Improving routing performance when several routing protocols are used sequentially in a WSN

Nancy El Rachkidy, Alexandre Guitton, Michel Misson

Improving routing performance when several routing protocols are used sequentially in a WSN

Nancy El Rachkidy[1,2], Alexandre Guitton[1,2], Michel Misson[3,2]
(1) Clermont Université, Université Blaise Pascal, LIMOS, BP 10448, F-63000 Clermont-Ferrand, France
(2) CNRS, UMR 6158, LIMOS, F-63173 Aubière, France
(3) Clermont Université, Université d'Auvergne, LIMOS, BP 10448, F-63000 Clermont-Ferrand, France
Emails:{nancy,guitton,misson}@sancy.univ-bpclermont.fr

***Abstract*—Wireless sensor networks can accommodate multiple applications by using a multi-stack architecture in order to deliver a large number of QoS. Multi-stack architectures can be optimized by allowing packet exchanges between stacks. However, routing loops may appear because of these exchanges. In this paper, we highlight the problem of routing loops generated when the same packet is routed according to two routing protocols. We define the delayable property of routing protocols by considering that some nodes might hold packets in order to avoid loops in the network. We show that minimizing the number of such nodes is an NP-complete problem. Then, we propose two heuristics to address this issue: a centralized deterministic heuristic requiring a global knowledge of the network, and a distributed stochastic heuristic reducing the number of hops from source to destination. Our two heuristics show important benefits: we reach a gain of up to 67% for the first heuristic and of up to 53% for the second heuristic, in terms of number of hops.**

I. Introduction

Wireless sensor networks (WSNs) are used for monitoring applications. The current trend of WSN is focusing on supporting simultaneously several applications in order to guarantee multiple levels of quality of service (QoS). A mono-stack WSN architecture, composed of a single combination $(\mathcal{M}, \mathcal{R})$ of a MAC protocol $\mathcal{M}$ and a routing protocol $\mathcal{R}$, cannot be adapted to fit all QoS requirements [1], [2]. However, multi-stack WSN architectures [3] can leverage QoS management.

A multi-stack WSN architecture integrates n combinations $(\mathcal{M}_i, \mathcal{R}_i)$, for $1 \leq i \leq n$. All nodes are synchronized: at a given time t, all nodes operate according to one combination. The combinations are periodically activated according to a specific schedule. At the application level, each packet is marked by a label i and processed by the combination $(\mathcal{M}_i, \mathcal{R}_i)$. The main drawback of these architectures is the dimensioning of the activity period of each combination. The dynamic dimensioning is not suitable for nodes dealing with limited resources. The static dimensioning may yield to unoptimized performance in terms of end-to-end delay and packet loss.

In [4], we overcome this dimensioning issue by proposing a queue-exchange mechanism that allows packets to be exchanged between the different combinations $(\mathcal{M}_i, \mathcal{R}_i)$ in order to enhance the network performances (namely, to reduce the end-to-end delay and to increase the throughput). The packet exchange consists in allowing packets marked by label i to be sent with combination $(\mathcal{M}_j, \mathcal{R}_j)$, with $i \neq j$. The main drawback of this cross-layering technique is that routing loops can occur in the network, because several routing protocols can be used to route the same packet. Such routing loops greatly penalize the performance, even if they are not frequent.

In this paper, we study a property of a set of routing protocols that we called delayability. Delayable routing protocols can be used jointly while avoiding routing loops, by allowing nodes to hold packets instead of forwarding them when a given condition is verified. Delayable protocols ensure that the network is loop-free, but yields to an increase of the hop count from source to destination, which in turn increases the delay. Thus, our goal in this paper is to reduce the number of nodes that hold packets for a given destination. We propose two approaches: a deterministic heuristic that requires a global knowledge of the network topology, and a stochastic heuristic that can be implemented easily on WSN nodes.

The remainder of this paper is organized as follows. Section II briefly describes the routing protocols that are used later as examples in our simulations and some related works. In Section III, we define the delayability property of routing protocols, and show that minimizing the number of nodes holding packets is an NP-hard problem. Then, we describe two heuristics for this problem. Section IV highlights the benefits of our heuristics. Finally, Section V concludes our paper.

II. State of the art

This section first describes the four routing protocols mostly used in WSN, then highlights the use of multi-stack architectures in the literature.

A. Routing protocols

1) Hierarchical tree routing protocol: The hierarchical tree routing protocol proposed in ZigBee [5] is referred to as the tree protocol in the remainder of this paper. The communication routes follow the links of a tree: only parent-child relationships are authorized. The tree protocol allows high energy-savings [6]. Indeed, the routing decision can be made without exchanging routing tables between nodes. Thus, the control overhead is limited to the tree maintenance. Moreover, the energy overhead is limited: when a node n is active, only its parent and children have to be active, while the other potential neighbors can switch to sleep mode. However, the tree protocol produces non-optimal paths in terms of hop-count.

2) Shortcut tree routing protocol: The shortcut tree routing protocol [7], referred to as the shortcut protocol in the following, enhances the tree protocol by using the knowledge of one-hop neighbors. Using the shortcut protocol, a node n forwards the packet to the neighbor providing the smallest expected number of hops according to the tree distance. The shortcut protocol always reaches the destination with less hops than the tree protocol. However, all the neighbors of a node have to be active when it has to transmit a packet, which increases the energy consumption.

3) Shortest path routing protocols: Shortest path routing protocols are based on optimal paths in terms of number of hops. From the knowledge of the whole network topology, each node is able to compute the shortest path from itself to the destination, and to forward packets accordingly. Such protocols include AODV [8].

4) OLSR: Optimized Link-State Routing (OLSR) protocol [9] is a routing protocol designed for mobile ad-hoc networks. It is a link-state routing protocol based on the concept of multipoint relays (MPRs). MPRs form a subset of one-hop neighbors of a node n that are in range of all two-hop neighbors of n. When n has to send a packet, it sends it to one of its MPRs, which can in turn forward the packet to a two-hop neighbor, which forwards it to the destination. Instead of requiring all the neighbors of n to be active when n transmits a packet, OLSR only requires the MPRs of n to be active.

B. Multi-stack architectures

Multi-stack architectures in WSNs are used in order to guarantee several QoS for several applications. These architectures consist in integrating several combinations of MAC and routing protocols. They use a time schedule repeated cyclically. The time schedule is divided into p periods. During each period p_i, a MAC protocol $\mathcal{M}_i$ and a routing protocol $\mathcal{R}_i$ are active in order to forward packets marked i by the application (or by an upper layer providing QoS management). The purpose of these architectures is to benefit from several combinations $(\mathcal{M}_i, \mathcal{R}_i)$ in order to leverage QoS management. IEEE 802.15.4/ZigBee [10], [5] and OCARI [11] are some examples of multi-stack architectures. In IEEE 802.15.4/ZigBee, the MAC sublayer integrates two MAC protocols: slotted/unslotted CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) for the contention access period, and TDMA (Time Division Multiple Access) for the guaranteed time slots. There is only one routing protocol used in a ZigBee network (it is either the tree protocol or AODV, depending on the network administrator). OCARI is a multi-stack architecture composed of MaCARI [12] as MAC sublayer in which both TDMA and CSMA/CA are used. The routing protocols used are the tree protocol for the scheduled activity period, and EOLSR [13] (which is an improvement of OLSR) for the unscheduled activity period.

The drawbacks of the multi-stack architecture concern the period dimensioning. Indeed, a period p_i dealing with a bursty traffic may yield to a high packet loss due to the overflow of queues. A period p_i dealing with a low traffic rate may waste time for other periods. In [4], a queue-exchange mechanism was proposed to overcome this problem. This mechanism consists in allowing packets marked i to be processed in period p_j (with $j \neq i$) and associated to the combination $(\mathcal{M}_j, \mathcal{R}_j)$. This is achieved by enabling $\mathcal{M}_j$ to retrieve packets from the proper queue. A special care has to be taken in order to ensure that the packets marked i, which were initially routed according to $\mathcal{R}_i$, can be routed according to $\mathcal{R}_j$. This queue-exchange mechanism is a cross-layering technique that optimizes the network performance in terms of end-to-end delay and throughput.

The queue-exchange mechanism may produce routing loops in the network, as a packet can be forwarded according to several different routing protocols, and thus be received by the same node several times. This yields to an increase of the path length between the source and the destination, and thus increases the end-to-end delay. In this paper, we focus on the routing loops and propose two heuristics that reduce their impact.

III. Proposition

When several routing protocols are used sequentially in a network, routing loops can occur (even if each protocol taken independently is loop-free). For instance, consider the example shown on Fig. 1 with a network of four nodes a, b, c and d, d being the destination of packets. Routing protocol $\mathcal{R}_1$ is identified by filled arrows, and routing protocol $\mathcal{R}_2$ by empty arrows. If node b decides to forward the packets to d according to $\mathcal{R}_1$, while node c decides to forward the packets to d according to $\mathcal{R}_2$, a routing loop occurs.

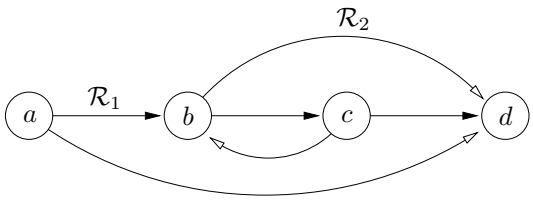

Figure 1. Routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$ might produce loops if each node decides arbitrarily to route packets according to $\mathcal{R}_1$ or $\mathcal{R}_2$.

A. Avoiding loops using p%-delayable protocols

The mechanism described in this subsection was proposed in [14] in order to avoid loops, provided that $\mathcal{R}_1$ and $\mathcal{R}_2$ verify specific conditions.

Definition 1 (Routing protocol)**.** *Given a directed graph $G = (V, E)$, a routing protocol $\mathcal{R}$ is defined as a function of $V \times V \rightarrow V$, such that $\mathcal{R}(n, d)$ is the next hop from node n to destination d. Note that the link $(n, \mathcal{R}(n, d))$ has to be in E. We assume that $\mathcal{R}(d, d) = d$ for any $d \in V$.*

Definition 2 (Decision function)**.** *Given a directed graph $G = (V, E)$, a destination $d \in V$, and a set of routing protocols $\{\mathcal{R}_i\}_i$, a decision function f_d is defined as a function of $V \rightarrow \mathbb{N}$ such that a node n is allowed to send a packet to*

destination d via next-hop $\mathcal{R}_i(n,d)$, for a given i, if and only if $f_d(\mathcal{R}_i(n,d)) < f_d(n)$.

When multiple routing protocols are used sequentially in a network, a node n having a packet for a destination d uses the decision function f_d to determine whether it can forward the packet without producing a loop. If the node is forbidden to send the packet, it simply waits for the routing protocol to change. We have shown in [14] that following this decision function ensures that no loops appear in the network[1]. However, two issues arise: (i) how to determine the decision function, and (ii) the decision function might be too conservative in the sense that it might forbid a node to send a packet that would not enter a loop. Addressing these two issues is the focus of this paper.

Definition 3 (p%-delayable protocols)**.** *Given a directed graph $G = (V, E)$, a destination $d \in V$, and two routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$, we say that these two protocols are p%-delayable protocols for the destination d and for function f_d if and only if the following condition is verified: $\min\{f(\mathcal{R}_1(n,d)), f(\mathcal{R}_2(n,d))\} < f_d(n)$ for all the nodes except for a percentage p; the p% nodes that do not verify the condition are called conflict nodes.*

If two routing protocols are 0%-delayable, no loops are produced in the network. If two routing protocols are p%-delayable (with $p > 0$), some of the nodes might produce routing loops.

B. Finding an optimal decision function is NP-hard

Definition 4 (Optimal decision function problem (ODFP))**.** *Given a directed graph $G = (V, E)$, a destination $d \in V$, a percentage p, and two routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$, ODFP consists in determining a decision function f_d such that $\mathcal{R}_1$ and $\mathcal{R}_2$ are p%-delayable.*

Theorem 1. *ODFP is NP-complete.*

Proof: We show that ODFP is NP-complete by reduction to the set cover problem (SCP), which is known to be NP-complete [15]. The proof is in three steps: (i) we show that ODFP is in NP, (ii) we show that SCP can be reduced to ODFP, (iii) we show that the reduction can be done in polynomial time.

First, ODFP is in NP. Indeed, to verify that a solution f_d to ODFP is valid, one must simply verify that function f_d satisfies the definition of p%-delayable routing protocols for all the nodes, except for the p% conflict nodes. This takes $\mathcal{O}(|V|)$, which is polynomial.

Second, let us reduce SCP to ODFP. Let $S = \{s_i\}_i$, and $U = \cup_i s_i$, be an arbitrary instance of SCP. Recall that the goal of SCP is to find a subset of S such that all elements of U are covered at least once. The reduction consists in building a directed graph where each s_i is mapped to a node, and each element of U represents a loop. Let us build the directed graph $G = (V, E)$ such that $V = \{n_i\}_i \cup \{d\}$, with node n_i mapped to set s_i. For each element l in U, we build a loop with $\mathcal{R}_1$-links in E. This loop goes through each node n_i such that $l \in s_i$. This transformation is shown on the left part of Fig. 2. As it is impossible (in ODFP) to have loops using only $\mathcal{R}_1$ (respectively $\mathcal{R}_2$), we now need to remove loops from E that use only $\mathcal{R}_1$-links (resp. $\mathcal{R}_2$-links). If a loop l appears in only one set s_i, we transform (arbitrarily) one of the links of the loop into an $\mathcal{R}_2$-link (all the other links are $\mathcal{R}_1$-link). Each other node is either outside a loop of $\mathcal{R}_1$-links, or has an out-degree $k \geq 2$. For each node n_i that has an out-degree $k \geq 2$ (using $\mathcal{R}_1$-links), we do the following: n_i is split into $k+2$ nodes, n_i^j (with $1 \leq j \leq k+2$). All incoming edges on n_i are connected to n_i^1. (n_i^1, n_i^2) is added to the $\mathcal{R}_1$-links, and (n_i^2, n_i^3) is added to the $\mathcal{R}_2$-links. For each outgoing link (n_i, n_m) of the initial n_i, there is an $\mathcal{R}_1$-link (n_i^{m+2}, n_m). Finally, for each $2 \leq j < k+2$, there is an $\mathcal{R}_2$-link (n_i^j, n_i^{j+1}). With this procedure, there are no loops using either $\mathcal{R}_1$-links or $\mathcal{R}_2$-links (due to the fact that the routing protocol changes between n_i^1 and n_i^3), and each node has at most two next-hops, one according to $\mathcal{R}_1$ and one according to $\mathcal{R}_2$. This transformation is shown on the right part of Fig. 2. Finally, if a node n_i does not have an $\mathcal{R}_1$-link (resp. $\mathcal{R}_2$-link), (n_i, d) is added to the $\mathcal{R}_1$-links (resp. $\mathcal{R}_2$-links).

To complete the reduction, we now have to prove that (i) if SCP has a solution for a given k, then ODFP has a solution for $p = 100.k/|S|$, and (ii) if SCP has no solution for a given k, then ODFP has no solution for $p = 100.k/|S|$.

- Let us assume that the SCP instance has a solution $S' = \{s'_1, \ldots, s'_k\} \subset S$. All the loops of the graph are covered by the set of k nodes $\{n'_i\}_i$ (each node n'_i corresponds to set s'_i). It is possible to build a function f_d that does not yield any conflict, except for these k nodes. Indeed, without these k nodes, the directed graph is acyclic (by definition), and f_d can be built according to a topological sort). This ensures that $f_d(\mathcal{R}_1(n,d)) < f_d(n)$ and $f_d(\mathcal{R}_2(n,d)) < f_d(n)$, which is equivalent to $\min\{f_d(\mathcal{R}_1(n,d)), f_d(\mathcal{R}_2(n,d))\} < f_d(n)$. $\mathcal{R}_1$ and $\mathcal{R}_2$ are p%-delayable with $p = 100.k/|S|$.
- Let us now assume that the SCP instance has no solution for a given k. In other words, it is not possible to cover all the loops (that is, all elements of U) with k nodes. Since each loop has to be covered at least once by a node in conflict (by definition), strictly more than k nodes have to be in conflict. The routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$ cannot be p%-delayable with $p \leq 100.k/|S|$, and ODFP has no solution.

Third, we have to show that the reduction of SCP to ODFP is polynomial. The computation of the graph before removing $\mathcal{R}_1$-links can be performed in $\mathcal{O}(|S|^2.|U|)$. The graph after removing $\mathcal{R}_1$-links loops can contain up to $|S|.|U|$ nodes: the transformation requires $\mathcal{O}(|S|.|U|)$. Overall, the transformation is polynomial. ■

[1]This property comes from the fact that the values of f_d are strictly decreasing on the path from n to d, and thus it is impossible for the same packet to reach twice the same node.

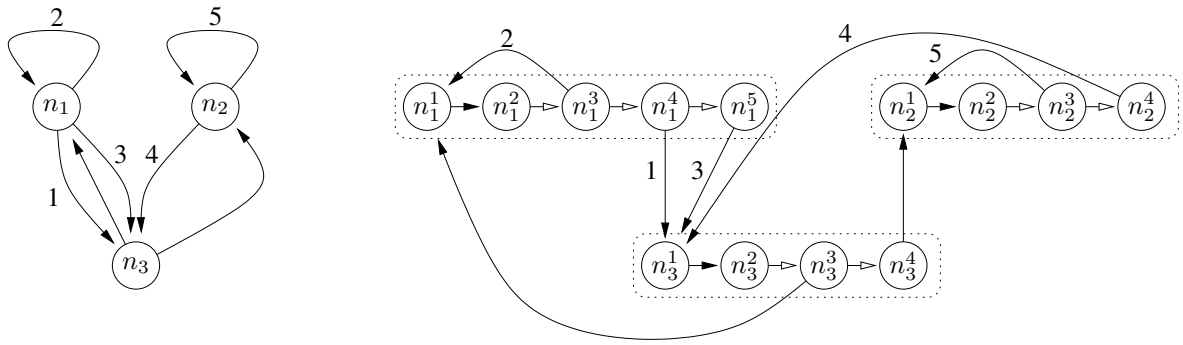

Figure 2. Transformation of the instance $\{\{1,2,3\},\{4,5\},\{1,3,4\}\}$ of SCP to an instance of ODFP, before removing $\mathcal{R}_1$-links loops (on the left), and after (on the right). Node d is not shown for clarity.

C. Topology modification heuristic

The topology modification heuristic (TMH) we propose is a deterministic heuristic that requires a global knowledge of the network. Let $f^G(n)$ be the hop count on graph G from n to node d, according to one of the two protocols (say $\mathcal{R}_1$)[2]. Let $G = (V, E)$ be the topology of the network. TMH computes the number of nodes in conflict in G according to f^G. Then, for each edge $e \in E$, the heuristic computes the number of nodes in conflict in $G_e = (V, E \backslash \{e\})$, provided that G_e is connected. Then, the heuristic chooses the graph $G_{e'}$ for which the number of nodes in conflict is minimum, and informs the routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$ that the edge e' should not be used[3]. The decision function used here is simply $f^{G_{e'}}$.

When TMH is used, one edge (at most) is logically removed from the topology. Even if this might slightly increase the distance for the routing protocols (because the routing protocols are not allowed to use the edge that has been removed), the objective is to greatly reduce the number of nodes in conflicts in this simplified topology. The main drawback of this heuristic is that it requires a global knowledge of the network, and also requires large processing capabilities for the centralized node that runs the heuristic.

D. Probabilistic delayable heuristic

The probabilistic delayable heuristic (PDH) is a stochastic heuristic that requires no knowledge of the topology. PDH allows nodes in conflict to decide whether they hold the packet (in order to avoid routing loops) or forward it according to the potential routing protocol $\mathcal{R}_i$. This decision is based on a probability ρ. For instance, $\rho = 0.25$ means that each time a node in conflict has a packet to process, it forwards the packet with a 25% probability. This heuristic assumes that some conflicts do not lead to routing loops, and that it is sometimes safe to send packets even though the node is in conflict. PDH may reduce the hop count per path as some nodes in conflict do not wait to forward the packet, but might also create routing loops (although not infinite due to the probabilistic nature of PDH).

IV. Results

This section first describes the parameter settings we used in our simulations and the metrics we considered. Then, it shows the benefits of our heuristics TMH and PDH.

A. Parameter settings

We consider a network of 100 nodes randomly distributed over a 100 m^2 area. We randomly choose one source s and one destination d in the network and study the path of packets from s to d. For simplicity reasons, we simulate the multi-stack architecture using two routing protocols $\mathcal{R}_1$ and $\mathcal{R}_2$. We consider four possible combinations based on the routing protocols described in Sect. II: (i) the tree protocol with a shortest path routing protocol, referred to as t-sp, (ii) the tree protocol with OLSR, referred to as t-OLSR, (iii) the shortcut tree routing protocol with a shortest path routing protocol, referred to as sc-sp, and (iv) the shortcut tree routing protocol with OLSR, referred to as sc-OLSR[4]. Time is divided into a repeated schedule. The schedule is divided into hop-periods, and is considered here to be the following: $\mathcal{R}_1$ is used during two hop-periods, and $\mathcal{R}_2$ is used during three hop-periods[5]. One hop-period corresponds to the time required to transmit a packet from a node to its next-hop. When a node holds a packet until the routing protocol changes, the node has to wait for as many hop-periods as remain before the routing protocol changes. Simulations are averaged over 100 repetitions.

We use the following two metrics: (i) the number of conflict nodes per path, which is defined as the number of nodes in conflict, on the path from the source to the destination, and (ii) the number of hops per path, which is the total number of hop-periods to route the packet from the source to the destination.

[2]The best results are obtained when the routing protocol having the longest hop count is used for f^G.

[3]TMH is also able to use graph G without removing any edge, if it corresponds to the minimum number of nodes in conflict.

[4]We do not consider pairs of protocols that do not yield routing loops [14].

[5]Having $\mathcal{R}_1$ last for less hop-periods than $\mathcal{R}_2$ corresponds to a scenario where the traffic for $\mathcal{R}_1$ is lower than the traffic for $\mathcal{R}_2$. Due to space constraints, we do not show results where $\mathcal{R}_1$ and $\mathcal{R}_2$ experience the same number of hop-periods.

B. Performance of routing combinations

Figure 3 shows the average number of conflict nodes per path as a function of the network density. The figure concerns the case where all conflict nodes hold packets in order to avoid routing loops. The decision function used is the distance on the tree, d_t. For all the combinations, the number of conflict nodes decreases when the network density increases. Indeed, an increase in the network density yields to a reduction of the number of hops per path and of the number of loops.

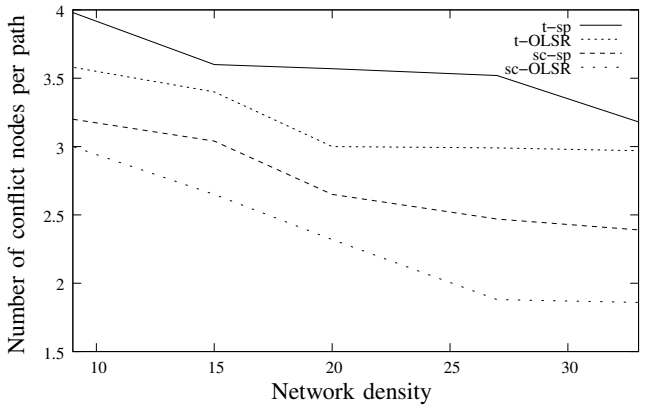

Figure 3. The average number of conflict nodes per path decreases when the network density increases.

Figure 4 shows the average number of hops as a function of the network density, when all conflict nodes hold packets. Again[6], the decision function used is d_t. The number of hops increases when a packet is sent from a node to another, but also when a node holds the packet. We can see that the number of hops decreases when the density increases: (i) when the density is high, each node has more routing options, and the path length decreases, and (ii) the number of conflict nodes is smaller (as shown on Fig. 3).

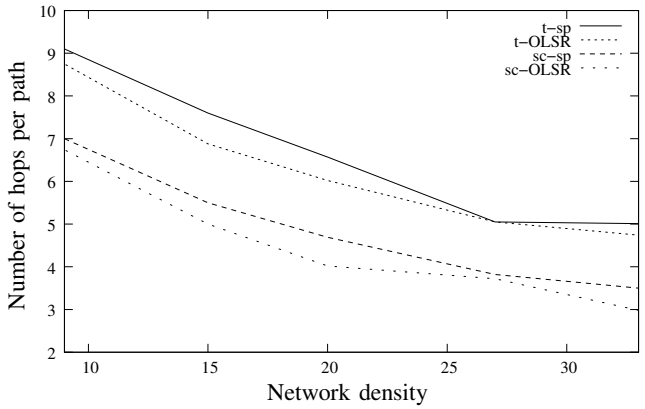

Figure 4. The average number of hops to reach the destination decreases when the network density increases.

[6]The same behavior is obtained when the shortest distance is used as the decision function.

C. Performance of TMH

Figure 5 illustrates the average number of conflict nodes when TMH is used, as a function of the network density. The number of conflict nodes decreases when the network density increases. Furthermore, the benefits of TMH with respect to the results of Fig. 3 are significant. We notice that when TMH is used, for combination t-sp, the gain reaches 62% for a density of 9 neighbors per node, and 68% for a density of 33. The gain reaches 57% for a density of 9 and 85% for a density of 33 for combination t-OLSR, and the minimum gain is 71% for both sc-sp and sc-OLSR combinations.

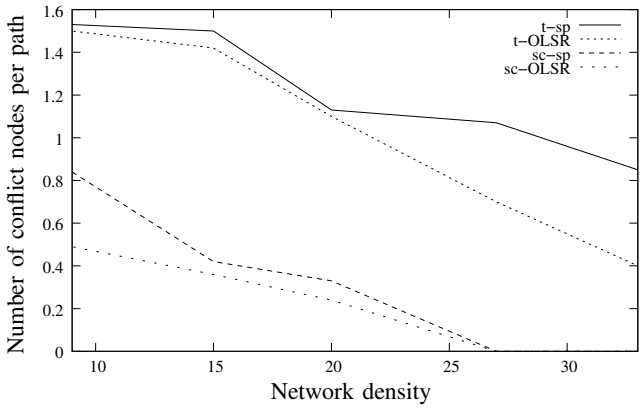

Figure 5. TMH greatly reduces the number of conflict nodes per path (by comparison with Fig. 3).

Figure 6 shows the average number of hops per path when TMH is used, as a function of the network density. We notice that the number of hops decreases when the density increases. Again, TMH is able to achieve great benefits. For combination t-sp, the gain varies between 34% (for a density of 9) and 50% (for a density of 33). For combination t-OLSR, the gain varies between 36% and 48%. For combinations sc-sp and combination sc-OLSR, the gain varies between 36% and 67%. These results show that even if TMH might slightly increase the path length (as it removes at most one edge from the logical topology), it can still significantly reduce the number of hops per path by reducing the number of nodes in conflict.

D. Performance of PDH

Figure 7 shows that the average number of conflict nodes when PDH is used, as a function of the probability parameter ρ, for a network density of 9 (which is rather large for a network of 100 nodes). The number of conflict nodes holding packets decreases as the probability ρ increases, by definition of ρ. It is interesting to note that the number of conflict nodes is not proportional to $1 - \rho$. Indeed, let us assume that when $\rho = 0$, a path with 5 nodes in conflict is chosen. With a probability $\rho = 0.5$, 50% of the 5 conflict nodes decide to forward the packet. In this case, the path chosen by these conflicting nodes might be completely different from the initial path, and have more nodes in conflict overall. For $\rho = 0.75$, we notice a gain of 27.6% for combination t-sp, 29.6% for combination t-OLSR, 25% for combination sc-sp, and 33.66%

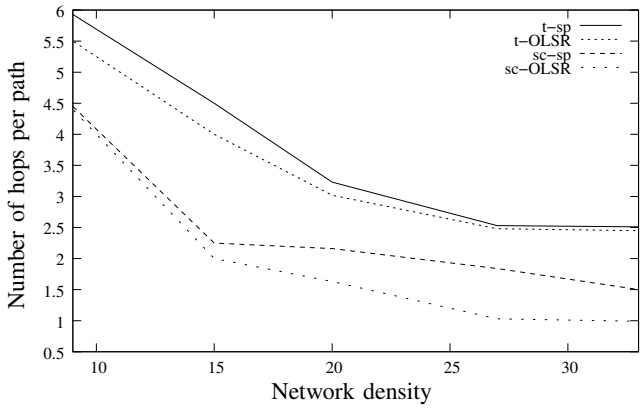

Figure 6. TMH greatly reduces the number of hops per path (by comparison with Fig. 4.

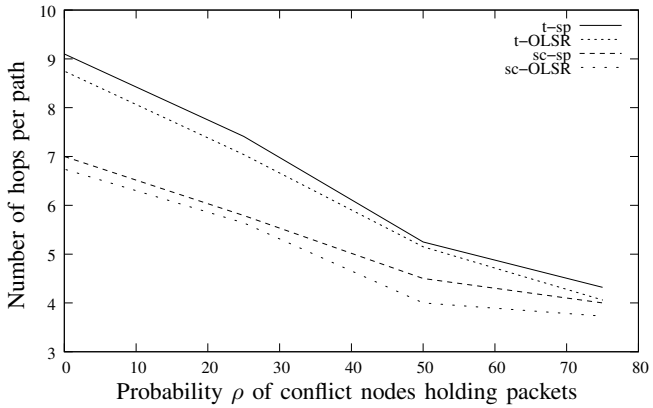

Figure 8. PDH reduces the number of hops per path (by comparison with density 9 of Fig. 4).

for combination sc-OLSR compared to the density 9 of Fig. 7. When the density increases, the gains of PDH are even larger.

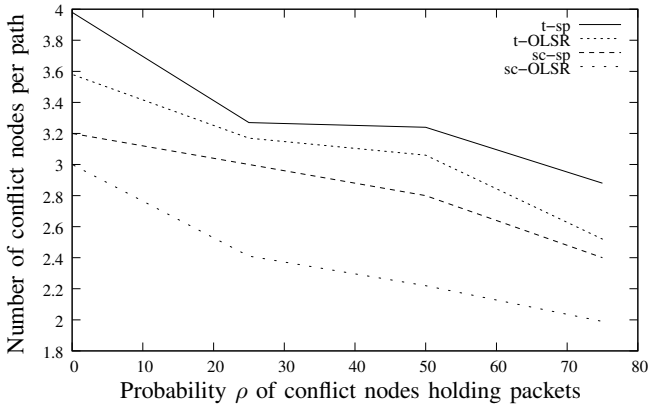

Figure 7. PDH reduces the number of conflict nodes per path (by comparison with density 9 of Fig. 3).

Figure 8 illustrates the average number of hops per path with PDH, as a function of ρ, for a network density of 9. The average number of hops consistently decreases as ρ increases, which is explained by the important reduction in the number of conflict nodes (and a small probability to increase path length due to routing loops). For $\rho = 0.75$, the gain of PDH reaches 52% for combination t-sp, 53% for combination t-OLSR, 42% for combination sc-sp, and 44% for combination sc-OLSR.

V. Conclusion

Queue-exchange algorithms in multi-stack architectures improve the overall performance, but might yield to routing loops. This paper highlighted the problem of routing loops, and introduced the delayability property of routing protocols used jointly in order to avoid routing loops. Delayable protocols are loop-free but tend to increase the number of hops on paths from source to destination. We showed that minimizing the hop count is an NP-complete problem, and we proposed two heuristics. TMH is a deterministic heuristic requiring a global knowledge of the network topology. PDH is a stochastic heuristic that can be easily implemented on WSN nodes. Both heuristics showed better performance in term of hop count (and thus, in term of delay). In our simulations, we obtained gains of up to 67% for TMH, and of up to 53% for PDH.

References

[1] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: A survey," *Computer networks*, vol. 47, pp. 445–487, 2005.

[2] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. Fun Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards," *Computer communications*, vol. 30, pp. 1655–1695, 2007.

[3] N. El Rachkidy, A. Guitton, and M. Misson, "Improving QoS in wireless sensor networks using a multi-stack architecture," in *IEEE Vehicular Technology Conference*, 2011.

[4] N. El Rachkidy, G. Chalhoub, A. Guitton, and M. Misson, "Queue-exchange mechanism to improve the QoS in a multi-stack architecture," in *PE-WASUN*. ACM, November 2011.

[5] ZigBee, "ZigBee Specification," ZigBee Standards Organization, Standard Zigbee 053474r17, January 2008.

[6] F. Cuomo, S. Della Luna, U. Monaco, and F. Melodia, "Routing in ZigBee: Benefits from exploiting the IEEE 802.15.4 association tree," in *IEEE International Conference on Communications (ICC)*, 2007, pp. 3271–3276.

[7] T. Kim, D. Kim, N. Park, S.-E. Yoo, and T. S. López, "Shortcut tree routing in ZigBee networks," in *ISWPC*, 2007.

[8] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on-demand distance vector (AODV) routing," IETF, Request For Comments 3561, July 2003.

[9] C. Adjih, T. Clausen, P. Jacquet, A. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol," IETF, RFC 3626, 2003.

[10] IEEE 802.15, "Part 15.4: Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs)," ANSI/IEEE, Standard 802.15.4 R2006, 2006.

[11] K. Al Agha, M. Bertin, T. Dang, A. Guitton, P. Minet, T. Val, and J. Viollet, "Which wireless technology for industrial wireless sensor networks? The development of OCARI technology," *IEEE Transactions on Industrial Electronics*, 2009.

[12] G. Chalhoub, A. Guitton, and M. Misson, "MAC specifications for a WPAN allowing both energy saving and guaranted delay - Part A: MaCARI: a synchronized tree-based MAC protocol," in *IFIP WSAN*, 2008.

[13] S. Mahfoudh and P. Minet, "EOLSR: an energy efficient routing protocol in wireless ad hoc and sensor networks," *Journal of Interconnection Networks*, vol. 9, no. 4, 2008.

[14] N. El Rachkidy, "Cross-layering et routage dans un réseau ad-hoc : Politique de relais de trame dans un réseau de capteurs sans fil selon une topologie en arbre," Ph.D. dissertation, Université Blaise Pascal, 2011, in French.

[15] R. M. Karp, "Reducibility among combinatorial problems," *Complexity of Computer Computations*, pp. 85–103, 1972.