)\right)\cdot\chi_i^h\cdot[CNS_i(j)+CDP(j,h)]\right].$$

Based on $g_i(t,h)$ and taking into account the 4th property of $PeP_i(t,h)$, the average accumulated cost up to iteration t can now be calculated:

$$c_i(t) = \sum_{j=1}^{t}\sum_{h=0}^{l_i} g_i(t,h). \tag{10}$$

All the equations required to calculate the target metrics are now available:

$$\mathbf{PoD_{i(t)}} = PeP_i(t,l_i) \tag{11}$$

$$\mathbf{c_i} = \lim_{t\to\infty} c_i(t). \tag{12}$$

5. Numerical results

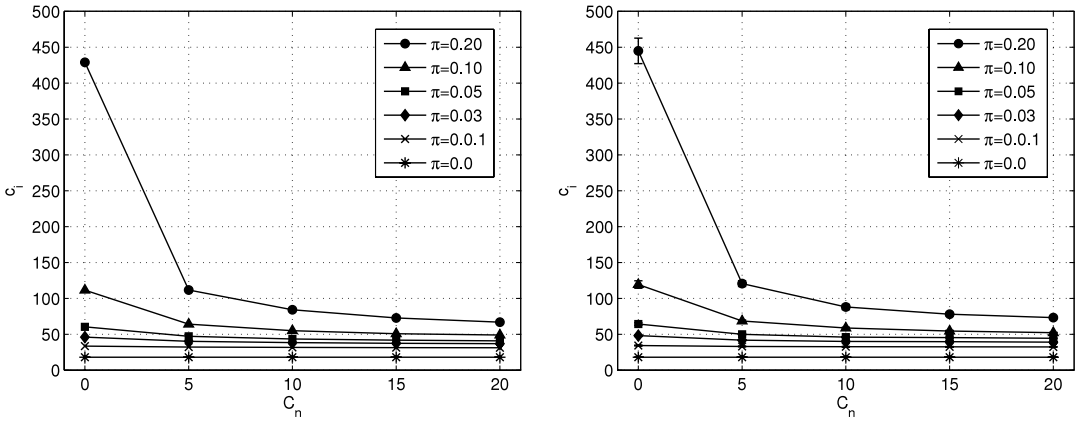
The presented analytical model was implemented in a software tool using the C programming language.[5] The tool assumes that network nodes are deployed in a grid topology. An input flow description file allows the user to set-up traffic flows, specifying the AW size and the flow path between sender and destination, which may comprise several segments featuring different $p^{\lambda}_{i,j}$ and explicitly assigned ω_i^n values.

In all scenarios it was assumed that the DATA, NACK and MACK packets are subject to a packet error Bernoulli trial based on configured $p^{\lambda}_{i,j}$ values. The latter are equal for all packet types, which is acceptable for the small DATA packet sizes commonly observed in WSN scenarios. This uniformity also allows for consistency in all scenarios that we have considered. However, since the simulation dynamics make the effective $p^{\lambda}_{i,j}$ dependent variables to a certain extent (e.g., due to collisions), the effective $p^{\lambda}_{i,j}$ obtained in the simulations are provided in subsequent tables, together with the respective simulation and analytical cost results where applicable. The adopted $\pi^{\lambda}_{i,j}$ value range is based on the measurements reported in [19] for IEEE 802.15.4, which point to maxima of around 0.035 for indoor scenarios and 0.1 for outdoor scenarios in the absence of IEEE 802.11b/g interference. With IEEE 802.11b/g interference, the worst-case $\pi^{\lambda}_{i,j}$ varies approximately between 0.20 (for 20-byte packets) and 0.80 (for 127-byte packets), even for very short communication distances (i.e., 5 m).

Validation of the analytical model was conducted with network simulation using an implementation of the DTSN protocol for the ns-2 [20] platform. Only the single flow validation results are presented. Regarding multiflow scenarios, the analytical model assumes that the performance of each flow can be calculated independently, once the cache partitioning weights are assigned. Experiments have shown that as far as the BSC characteristics of the communication channel are kept and there is no congestion in the network, this also holds true for the simulations. The considered wireless medium was 802.11b with RTS/CTS disabled. For each measurement, simulation runs were executed until the average fell within the 95% confidence interval. To determine the goodness-of-fit between the analytical and simulation results, we have calculated

[5] The authors will gladly provide the source code of the developed tool upon request by interested readers.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002



(a) Analytical. (b) Simulation.

Fig. 2. Single flow: cost c_i as a function of cache size C_n for different $\pi^{\lambda}_{i,j}$.

the transmission cost differences between points in the simulation and analytical curves, as well as the two-dimensional correlation coefficient R using the following formula:

$$R = \frac{\sum_m \sum_n (A_{mn} - \bar{A})(B_{mn} - \bar{B})}{\sqrt{\left(\sum_m \sum_n (A_{mn} - \bar{A})^2\right)\left(\sum_m \sum_n (B_{mn} - \bar{B})^2\right)}} \tag{13}$$

where A = matrix of analytical values, B = matrix of simulation values, $\bar{A} = mean(A)$, $\bar{B} = mean(B)$ and $-1 \leq R \leq 1$. A value close to 1 suggests that there is a positive relationship. A values close to -1 suggests that there is a negative relationship (anti-correlation). A value close to or equal to 0 suggests that there is no relationship.

5.1. Single flow scenario

The model is first applied to a single flow scenario. Figs. 2 and 3 present both the analytical and simulation results as a function of cache size and hop distance (in this case for configured $\pi^{\lambda}_{i,j} = 0.1$). The R coefficient values are 0.9998 and 0.9978, respectively. As can be seen, the cost c_i increases with the configured packet error probability ($\pi^{\lambda}_{i,j}$) and with the hop distance between sender and destination (l_i). On the other hand, it decreases as the cache size C_n increases. In Fig. 2, it also becomes obvious that the impact of caching relative to a pure end-to-end solution increases with the configured $\pi^{\lambda}_{i,j}$. The results have also shown that making $C_n > \frac{AW}{2}$ does not result in significant performance increase. The explanation for this behavior is that even if a single node is only able to cache one half of the AW, intermediate caching can achieve near optimal performance. This result is very important with respect to cache assignment optimization in scenarios featuring concurrent flows.

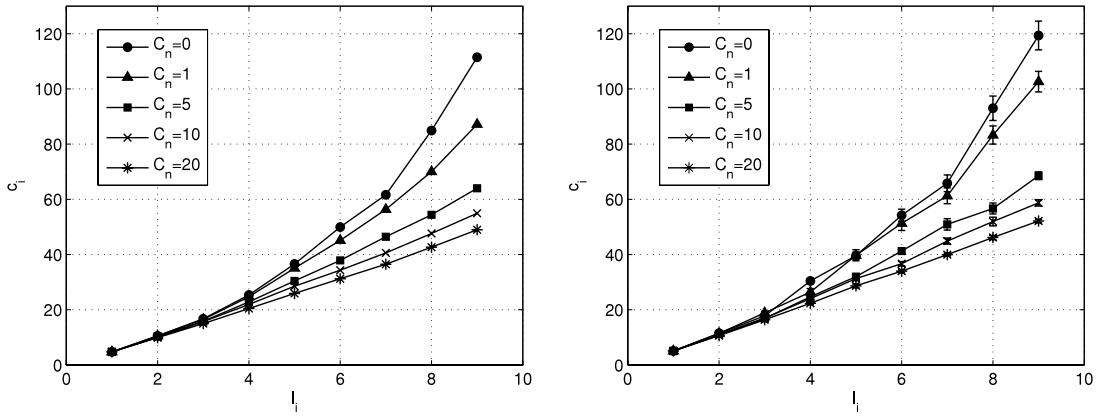
It is clear that there is an almost perfect match between the analytical and simulation results, with the difference between curve points never exceeding 10% of the simulated cost. It should be noted that the confidence intervals become larger precisely for high $\pi^{\lambda}_{i,j}$ and low cache size. This higher unpredictability of the cost in this region of the parameter space has a parallel in the analytical model, since the c_i becomes highly nonlinear. In other words, every small deviation from the average $\pi^{\lambda}_{i,j}$ in some of the links, may have a significant impact on cost. In Tables 2 and 3, the effective $\pi^{\lambda}_{i,j}$ average values are presented together with the respective simulation and analytical costs.

5.2. Single-flow transport-based vs. MAC-based end-to-end reliability

Having a huge link-layer retry limit assures a high-level of hop-by-hop reliability and minimizes the number of end-to-end retransmissions, albeit significantly slowing down the detection of topology changes. Since WSNs are usually considered to be fairly static, this should not represent a very significant drawback and would in principle avoid the need for caching. As such, the second set of results allows the comparison between the two solutions regarding the cost required to achieve a given value of $PoD_i(t)$.

Fig. 4 depicts the cost c_i as a function of the packet delivery probability $PoD_i(t)$ with $p^{\lambda}_{i,j} \approx 0.56$ (which makes $\pi^{\lambda}_{i,j} = 0.1$ when $r = 3$) and $l_i = 9$. The different values of $PoD_i(t)$ were achieved by varying t. As expected, c_i decreases as C_n increases, for fixed values of r and $PoD_i(t)$. On the other hand, when the caching mechanism is switched off ($C_n = 0$) and the $PoD_i(t)$ is fixed, c_i decreases as r increases until $r = 6$. For $r = 7$, c_i starts to increase again for the same values of $PoD_i(t)$. This

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002



(a) Analytical. (b) Simulation.

Fig. 3. Single flow: cost c_i as a function of the end-to-end hop length l_i for different cache size C_n and $\pi^{\lambda}_{i,j} = 0.10$.

Table 2
Effective physical packet error rates for the scenarios in Fig. 2.

Configured $p_{i,j}$	C_n	$p^{DATA}_{i,j}$	$p^{NACK}_{i,j}$	$p^{MACK}_{i,j}$	Simulation cost	Analytical cost	Difference (%)
0	0	0.000413985	0	0.000689849	18.0162	18.016156	0.00
	5	0.000413985	0	0.000689849	18.0162	18.016156	0.00
	10	0.000413985	0	0.000689849	18.0162	18.016156	0.00
	15	0.000413985	0	0.000689849	18.0162	18.016156	0.00
	20	0.000413985	0	0.000689849	18.0162	18.016156	0.00
0.3162	0	0.361943	0.290088	0.320186	34.22	33.542453	1.98
	5	0.361153	0.299205	0.319741	32.9331	32.186056	2.27
	10	0.360496	0.295042	0.321483	32.6519	31.682894	2.97
	15	0.3608	0.296275	0.320619	32.4919	31.400986	3.36
	20	0.360329	0.297958	0.321408	32.3775	31.263856	3.44
0.4162	0	0.472033	0.44866	0.416773	48.0725	46.007457	4.30
	5	0.468245	0.448932	0.414577	41.7606	40.184874	3.77
	10	0.46743	0.447753	0.415811	40.0056	38.292551	4.28
	15	0.466957	0.438087	0.414722	39.6494	37.337167	5.83
	20	0.467751	0.45057	0.416176	39.065	36.907109	5.52
0.4729	0	0.535493	0.506732	0.475102	64.2288	60.272568	6.16
	5	0.529396	0.510097	0.47354	49.98	47.183824	5.59
	10	0.525864	0.512887	0.474061	46.0712	43.368873	5.87
	15	0.52516	0.507655	0.475398	45.3294	41.734722	7.93
	20	0.525391	0.522376	0.472598	44.3956	40.776711	8.15
0.5623	0	0.626122	0.594481	0.565052	119.339	111.470291	6.59
	5	0.615283	0.600047	0.558556	68.5318	64.025591	6.58
	10	0.611471	0.596603	0.567429	58.7076	54.937569	6.42
	15	0.610517	0.593505	0.556674	54.5338	50.776135	6.89
	20	0.60858	0.592506	0.557424	52.1275	48.952637	6.09
0.6687	0	0.720526	0.692955	0.670632	444.878	428.935071	3.58
	5	0.709892	0.691581	0.669602	120.578	111.645297	7.41
	10	0.699485	0.681215	0.664193	87.9786	84.116048	4.39
	15	0.703867	0.682586	0.666652	77.9115	72.646702	6.76
	20	0.710014	0.685137	0.669862	73.2265	66.802011	8.77

apparently odd result is simple to explain. The MAC layer attempts to transmit the packet until the retry limit is reached or a MACK frame is received. However, not to receive the MACK does not always mean that the data packet was lost. In fact, it may be the case that it was the MACK frame that could not be received correctly due to physical errors, in which case the ensuing retries constitute useless overhead. The more the retry limit is oversized relative to the experienced $p^{\lambda}_{i,j}$, the more frequently such events will occur. In most cases, it is better to follow a more optimistic policy by reducing the MAC retry limit while trusting the caching mechanism to minimize end-to-end retransmissions in case of packet loss. This is in accordance with the obtained results: the minimum cost that can be achieved without cache ($r = 6$) is approximately the same as the cost achieved when $C_n = 5$, $r = 3$.

It is also interesting to study how fast each configuration converges to $PoD_i(t) = 1$. Fig. 5 depicts $PoD_i(t)$ as a function of the number of iterations t. The configurations with higher r converge faster even without caching. It can also be clearly

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

Table 3
Effective physical packet error rates for the scenarios in Fig. 3.

Hops	C_n	$p_{i,j}^{DATA}$	$p_{i,j}^{NACK}$	$p_{i,j}^{MACK}$	Simulation cost	Analytical cost	Difference (%)
1	0	0.560228	0.503004	0.54476	5.06	4.737944	6.36
	1	0.560228	0.503004	0.54476	5.06	4.737944	6.36
	5	0.560228	0.503004	0.54476	5.06	4.737944	6.36
	10	0.560228	0.503004	0.54476	5.06	4.737944	6.36
	20	0.560228	0.503004	0.54476	5.06	4.737944	6.36
2	0	0.587001	0.530348	0.559678	11.49	10.528386	8.37
	1	0.587225	0.528042	0.561035	11.401	10.502728	7.88
	5	0.585353	0.524394	0.563234	11.199	10.344848	7.63
	10	0.583323	0.510598	0.555869	10.898	10.101717	7.31
	20	0.58291	0.519643	0.567508	10.587	9.868532	6.79
3	0	0.576015	0.584001	0.561958	17.7613	16.696974	5.99
	1	0.57517	0.575266	0.557509	17.6053	16.465815	6.47
	5	0.572214	0.566208	0.554542	16.8186	15.831762	5.87
	10	0.581481	0.577648	0.569827	16.965	15.671234	7.63
	20	0.583757	0.552406	0.571605	16.349	14.952874	8.54
4	0	0.598519	0.595996	0.559764	27.847	25.34869	8.97
	1	0.595298	0.605155	0.565576	26.3981	24.696633	6.45
	5	0.590254	0.587744	0.567309	24.6516	22.830541	7.39
	10	0.599202	0.572932	0.572101	24.024	21.923747	8.74
	20	0.597955	0.573187	0.575619	22.354	20.434258	8.59
5	0	0.614867	0.575882	0.556697	39.5568	36.563451	7.57
	1	0.613694	0.57919	0.563584	38.8682	35.044036	9.84
	5	0.606675	0.579186	0.555108	32.0017	30.427542	4.92
	10	0.613804	0.597695	0.562512	31.273	28.523371	8.79
	20	0.610245	0.57405	0.562489	28.63	25.91089	9.50
6	0	0.618395	0.59078	0.570508	54.2359	49.940114	7.92
	1	0.610711	0.590884	0.566931	49.5552	45.136266	8.92
	5	0.603719	0.605676	0.56695	41.2581	37.874987	8.20
	10	0.603613	0.594272	0.571433	36.6813	34.319706	6.44
	20	0.601388	0.5967	0.573114	33.977	31.246063	8.04
7	0	0.610472	0.595532	0.550919	65.814	61.616875	6.38
	1	0.608294	0.594777	0.569733	61.2655	56.340762	8.04
	5	0.611851	0.600354	0.563008	50.939	46.411416	8.89
	10	0.607227	0.595212	0.558088	44.8602	40.599909	9.50
	20	0.603453	0.580932	0.558351	39.9645	36.480847	8.72
8	0	0.621174	0.600203	0.566616	92.9968	84.903209	8.70
	1	0.612641	0.583665	0.560534	74.4109	70.02037	5.90
	5	0.612484	0.585701	0.561229	56.687	54.385681	4.06
	10	0.610852	0.59605	0.564148	51.955	47.605964	8.37
	20	0.611266	0.592888	0.561427	46.1578	42.662995	7.57
9	0	0.626122	0.594481	0.565052	119.339	111.470291	6.59
	1	0.616901	0.593898	0.564312	91.5931	87.18388	4.81
	5	0.615283	0.600047	0.558556	68.5318	64.025591	6.58
	10	0.611471	0.596603	0.567429	58.7076	54.937569	6.42
	20	0.60858	0.592506	0.557424	52.1275	48.952637	6.09

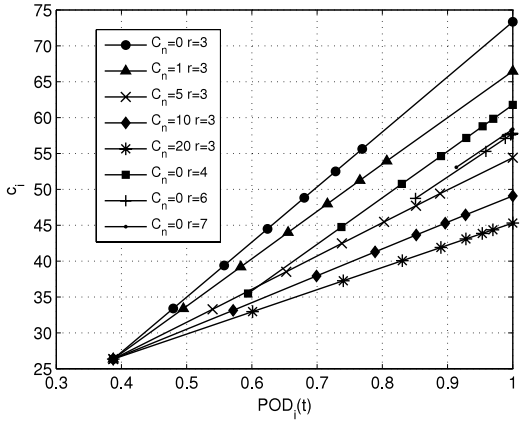


Fig. 4. Cost c_i as a function probability of delivery $PoD_i(t)$.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

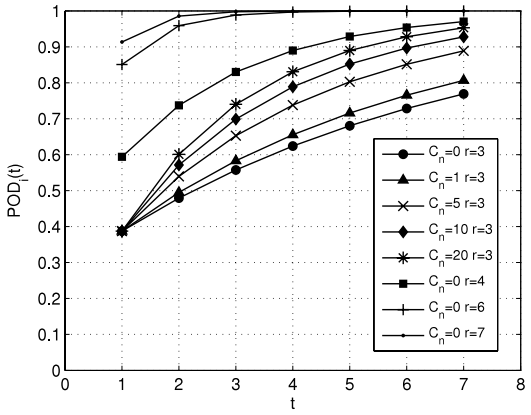
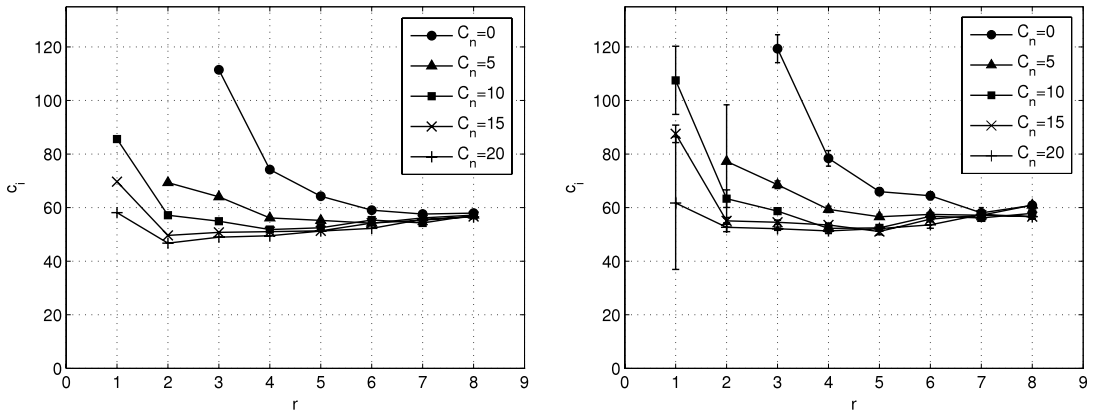


Fig. 5. Probability of delivery $PoD_i(t)$ as a function of t with $p^{\lambda}_{i,j} \approx 0.56$ and $l_i = 9$.



(a) Analytical. (b) Simulation.

Fig. 6. Single flow: cost c_i as a function of retry limit r with configured $p^{\lambda}_{i,j} \approx 0.56$ and $l_i = 9$.

seen that there is a significant increase in convergence speed between $r = 3$ and $r = 4$ and then between $r = 4$ and $r = 6$. Regarding the configurations with caching, the convergence speed increases with C_n.

Fig. 6 depicts c_i as function of r for different C_n values and now includes both analytical and simulation results. The R coefficient is 0.9982. Since the cost values with $C_n = 0$ or $C_n = 1$ are much higher compared to the other values, we just show the plots starting from $r = 3$ and $r = 2$, respectively. The costs for $r = 1$ and $r = 2$ turned out to be less accurate than for $r > 2$, since these points belong to the already mentioned nonlinear region. The increased unpredictability in this region is also attested by the size of the confidence intervals in the simulation results. However, the curves still accompany each other, with the cost difference exceeding 30% only for $r = 1$ with $C_n = 0$ and $C_n = 5$, and never exceeding 15% for $r = 2$. The difference stays always below 10% for all other r values. In Table 4, the effective $\pi^{\lambda}_{i,j}$ average values are presented together with the respective simulation and analytical costs.

Confirming the previous discussion, all the curves present a global minimum, whose value decreases with C_n. The results are similar for higher $p^{\lambda}_{i,j}$ values, though the argmin values become increasingly more spread in the horizontal axis.

In summary, the results presented in this section demonstrate that arbitrarily increasing the MAC retry limit may lead to higher costs in terms of energy, throughput and interference, even if it can potentially speed-up end-to-end delivery. It also does not eliminate the benefits of intermediate caching at the transport layer. Instead, these solutions should be integrated and the number of MAC retry limit should be dynamically adapted based on measured link conditions. The need for optimization of the MAC retry limit and its impact on transport already constitutes an open research topic (e.g., [21]).

5.3. Concurrent flows with uniform link quality and length

In all multiflow scenarios it is assumed that the $\pi^{\lambda}_{i,j}$ for DATA, NACK and MACK packets are the same, in order to allow a consistent comparison with the results for single flow scenarios. The fact that the amount of cache assigned to a flow has a significant impact on the cost of delivery raises the issue of how to partition the cache when the node is crossed by

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

Table 4
Effective physical packet error rates for the scenarios in Fig. 6.

r	C_n	$p_{i,j}^{DATA}$	$p_{i,j}^{NACK}$	$p_{i,j}^{MACK}$	Simulation cost	Analytical cost	Difference (%)
1	0	0.645025	0.583938	0.562544	2088.79	2727.97782	−30.60
	5	0.609968	0.583705	0.562692	227.6	136.503971	40.02
	10	0.604045	0.580073	0.542225	107.525	85.598434	20.39
	15	0.606249	0.578666	0.559406	87.55	69.696689	20.39
	20	0.597496	0.572447	0.545347	61.7	58.10221	5.83
2	0	0.628559	0.592796	0.564844	243.926	237.441118	2.66
	5	0.59496	0.584583	0.551577	77.2667	69.352053	10.24
	10	0.603769	0.589414	0.561393	63.3409	57.169115	9.74
	15	0.600773	0.566213	0.553275	55.04	49.60971	9.87
	20	0.600261	0.583903	0.547298	52.6429	46.732815	11.23
3	0	0.626122	0.594481	0.565052	119.339	111.470291	6.59
	5	0.615283	0.600047	0.558556	68.5318	64.025591	6.58
	10	0.611471	0.596603	0.567429	58.7076	54.937569	6.42
	15	0.610174	0.593684	0.553223	54.5338	50.776135	6.89
	20	0.60858	0.592506	0.557424	52.1275	48.952637	6.09
4	0	0.607554	0.580622	0.565334	78.3979	74.205735	5.35
	5	0.601694	0.595453	0.564772	59.3635	56.16042	5.40
	10	0.582198	0.603257	0.559973	52.2667	51.811635	0.87
	15	0.5992	0.585247	0.558741	53.506	51.044408	4.60
	20	0.595679	0.578679	0.553352	51.2893	49.480693	3.53
5	0	0.596486	0.579336	0.567318	65.9693	64.272349	2.57
	5	0.590828	0.611826	0.559365	56.5828	55.241823	2.37
	10	0.577452	0.599243	0.547004	52.41	52.457283	−0.09
	15	0.572322	0.631193	0.550461	51.05	51.330259	−0.55
	20	0.581369	0.598653	0.549189	52.1437	51.242258	1.73
6	0	0.593434	0.583668	0.565107	64.4226	59.038472	8.36
	5	0.586599	0.572037	0.553642	57.4741	54.191508	5.71
	10	0.585062	0.563618	0.556064	56.7621	55.347149	2.49
	15	0.581583	0.551044	0.551226	55.7194	54.218353	2.69
	20	0.579498	0.512286	0.52595	53.5833	52.238965	2.51
7	0	0.583233	0.499566	0.526946	58.05	57.60894	0.76
	5	0.588764	0.532741	0.525389	57.17	56.163115	1.76
	10	0.576819	0.583667	0.519683	56.16	54.255603	3.39
	15	0.586625	0.571238	0.532333	57.1375	55.335621	3.15
	20	0.58668	0.53776	0.539607	57.43	55.609651	3.17
8	0	0.589325	0.362612	0.549178	60.85	57.926418	4.80
	5	0.590449	0.403133	0.552173	60.9086	57.208549	6.07
	10	0.583112	0.254292	0.53288	57.85	57.154557	1.20
	15	0.582534	0.151216	0.530706	56.6313	56.671036	−0.07
	20	0.582534	0.151216	0.530706	56.6313	56.546956	0.15

more than one flow. Fig. 3 suggests that when each node in the network is simultaneously crossed by two flows, to divide the cache equally between the flows (equal to ω_i^n) already results in near-optimal performance. In more complex scenarios with a higher number of concurrent flows, the optimality of this uniform ω_i^n is not obvious, namely when the flows have different lengths or experience different link conditions.

The first scenario consists of a 10×10 grid with crossing 9-hop flows similar to the one depicted in Fig. 7. Since each flow shares its path with a reverse flow, each intermediate node is crossed by four flows. Five cache partitioning policies were tested:

1. Uniform: $\omega_i^n = 1$ for all flows at all nodes.
2. Increasing: For each flow, $\omega_i^n = 1$ at the destination, and this is divided by two in every other hop as the distance from the destination increases.
3. Decreasing: For each flow, $\omega_i^n = 1$ at the source, and this is divided by two in every other hop as the distance from the source increases.
4. U-shape: For each flow, $\omega_i^n = 1$ at one node located in the center of the path, and this is divided by two in every other hop as the distance from the center node increases.
5. Inv-U-shape: For each flow, $\omega_i^n = 1$ at the source and this is divided by two in every other hop away from it until a center node is reached, then it is multiplied by two at each hop as the distance from the destination decreases.

It should be recalled that the ω_i^n is just a weight and not the actual fraction of cache assigned to flow $\varphi(i)$ at node n, which is $\rho_i^n = \frac{\omega_i^n}{\sum_{j=0}^{Fn} \omega_j^n}$.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

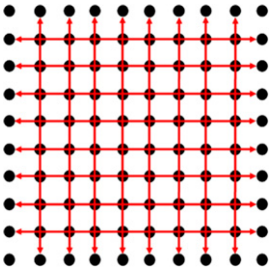


Fig. 7. Concurrent flows in a grid topology.

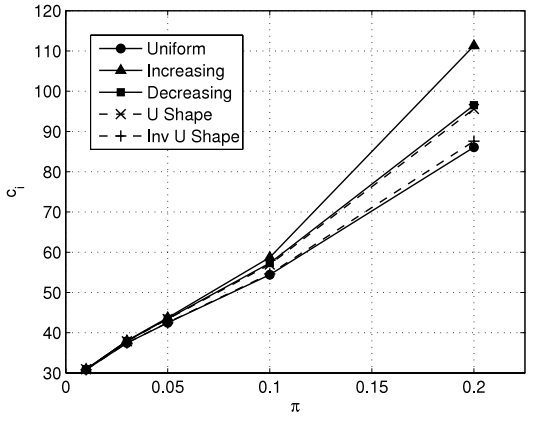


Fig. 8. Square grid with constant $\pi^{\lambda}_{i,j}$: the cost c_i as a function of $\pi^{\lambda}_{i,j}$ with $C_n = 20$ for different cache partitioning policies with $l_i = 9$.

The cost c_i is depicted in Fig. 8 as a function of $\pi^{\lambda}_{i,j}$ with $C_n = 20$ and $l_i = 9$ for different cache partitioning policies. As can be seen, the Uniform and Inv-U-Shape are the most efficient policies, presenting almost identical results. However, Uniform policy is the fairest, with null standard deviation.

5.4. Concurrent flows with non-uniform link quality and length

The previous results have assumed that the link quality is uniform across the network. In case some flow experiences worse conditions (e.g., greater hop length and/or bad links), it is obvious that it should be assigned a larger slice of the cache. Besides, even for a flow that experiences bad end-to-end conditions, its path may comprise links that are in worse conditions than others. In this case, it is desirable to cache the packet once it is successfully transmitted beyond a bad link. A non-Uniform caching policy is defined, which takes into account both the link error rates and the length of the flow, assigning ω_i^n values as follows:

$$\omega_i^{\varphi_i(k)} = \left(\pi^{DATA}_{\varphi_i(k),\varphi_i(k+1)} \cdot \pi^{NACK}_{\varphi_i(k),\varphi_i(k+1)}\right)^a (l_i)^b \tag{14}$$

where $a \geq 0$ and $b \geq 0$. From the analysis of the equation, it comes out that in a scenario where all the links have equal error rates and all flows have the same length, this policy converges to the Uniform policy. It is also equivalent to the Uniform policy when $a = b = 0$. Otherwise, it assigns greater weight to the flows that are longer and links that present higher $\pi^{\lambda}_{i,j}$.

Fig. 9 depicts the cost c_i in the same grid topology, but where the vertical links present $\pi^{\lambda}_{i,j} = 0.2$ and the horizontal $\pi^{\lambda}_{i,j}$ varies. The other conditions are kept the same, namely $C_n = 20$ and $l_i = 9$. Again, it is assumed that the $\pi^{\lambda}_{i,j}$ for DATA, NACK and MACK packets are the same. The Uniform cache policy is compared with the non-Uniform cache policy, where $a = 2$ and $b = 2$ (in this experiment, parameter b has no impact since the flow lengths are the same). The non-Uniform policy can achieve improved fairness for horizontal $\pi^{\lambda}_{i,j}$ values within $[0, 0.1]$, lowering the maximum c_i and also the standard deviation. The obvious trade-of is an increase of the minimum c_i. The average c_i is also slightly lower than Uniform until the horizontal and vertical $\pi^{\lambda}_{i,j}$ becomes close enough ($\pi^{\lambda}_{i,j} = 0.1$) that it rises beyond Uniform due to the disproportionate ω_i^n values. This is also the point where horizontal flow performance starts to be worse than vertical flow performance (although this cannot be seen in the graphic, the maximum and minimum values belong to horizontal and vertical flows respectively), which is not desirable. As an attempt to mitigate the problem, lower values for the non-Uniform index a were tested. However, as a decreased, the fairness improvement relative to the Uniform policy also decreased. The conclusion is that the cache assignment that results from Eq. (14) is not optimal. The use of more advanced optimization techniques such as convex

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

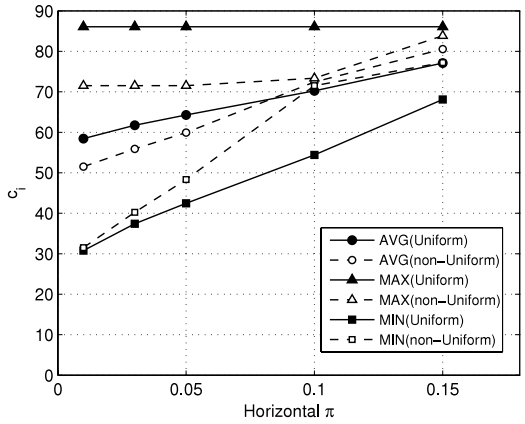


Fig. 9. Square grid with different $\pi_{i,j}^{\lambda}$: the cost c_i as a function of the horizontal $\pi_{i,j}^{\lambda}$ with vertical $\pi_{i,j}^{\lambda} = 0.2$, $C_n = 20$, $l_i = 9$, for the Uniform and non-Uniform cache policies.

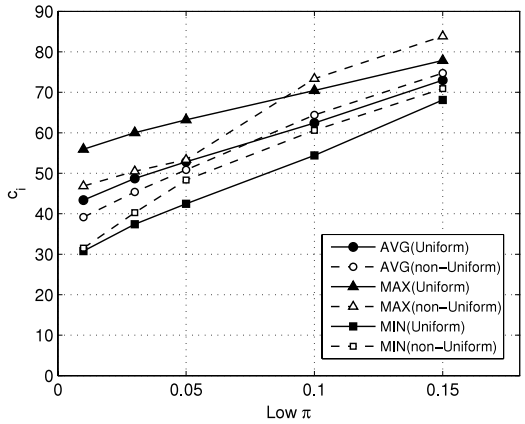


Fig. 10. Square grid with higher $\pi_{i,j}^{\lambda} = 0.2$ in one half of the vertical flows: the cost c_i as a function of the $\pi_{i,j}^{\lambda}$ in the remaining links, with $C_n = 20$, for the Uniform and two non-Uniform cache policies.

optimization and game theory appear to be promising solutions to find more optimal policies. Furthermore, algorithms that adapt to dynamic network conditions need to be explored.

Additional tests have also shown that the impact of the cache partitioning parameters increases with the vertical $\pi_{i,j}^{\lambda}$, which is not a surprising result. On the other hand, for vertical $\pi_{i,j}^{\lambda} = 0.1$, the curves present similar shapes, but the difference between policies is not as noticeable (the decrease in the maximum c_i is still below 10%).

A variant of this scenario was prepared in which the first 5-hop segments of the vertical flows endure $\pi_{i,j}^{\lambda} = 0.2$, while the remaining links in the network endure $\pi_{i,j}^{\lambda} \in [0.01, 0.15]$. Fig. 10 depicts c_i as a function of the lower $\pi_{i,j}^{\lambda}$ values. Once again, it can be seen that the non-Uniform policy results in an improvement of the maximum (and average) c_i until the low $\pi_{i,j}^{\lambda}$ reaches approximately 0.1. It is at this point that the horizontal flows start to present worse performance than the vertical flows, although the latter present a greater average $\pi_{i,j}^{\lambda}$. The manipulation of the a parameter as well as of the $\pi_{i,j}^{\lambda}$ range provided results that are similar to the previous scenario.

The length of the flow is another variable with significant impact on the c_i. For the same link quality, concurrent flows with greater hop-length have more difficulty in completing end-to-end delivery and thus should receive a larger cache slice. The next scenario is based on the former, but at this time the grid is rectangular, with 16 5-hop horizontal flows (8 rows, each with two flows in opposing directions) cross eight 9-hop vertical flows (4 columns, each with two flows in opposing directions). Like in the previous scenario, the vertical links present $\pi_{i,j}^{\lambda} = 0.2$ and the horizontal $\pi_{i,j}^{\lambda}$ varies. The non-Uniform policy is parameterized with $a = 2$ and $b = 2$. The results are depicted in Fig. 11. The difference in terms of maximum c_i is around 16.9%. The average c_i for non-Uniform is slightly lower until $\pi_{i,j}^{\lambda}$ is approximately 0.15 for the horizontal flows. When the horizontal and vertical $\pi_{i,j}^{\lambda}$ values are equal at 0.2, the non-Uniform policy still presents a lower maximum c_i, since it favors the longer flows. Although not shown in the graphic, the standard deviation is always significantly lower for the non-Uniform policy, attesting its greater fairness.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

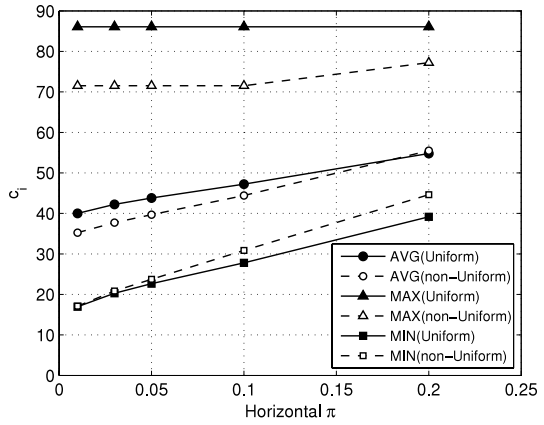


Fig. 11. Rectangular grid with different $\pi^{\lambda}_{i,j}$: the cost c_i as a function of the horizontal $\pi^{\lambda}_{i,j}$ with vertical $\pi^{\lambda}_{i,j} = 0.2$, $C_n = 20$, for the Uniform and non-Uniform cache policies.

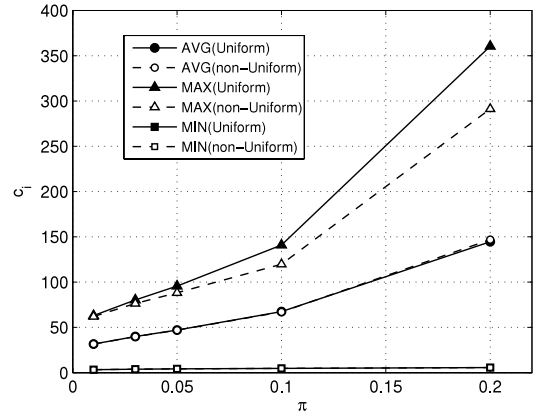


Fig. 12. Convergecast pattern on a grid: the cost c_i as a function of $\pi^{\lambda}_{i,j}$ with $C_n = 20$ for the Uniform and length-dependent cache policies.

5.5. Convergecast scenario

The next scenario consists of a convergecast flow pattern (typical in mainstream WSNs) in a 10×10 grid network, where the node in the top left corner is the sink node and all the other 99 nodes are data sources. The flow paths were selected based on a shortest-path algorithm and the $\pi^{\lambda}_{i,j}$ values are constant across the grid. Again, it is assumed that the $\pi^{\lambda}_{i,j}$ for DATA, NACK and MACK packets are the same. The Uniform policy is compared with the same non-Uniform policy, parameterized with $a = 2$ and $b = 3$.

Fig. 12 depicts the cost c_i as a function of $\pi^{\lambda}_{i,j}$ with $C_n = 20$ for the Uniform and length-dependent cache policies. While the average and minimum c_i are practically the same for both policies, the length-dependent policy reduces the maximum c_i by approximately 15% for $\pi^{\lambda}_{i,j} = 0.1$ and 19% for $\pi^{\lambda}_{i,j} = 0.2$. Accordingly, the standard deviation is reduced from 34.05 to 31.59 with $\pi^{\lambda}_{i,j} = 0.1$ and from 87.85 to 78.85 for $\pi^{\lambda}_{i,j} = 0.2$. These reductions tend to increase with $\pi^{\lambda}_{i,j}$.

6. Conclusion and future work

This paper has presented an analytical model of WSN reliable transport with intermediate caching using a probabilistic formulation which has been validated through simulations. The model allows the performance evaluation of end-to-end delivery in terms of the required number of physical layer transmissions required to achieve a given probability of end-to-end delivery. The presented cost function can be fed to a technology-dependent energy model, allowing the prediction of the related energy cost. Although the analytical model is mainly inspired by the mechanisms of the DTSN transport protocol, it was designed to be more generic, allowing the manipulation of intermediate caching parameters that apply to other WSN transport protocols.

The WSN low-power radio links present high packet loss ratios, especially when enduring interference from other radio technologies (e.g., interference of IEEE 802.11 on IEEE 802.15.4 WSNs). The presented results show the advantages of integrating the transport-layer intermediate caching mechanism in such environments, even if ARQ-based link-layer error recovery is present. In the latter case, a dynamic adjustment of the MAC retry-limit should also be used to improve

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

the efficiency of end-to-end delivery. Further results demonstrate the need to optimize cache partitioning in scenarios with concurrent flows, taking into account the error rate of the links, as well as the length of the flows.

The authors are currently exploiting the presented results in order to develop distributed cache optimization algorithms that adapt to dynamic network conditions. Transport-routing cross-layer caching optimization constitutes another promising topic for future research.

Acknowledgments

This work has been partially supported by the European Commission Seventh Framework Programme under grant agreement no. 225186, project WSAN4CIP. The research also received funding from FCT (INESC-ID multiannual funding) through the PIDDAC program funds. Nestor Tiglao would like to acknowledge the support of the Philippine Department of Science and Technology-Science Education Institute (DOST-SEI) and the University of the Philippines Engineering Research and Development for Technology (UP ERDT).

References

[1] IEEE Standard for Information Technology Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs), IEEE Std 802.15.4-2003
[2] A. Dunkels, J. Alonso, T. Voigt, H. Ritter, Distributed TCP caching for wireless sensor networks, in: Proceedings of the 3rd Annual Mediterranean Ad-Hoc Networks Workshop, Med-Hoc-Net, Bodrum, Turkey, 2004.
[3] B. Marchi, A. Grilo, M. Nunes, DTSN: distributed transport for sensor networks, in: Proceedings of the IEEE Symposium on Computers and Communications, ISCC'07, IEEE, Aveiro, Portugal, ISBN: 978-1-4244-1520-5, 2007, pp. 165–172.
[4] F. Rocha, A. Grilo, P.R. Pereira, M.S. Nunes, A. Casaca, Performance evaluation of DTSN in wireless sensor networks, in: Proceedings of Wireless Systems and Mobility in Next Generation Internet: 4th International Workshop of the EuroNGI/EuroFGI Network of Excellence Barcelona, Spain, January 16–18,2008, Revised Selected Papers, Springer-Verlag, 2008, pp. 1–9.
[5] C. Wang, K. Sohraby, B. Li, M. Daneshmand, Y. Hu, A Survey of Transport Protocols for Wireless Sensor Networks, in: IEEE Network, vol. 20 (3), IEEE, 2006.
[6] F. Stann, J. Heidemann, RMST: reliable data transport in sensor networks, in: Proceeding of the 1st IEEE International Workshop on Sensor Net Protocols and Applications, IEEE, Anchorage, Alaska, USA, ISBN: 0-7803-7879-2, 2003, pp. 102–112.
[7] C. Wan, A. Campbell, L. Krishnamurthy, PSFQ: a reliable transport protocol for wireless sensor networks, in: Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications, WSNA 2002, ACM, Atlanta, Georgia, USA, ISBN: 1-58113-589-0, 2002, pp. 1–11.
[8] S.-J. Park, R. Vedantham, R. Sivakumar, I. Akyildiz, A scalable approach for reliable downstream data delivery in wireless sensor networks, in: Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2004, ACM, Tokyo, Japan, 2004, pp. 78–79.
[9] S.-J. Park, R. Vedantham, R. Sivakumar, I. Akyildiz, GARUDA: Achieving Effective Reliability for Dowstream Data Delivery in Wireless Sensor Networks, in: IEEE Transactions on Mobile Computing, vol. 7 (2), IEEE, 2008, pp. 214–230.
[10] Y. Liu, H. Huang, K. Xu, Multi-path-based distributed TCP caching for wireless sensor networks, in: Proceedings of the Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, SNPD 07, IEEE, Washington DC, USA, ISBN: 978-0-7695-2909-7, 2007, pp. 331–335.
[11] A. Ayadi, P. Maille, D. Ros, Improving distributed TCP caching for wireless sensor networks, in: Proceedings of the 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop, Med-Hoc-Net 2010, IEEE, Juan Les Pins, France, ISBN: 978-1-4244-8436-2, 2010, pp. 1–6.
[12] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, D. Estrin, A wireless sensor network for structural monitoring, in: Proceedings of the Second International Conference on Embedded Networked Sensor Systems, SenSys 2004, ACM Press, Baltimore, MD, USA, ISBN: 1-58113-879-2, 2004, pp. 13–24.
[13] E. Paek, R. Govindan, RCRT: rate-controlled reliable transport for wireless sensor networks, in: Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (Sensys 2007), ACM Press, Sydney, Australia, ISBN: 978-1-59593-763-6, 2007, pp. 305–319.
[14] M. Yaghmaee, D. Adjeroh, A reliable transport protocol for wireless sensor networks, in: Proceedings of the 4th International Symposium on Telecommunication, IST2008, IEEE, Tehran, Iran, ISBN: 978-1-4244-2750-5, 2008, pp. 440–445.
[15] H. Zhou, X. Guan, C. Wu, Reliable transport with memory consideration in wireless sensor networks, in: Proceedings of the IEEE International Conference on Communications, ICC 2008, IEEE, Beijing, China, ISBN: 978-1-4244-2075-9, 2008, pp. 2819–2824.
[16] N. Handigol, K. Selvaradjou, C. Murthy, A reliable data transport protocol for partitioned actors in wireless sensor and actor networks, in: Proceedings of the International Conference on High Performance Computing, HiPC 2010, IEEE, Goa, India, ISBN: 978-1-4244-8518-5, 2010, pp. 1–8.
[17] B. Deb, S. Bhatnagar, B. Nath, Information assurance in sensor networks, in: Proceedings of the 2nd ACM International Workshop on Wireless Sensor Networks and Applications, WSNA 2003, ACM, San Diego, CA, USA, ISBN: 1-58113-764-8, 2003.
[18] B. Deb, S. Bhatnagar, B. Nath, ReInForM: reliable information forwarding using multiple paths in wireless sensor networks, in: Proceedings of the 28th Annual conference on Local Computer Networks, LCN2003, IEEE, Bonn, Germany, ISBN: 0-7695-2037-5, 2003.
[19] M. Petrova, J. Riihijärvi, P. Mähönen, S. Labella, Performance study of IEEE 802.15.4 using measurements and simulations, in: Proceedings of the Wireless Communications and Networking Conference, (WCNC 2006), IEEE, Las Vegas, NV, USA, ISBN: 1-4244-0269-7, 2006, pp. 487–492.
[20] The Network Simulator—ns-2, http://www.isi.edu/nsnam/ns/.
[21] A. Shadmad, M. Shikh-Bahaei, TCP dynamics and adaptive MAC retry-limit aware link-layer adaptation over IEEE 802.11 WLAN, in: Proceedings of the Seventh Communication Networks and Services Research Conference, CNSR '09, IEEE, Moncton, Canada, ISBN: 978-1-4244-4155-6, 2009, pp. 193–200.

Nestor Michael C. Tiglao received his B.S. and M.S. degrees in Electrical Engineering from the University of the Philippines, Diliman, Quezon City, Philippines, in 1995 and 2005, respectively. He is currently pursuing his Ph.D. in Electrical and Computer Engineering at Instituto Superior Técnico, Technical University of Lisbon, Portugal, under the guidance of Dr. António M. Grilo. His main research interests are in wireless sensor networks, traffic management, and resource allocation.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002

António M. Grilo received the Information Technology Engineering degree in 1996, the M.Sc. degree in Electrotechnical Engineering and Computers in 1998 and the Ph.D. also in Electrotechnical Engineering and Computers in 2004, all from the IST, Technical University of Lisboa, Portugal. He is currently an Assistant Professor at IST, Department of Electrotechnical and Computer Engineering. In 1995 he joined INESC, Lisboa, becoming a senior researcher in 2004. From 1996 until now he has been working in several European projects, namely ACTS projects ATHOC and AROMA, and IST projects MOICANE, OLYMPIC, AIRNET and UbiSec&Sens. He is currently participating as task leader in FP7 project WSAN4CIP. His main research interests are related with Wireless and Mobile Networks in general, with emphasis on Ad-hoc and Sensor Networks.

Please cite this article in press as: N.M.C. Tiglao, A.M. Grilo, An analytical model for transport layer caching in wireless sensor networks, Performance Evaluation (2012), doi:10.1016/j.peva.2011.12.002