Implementation of an On-Demand Routing Protocol for Wireless Sensor Networks

Yang Zhang, Jian Wu and Paul Havinga,

Abstract— **We present our experiences in implementing and validating the on-demand EYES Source Routing protocol (ESR) in a real wireless sensor network (WSN) environment. ESR has a fast recovery mechanism relying on MAC layer feedback to overcome frequent network topology changes resulting from node mobility and unreliability. A geographically restricted directional flooding scheme reduces energy consumption in the route re-establishment. ESR is implemented in our WSN environment consisting of EYES sensor node prototypes using the Lightweight Medium Access Control protocol (LMAC) on top of the AmbientRT operating system. We describe the key design and implementation features of our protocol and report experiment results of ESR and Ad hoc On-demand Distance Vector protocol (AODV), a conventional routing protocol for ad hoc networks.**

***Index Terms*— WSNs, ESR, LMAC, AmbientRT kernel**

I. Introduction

A Wireless Sensor Network (WSN) is a collection of small inexpensive sensor devices with sensing, processing and wireless communication capabilities forming an autonomous ad hoc wireless network for data collection, communication and delivery. Several constrains of WSNs are introduced to routing protocol design. The nodes in the WSN need to contend for the limited resources of WSNs, especially conserving battery power. The routing algorithm must handle frequent network topology changes caused by node mobility and unreliability. The routes of WSNs are often "multi-hop" due to limited range of each node's radio propagation range. These constrains pose challenges for the design of routing protocols for WSNs.

The conventional routing protocols for ad hoc networks such as Dynamic Source Routing protocol (DSR) [1] and Ad hoc On-demand Distance Vector protocol (AODV) [2] are not well suited for WSNs. DSR causes excessive overhead in the data packet's header containing the complete hop-by-hop route to the destination. AODV adds the significant overhead for Hello message, without the support from MAC layer. Moreover, these schemes re-flood the whole network in route re-establishment. Such a measure would be less efficient with the increase of the average movement speed of nodes and the diameter of network.

To overcome these limitations, we developed the on-demand EYES [3] Source Routing protocol (ESR) [4]. ESR applies an on-demand routing technique to establish dynamic, self-starting and multi-hop route. It uses fast recovery mechanism replying on MAC layer feedback, to recover the broken link in a local and efficient manner. By a limited directional flooding scheme, energy consuming re-flooding of the whole network in the route re-establishment is avoided. ESR keeps all the routing messages as small and fixed length, and stores limited routing information in each sensor node.

ESR and its performance have been studied in the extensive simulation [4], and it is shown to outperform conventional routing protocols. Our next step in protocol development is to validate the protocol working with the Lightweight Medium Access Control protocol (LMAC) [5] on top of the AmbientRT [6] operating system in a real WSN environment. In this paper, we make a performance comparison between ESR and AODV in the same network environment.

The rest of this paper is organized as follows. ESR is briefly introduced in Section II followed by the routing protocol implementation description in Section III. Performance evaluation results and the correctness of ESR in dynamic scenarios are discussed in Section IV, and concluding remarks are made in Section V.

Manuscript received March 27, 2006. This work is performed as a part of the European EYES project (IST-2001-34734) [3].

Yang Zhang, Jian Wu and Paul Havinga are with University of Twnete, Enschede, 7521HA, The Netherlands (phone: +31-53-4893626; fax: +31-53-4894590; e-mail: zhangy|jian|havinga@cs.utwente.nl).

II. ESR Overview

This section gives a short overview of the on-demand EYES Source Routing protocol (ESR). For a detailed operation of the protocol, readers are referred to [4].

ESR establishes the route by the source node on demand. Similar to on-demand unicast routing protocols, a query phase and a reply phase comprise the protocol. After *Route Setup*, each intermediate node learns its best source and destination neighbors, which reduces routing overhead during data packet transmission.

A *Route Re-catch* message is broadcast locally when a node notices its best destination neighbor floats away from its own transmission range. Eventually, the broken link is recovered successfully by the reply from an on-route node. A *Route cut* message is sent in one-hop neighborhood when a node notices its second-order upstream neighbor comes into its transmission range. Eventually, a simple message effectively shortens the redundant link.

If *Route Re-catch* process fails, a limited directional flooding with a *Route Re-request* message makes old on-route nodes act as a HTL repeater and forward the request directly to reach the destination. It avoids costly energy consuming re-flooding.

III. Implementation

A. Implementation Platform

1) Operating System and Software: We developed ESR on the AmbientRT kernel version 0.4. All tools and software packages we used in our development originate from software bundle incorporated within the AmbientRT version 0.4 operating system package. AmbientRT needs and is tested with the third party software packages, such as MSPGCC, MinGW and Minimal SYStem. We chose the AmbientRT operation system for its real-time scheduler, dynamic memory allocation support, and a hardware abstraction layer creation.

2) Hardware: a wireless sensor node consists of a small and low-power processor, a single channel transceiver and some additional components like serial memory and debugging interface. The processor (MSP430F149) produced by Texas Instruments is a 16-bit processor, running at 4.6 MHz, with 2048 bytes of Random Access Memory (RAM), and 60 KB of programmable flash memory. The single channel transceiver (RFM TR1001) supports transmission rates up to 115.2 Kbps. For programming the CPU, this hardware has a JTAG interface and the power is supplied by two AA Alkaline batteries.

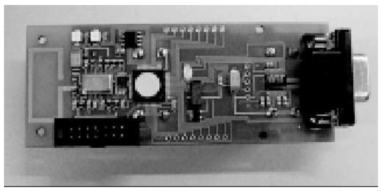
Figure 1 EYES Sensor Node Prototype

B. LMAC Operation

We developed ESR with the support of LMAC, a novel MAC protocol based on Time Division Multiple Access (TDMA) for WSNs. It allows nodes to autonomously choose one time slot of a frame to control. Each node can transmit data messages in its own time slot controlled to avoid the competition of medium with other nodes, and to save energy from collision and overhearing. Moreover, LMAC takes into account the physical layer properties. It minimizes the number of transceiver switches to make the sleep interval for sensor nodes adaptive to the amount of data traffic and to limit the complexity of the implementation.

Each time slot includes two sections: Traffic Control (TC) and data section. The owner of a time slot will always transmit a TC message in the time slot. In our implementation, routing request messages are allowed to exist inside the structure of TC message, which reduces the routing overhead and transmission energy. All nodes within one-hop distance of the controller of the current time slot will put effort into receiving the message, since this message is used for synchronization purpose and control information. Also the TC message can indicate that the controlling node is about to send data message. By listening to TC sections of neighboring nodes, nodes have the knowledge of local topology, which assists routing and reduces the number of routing messages in the network.

C. Real-Time Tasks

We use two specific entities within AmbientRT, the data type and the task. They can be identified to OS by a Data Specification File (DSF), which not only has a more readable and writable form, but also prevents implementation mistakes and simplifies the implementation process.

A data type with *radio_message* is used to store data message received on the radio transceiver, and to publish in the receivers. It is just the glue that interconnects different tasks together. Two tasks are deployed in each sensor node. One is *timer*, which is subscribed to a timer and has exclusive access to the resource of radio transceiver. It runs at the beginning of each new time slot, and to transmit TC and network messages in the time slot. Each node in the WSN calls it periodically and synchronously. The other task is aperiodic *radio_in*, which is subscribed when message is received on the radio transceiver, also has exclusive access to the resource of radio transceiver. It will read and write the data type *radio_message*. Most of LMAC and ESR modules operate within the task.

The kernel of AmbientRT is a real-time scheduler, which uses dynamic priorities to enable multi-tasking. This means that the priority of a task changes over time. The scheduler uses the *absolute deadline* as the priority of a task. Such a measure leads to a better utilization of the processor than fixed priorities. In order to preserve the integrity of a resource in the multi-tasking system, the mutual exclusion mechanism in the AmbientRT operation system avoids this concurrent use of un-shareable resources, e.g. radio transceiver in our implementation.

D. Software Architecture

ESR is built upon the LMAC protocol. The implementation architecture of ESR is shown in Figure 2. ESR accepts all data messages, stores them in the data message cache, and starts the cache timer. The cached messages are discarded when the timer expires. The cache utilizes the list structure, which is easy to insert or delete message. The *Route Table* creates and maintains route entries on demand and keeps freshness. ESR opens five interfaces for communicating with LMAC. The TC message and network messages are buffered into the corresponding interfaces. Specifically, ESR routing request message is to be transmitted along TC message in the *Transmitted TC MSG* interface, other hop-by-hop routing message and data message are transmitted in the *Transmitted Network MSG*. Conversely, ESR handles route request message probably existed in the *Received TC MSG* interface, and handles other hop-by-hop routing message and data message in *the Received Network MSG* interface. The *Neighbor Table* interface mirrors LMAC neighbor table, which provides local topology information, and further triggers ESR *Re-catch* module. The advantage of the *Interface Layer* is to keep the independence of ESR and make it to apply for different applications. When nodes try to discover the network again, all information of LMAC is lost, while ESR maintains all routing information, independent on LMAC. In the following sections, we describe our modules to manage the network messages and the route table, and discuss ESR *Re-catch* and *Re-cut* modules.

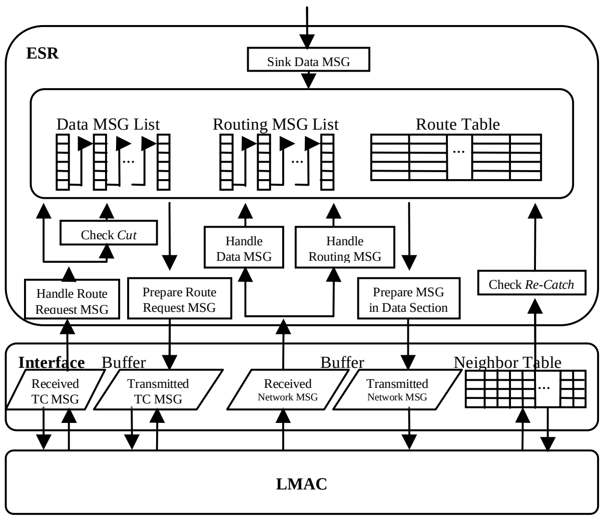

Figure 2 ESR Implementation Structure

1) Packet and Route Table Management: The *Route Table* stores only the best neighbors for each source-destination pair. The sequence number is to determine freshness of routing information and prevent routing loops. The *Route Table* controls the change in data and routing messages caches. The chains of operations for the three parts are often as an integrated one to manage. In our implementation, ESR routing messages have higher priority than data messages because they are urgent to handle dynamic network events.

When intermediate nodes receive the *Route Error* message, they maintain the route entry, only discard the best neighbors and restart the timer of unsent data messages. Thus, these nodes with the old route entry would rebroadcast the *Route Re-request* message, as the HTL repeater to achieve the limited directional flooding in the route re-establishment.

2) Route Re-Catch and Cut Modules: The neighbor table provides local topology information to ESR. If ESR detects that the best destination neighbor disappears in the neighbor table, it starts *Route Re-catch* module. In order to avoid routing loop caused by the reply from the upstream on-route nodes, only the downstream node of the broken link replies the first received *Re-catch Request* message. Moreover, the message delays one frame to send, considering at the same time two sides of the broken link may not detect it. In the LMAC implementation, in order to solve unidirectional link problem, one node could delete another one in the neighbor table if its own slot doesn't exist in the corresponding node's *slot_usage*.

Because each node only knows the local topology in its one-hop distance, it is very difficult to realize its second-order upstream neighbor come into its transmission range. In our implementation, hence, ESR starts *Route Cut* module only when the node is not addressed by incoming data message and also the receiver is as one of its best source neighbors in the route table. In *Route Cut* module, ESR maintains the *Route Cut* message in the cache for a while, to avoid duplicate ones. The second-order upstream neighbor node responds the message only if the same route entry exists, and its next-hop destination neighbor is just the old best source neighbor of the sender. The old best source neighbor node responds the message only if it is certainly the intermediate node between the above two nodes.

3) ESR Timers: In the message caches, we set a cache timer to refresh messages. In the route table, there are three kinds of timers. Waiting timer for the *Route Reply* message works in the route discovery phase. Waiting timer for the *Route Re-catch* message works in the *Route Re-catch* discovery phase and has shorter period than the first one. Waiting timer for the data message works after establishes the route successfully and has the longest period in all of them. The route entries are discarded when the timer expires.

IV. PERFORMANCE EVALUATION

We created a wireless sensor network with 12 nodes built within a maximal flat space of 12×8 m^2. There are two routes between two pairs of different sources and destinations. The hops distance from the source to the destination changes with the dynamic network topology. The maximal distance is about 3~4 hops. Each node moves with a speed of 0~2m/s to a location of any of node within this area. After a rest for a random period between 0~1 minute, the node moves again to the next location. We collect performance information in the sensor-to-gateway way, and read them by the hyper terminal of the PC. We compare ESR performance with the implementation based on basic AODV algorithm. In our experiments, the network setup and parameters are identical for both protocols. We present and analyze the results of ESR and AODV in this section.

A. AODV Overview

AODV (Ad hoc On-demand Distance Vector) is based on on-demand routing algorithm. In this protocol, routes are built between nodes only as desired by source nodes. And it uses traditional routing table, one entry per destination, to maintain routing information. AODV uses sequence numbers to ensure the freshness of routes, and also uses a timer-based route expiry mechanism to promptly discard stale routes. It sends Hello message periodically to detect and monitor links to neighbors, without the support of MAC layer. Moreover, destination node and any node that has a current route to the destination node can generate the reply message.

B. Dynamic Network Scenario and Results

1) Re-catch and Re-request Success Ratio Analysis: The most significant characteristic of ESR is that *Route Re-catch* recovers the broken link efficiently, which suppress the rate of energy consuming route re-establishment to a minimal level. In the experiments, we evaluate the re-catch and re-request success ratio by different topological rate and network size. Figure 3 shows that it has good performance with the increase of the movement speed of nodes and the hop distance. As long as the topological rate changes not much, ESR *Re-catch* module have enough time to recover the broken link successfully, whatever the increase of the size of the network. Once *Re-catch* process fails, *Re-request* success ratio is always close to 100% and establish a new route for data transmission in our experiments.

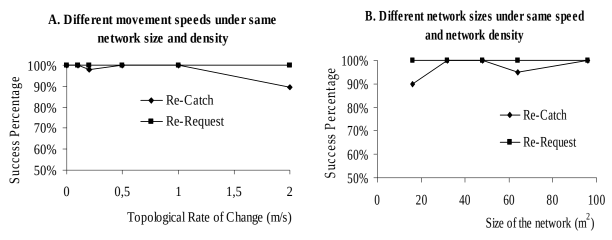

Figure 3 ESR with *Re-catch/Re-request Success Ratio*

2) End-to-End Data Throughput Analysis: The important quantitative criterion reflects the performance of data received in the destination. In our experiments, we compare ESR with AODV. Figure 4 shows that ESR performs better than AODV in network throughput with the increase of the movement speed of nodes and the hop distance. When the topological rate changes not much, and the hop distance grows, high *Re-catch* success ratio guarantees data delivery. If *Re-catch* fails, a limited directional flooding mechanism of ESR makes it more possible chance to create the route still existed in those old on-route nodes. Hence unsent data messages can be transmitted before the timer expires. On the other hand, AODV may cause more data loss due to too long waiting time spent on the transmission for error message and the new route re-flooding phase.

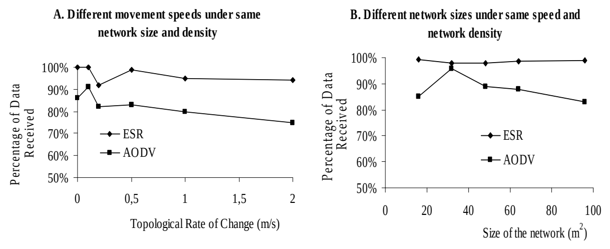

Figure 4 ESR and AODV with *Data Throughput*

3) Amount of Routing Message Analysis: Whether a routing protocol is energy efficient depends on total amount of routing control messages transmitted in the network. In our experiments, we compare ESR with AODV. Figure 5 shows that ESR performs better than AODV in control overhead with the increase of the movement speed of nodes and the hop distance. The reason is that *Re-catch* module recovers the broken link efficiently at the low amount of routing control messages. Even if *Re-catch* fails, the limited directional flooding also works well instead of re-flooding the whole network. On the other hand, AODV doesn't rely on MAC layer but sends periodical Hello messages, which increase routing overhead.

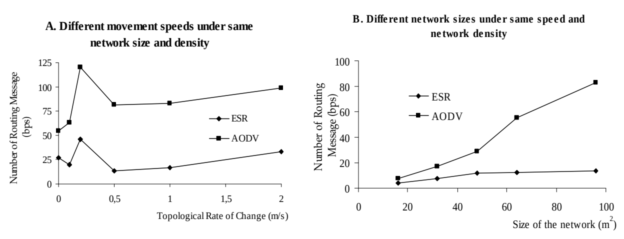

Figure 5 ESR and AODV with *Amount of Routing Message*

V. Conclusion

We presented our experience on implementation of ESR (EYES Source Routing protocol), an on-demand routing protocol for WSNs in our WSN environment using EYES sensor node prototypes using the LMAC on top of the AmbientRT operating system. In the implementation, LMAC transmits short ESR routing request message to reduce routing overhead, and also provides local topology information. The AmbientRT operating system enables real-time multi-tasking and the mutual exclusion mechanism avoids the concurrent use of un-shareable resources. Several interfaces in *Interface Layer* are used for communicating between ESR and LMAC. Further, we design the *Re-catch* module to recover the broken link efficiently, and the *Cut* module to shorten the redundant link. Beside these, packet and route table management performs the limited directional flooding mechanism.

Our experiments consist of ESR performance evaluation in *Re-catch* and *Re-request* success radio, and direct performance comparison between ESR and AODV in data throughput and the amount of routing control messages. The results show that ESR achieves much improved throughput and reduces power consumption on the routing control overhead performance when the topological rate of change, the size of the network increase. Furthermore, our experiments confirmed the fact that ESR can work collaboratively with LMAC and the AmbientRT useful and effective for implementing new protocol.

The future work will focus on extending ESR applications. We also plan to compare the performance of ESR with other routing protocol implementation in more complex and large network setup.

References

[1] D. Johnson, Y. Hu and D. Maltz. "The dynamic Source Routing Protocol for Mobile Ad Hoc Networks", IETF Internet draft, Apr 2003.

[2] C. Perkins, E. Royer, S. Das, "Ad Hoc On Demand Distance Vector (AODV) Routing", IETF Internet draft, Feb 2003.

[3] European EYES project (IST-2001-34734), http://eyes.eu.org.

[4] J. Wu, P. Havinga, S. Dulman, T. Nieberg, "EYES Source Routing Protocol for Wireless Sensor networks", European Workshop on Wireless Sensor Networks EWSN, January 2004.

[5] L. van Hoesel and P. Havinga. "A lightweight medium access protocol (LMAC) for wireless sensor networks", 1st Int. Workshop on Networked Sensing Systems (INSS 2004), Tokyo, Japan, June 2004.

[6] T. Hofmeijer, S. Dulman, P. G. Jansen, and P. J. M. Havinga, "AmbientRT - real time system software support for data centric sensor networks," in 2nd Int. Conf. on Intelligent Sensors, Sensor Networks and Information Processing. Melbourne, Australia: IEEE Computer Society, Washington, DC, Dec 2004, pp. 61–66.