On the Impact of Dynamic Routing Metrics on a Geographic Protocol for WSNs

Davi Resner, Gustavo Medeiros de Araujo and Antônio Augusto Fröhlich
Software/Hardware Integration Lab
Federal University of Santa Catarina
PO Box 476, 88040-900 - Florianópolis, SC, Brazil
{davir,gustavo,guto}@lisha.ufsc.br

***Abstract*—Message routing is a keystone of Wireless Sensor Networks (WSN). In many routing algorithms, sensor nodes forward data in a multicast fashion and a specific routing metric defines which of the receiving nodes is the next forwarder. If this metric is static, optimal routes tend to be overused and nodes in that route die quickly. The Trustful Space-Time Protocol (TSTP) is a cross-layer protocol designed to deliver authenticated, encrypted, timed, and georeferenced messages containing SI-compliant data in a resource-efficient way. It defines a novel, data-centric paradigm for programming WSNs and the IoT. In this paper, we discuss and evaluate dynamic routing in TSTP, which can explore routing metrics naturally given by the protocol's cross-layer, geo-oriented design, such as distance to the destination and message TTL (Time-to-Live). We introduce the idea of spatial distortion to map virtually any metric into a time-to-transmit offset, and evaluate the impact of different combinations of metrics. We assess the spatial distortion metrics on a TSTP implementation over IEEE 802.15.4 on the OMNET++ simulator, comparing their impact on the network in terms of end-to-end delay, fairness index, and delivery ratio.**

Keywords*—Wireless Sensor Networks, Geographic Routing, Cross-Layer Protocol.

I. Introduction

Wireless Sensor Networks (WSNs) have been the focus of intense research for well over a decade by now. Several physical layers have been proposed, along with a myriad of medium access and routing protocols. Such protocols have been made energy-aware; aggregation and fusion strategies have been employed; basic infrastructures have been enriched with location, timing and security protocols; operating systems have been designed to support higher-level abstractions, along with large-scale management systems designed to handle the produced data properly. We are currently seeing these networks being connected to the Internet of Things (IoT).

In this paper, we discuss the impact of different routing metrics in the scope of the *Trustful Space-Time Protocol* (TSTP) [1]: an application-oriented, cross-layer communication protocol initially developed for the Embedded Parallel Operating System (EPOS). TSTP was designed to resource-efficiently deliver authenticated, encrypted, timed, geo-referenced, SI-compliant data communication support to IoT devices interacting with an IoT gateway. Actually, it reaches beyond a communication protocol as it defines a user interface inspired by the IEEE 1451 Smart Transducer concept of "transducer electronic data sheets". Applications simply declare interest in a given physical quantity inside a portion of space-time that is to be measured with a minimum precision and at a given frequency. Nodes matching the criteria periodically send the corresponding data that is selectively forwarded to the gateway.

By using spatial coordinates as the network address itself, TSTP implicitly causes these coordinates to be constantly transmitted throughout the network in every message, enabling passive location estimation via trilateration [1] with no insertion of control messages in most cases. It also allows geographic routing strategies to be incorporated naturally at the Medium Access Control (MAC) level in a fully distributed and dynamic way. We show how we can apply the concept of spatial distortion to make TSTP route messages not only towards the destination, but also taking into account the Time To Live (TTL) of messages, frequency of transmission, energy level, and virtually any metric of interest. We run simulations to analyze the impact of two spatial distortion metrics (TTL and effort) on the network, and show that different metrics successfully enhance distinct aspects of network performance.

The rest of this work is organized as follows: Section II overviews characteristics of TSTP concerning message routing and introduces the idea of spatial distortion. Section III explains the simulated scenarios and discusses the results. Section IV discusses related work, and in Section V the final remarks and directions for future work are presented.

II. TSTP and Distributed Spatial Distortion Algorithm

Cross-layer designs have been shown to be very efficient in optimizing wireless networks [2]. In practice, cross-layer designs usually work by taking information from one or more layers of a typical layered stack to optimize a set of parameters or make a decision in another layer or set of layers [3]. From the myriad of cross-layer proposals, a minority involves the application layer, and even fewer encompass the complete stack to present a truly application-oriented, domain-specific solution.

The *Trustful Space-Time Protocol* (TSTP) [1] is an application-oriented, cross-layer protocol for WSNs and the IoT. Instead of focusing on keeping the original protocol interfaces in a modular, layered architecture with shared data, TSTP focuses on efficiently delivering functionality recurrently needed by WSN applications: trusted, timed, geo-referenced, SI-compliant data that is resource-efficiently delivered to a sink. TSTP delivers this functionality directly to the application in the form of a complete communication solution, which allows the design of optimized, synergistic co-operation of protocols while eliminating the need for additional, heterogeneous software layers that come with an integration cost and often result in replication of data.

TSTP is composed of 6 actors closely integrated in a cross-layer architecture:

- **Time Manager:** Responsible for precise clock synchronization across the network [4];
- **Locator:** Responsible for keeping spatial coordinates up to date, particularly in nodes devoid of a GPS receiver;
- **Router:** Responsible for defining and implementing the metrics for relay selection used by the MAC when forwarding messages;
- **Security Manager:** Responsible for management of encryption, authentication and cryptographic keys [5];
- **MAC:** Responsible for efficiently managing the communication channel and disseminating information from and to actors across the network [6];
- A **Data-centric API**, through which applications can benefit from the implemented services.

In this work, we explore the interaction between the Router and the MAC components.

Figure 1 shows the header format present in every TSTP message [1], which is visible by every component. The header is composed mostly of information necessary to characterize the *data* being transmitted: *where* and *when* it was produced, and *where* and *until when* it should be consumed. Other fields characterize the node relaying the message, and serve to synchronize the nodes to one another in space and time [1] [4]. The first byte is used for classification of the message itself. Since most of the header is inherited from the *data* itself, it is natural in TSTP to use any of these fields as metrics for routing. In particular, TSTP MAC considers the "Last Hop x,y,z" field [6]. In this work, we analyze an enrichment to the routing mechanism by taking advantage of other information present, such as the "TTL" field.

A. TSTP MAC

TSTP MAC [6] follows the general principles of RB-MAC [7] (Section IV): senders send a long preamble composed of microframes before each message, such that at most one message occupies the channel at every period S (for a full period); sensor nodes sleep for most of the time and, when they receive a message, nodes closer to the destination become relay candidates and use their own distances to the message's destination to derive the time offset for Clear Channel Assessment (CCA) and transmission: the relay candidate closest to the destination assesses the channel earlier and wins the contention, resulting in a greedy, fully-reactive geographic routing. The offset δ for a message m at any given node is locally calculated as [6]:

$$\delta(m) = \left\lfloor \frac{|D - (D_m - R)|}{gR/S} \right\rfloor \times g \quad (1)$$

with D representing the current node's distance to the message's destination, D_m representing the message sender's distance to the destination, R representing a network-wide parameter corresponding to the radio range of the nodes, and g representing the time gap between a transmitter sensing an available channel and finishing transmission of enough symbols for other nodes to sense a busy channel (8 symbols, or 320μs in IEEE 802.15.4). This equation makes nodes closer to the destination wake up earlier, normalizes the offset to the period S, and ensures that, if the destination is extremely far away, neighboring nodes (that are at the same discrete point in space in the scale of the total distance) will still have different offsets.

TSTP MAC takes a toll on senders and on overall latency to reach ultra-low power consumption at receivers. Our implementation of TSTP MAC over an IEEE 802.15.4 2450MHz DSSS PHY layer uses values for the period S greater than 116ms to reach an idle listening duty cycle smaller than 1% [6].

Each node maintains a queue of messages, in this work referred to simply as Q. Each entry in the queue represents a message that is scheduled for transmission or retransmission. From any given entry Q_i in Q, the following information shall be directly accessible (e.g. in the TSTP header) or derivable:

New	An indication of whether this is a new message (generated by this node) or a forwarding;
TTL	The point in time until when the message must arrive at its final destination;
ID	The message's identification code;
m	The message itself;
$\delta(m)$	The offset time;
Dst	The destination's coordinates.

New messages (e.g. a sensor value read by this sensor node) are added to Q with the *New* indication. The *ID* is generated during insertion of new messages according to some criterion and kept the same for the entire lifetime of the message across the network.

TSTP MAC uses the notion of implicit acknowledgment (ACK). A node removes a message from its queue *when and only when* another message with the same ID is overheard in the network. This case means that another node already handled the forwarding of that message towards its destination. The only case when the ACK is explicit is when the message reaches its final destination node: that node must retransmit the same message, just to acknowledge the last forwarder and any neighbors that might have that message queued. This algorithm is explained in further detail in Subsection II-C.

B. Spatial Distortion

TSTP's transmission time offset, given by Equation 1, is made sensitive to other metrics by the Router according to application's needs. For example, it may take into account the remaining battery charge of the node to ensure that nodes in an optimal path are not going to have their batteries depleted too quickly [8]. To accomplish this, we use the notion that different metrics *bend space*, such that, for instance, nodes that are transmitting messages that are close to expiring may *become*

Header Format									
Bits: 3	**1**	**2**	**2**	**8**	**3*sb**	**64**	**3*sb + tb**	**64**	**0 or 32**
Message Type	Time Request	Spatial Scale	Temporal Scale	Location Confidence	Last Hop x,y,z	Last Hop Time-stamp	Origin x,y,z,t	TTL	Location Deviation

Fig. 1. TSTP message header format. *sb* and *tb* are variables defined by the *Spatial Scale* and *Temporal Scale* codes [1].

closer to the destination than they would on an Euclidean space, which in turn causes them to be more likely to win the contention and get the message to its destination sooner.

Equation 2 redefines the offset equation, introducing a distortion coefficient α:

$$\delta(m) = \left\lfloor \frac{\alpha \times |(D - (D_m - R))|}{gR/S} \right\rfloor \times g \tag{2}$$

Since the offset has an upper bound of the MAC period S, the distortion coefficient has an upper bound β, which would make Equation 2 evaluate to S:

$$\alpha \leq \beta = \frac{gR}{|(D - (D_m - R))| \times g} \tag{3}$$

The distortion coefficient defines how much will other metrics influence the perceived distance, and hence the offset used for contention. It may take into account virtually any metric of interest, as long as it is bounded by β. A value of $\alpha < 1$ decreases $\delta(m)$, increasing the node's likelihood of winning the contention, while $\alpha > 1$ has the opposite effect. Theoretically, an infinite number of metrics can be applied to the function defining α. In this work, we consider cases with a single metric (Section III) for better analysis of each chosen function. Equation 2 maintains the property that the offset is normalized to S, but contention winners are not necessarily nodes closer to the destination anymore: they may be the nodes with the most urgent message, the node with the most energy left, etc.. We consider two different metrics, explained next.

1) TTL metric: The TTL (Time To Live) metric defines a space influenced by how much time the message has left to reach its destination. The longer the time until the message expires, the more dilated the space is. Messages that are close to expiring make space shrink. Therefore, a node forwarding a message close to expiring will be closer (in this definition of space) to its destination, have a shorter offset $\delta(m)$, and have a better chance to win the contention and forward that message quickly. Equation 4 defines the distortion coefficient for the TTL metric.

$$\alpha = \left(\frac{TTL_m - St_m}{TTL_m} \right) \times \beta \tag{4}$$

where TTL_m is the time stamp in which message m expires, St_m is the time already spent since the message's generation time and β is the upper bound defined in Equation 3. Both TTL_m and St_m are taken from the message's header (Figure 1).

2) Effort metric: The effort metric distorts space based on how much the node has cooperated with the network doing message forwarding in the past. The more messages the node has forwarded, the more space is dilated and the less likely that node becomes to forward again. This metric seeks a balance on the network for neighbors on the same hop from the destination to better alternate the effort of forwarding messages. Equation 5 defines the distortion coefficient for the effort metric.

$$\alpha = \left(\frac{F_n}{Ts_n} \right) \times \beta \tag{5}$$

where F_n is the number of queued messages from neighbors (according to Algorithm 1, explained in Subsection II-C) that node n relayed. Ts_n is the total number of messages that the node has transmitted in the past, including its own generated messages. The equation is normalized to β, which is defined in Equation 3. The effort metric takes information from the node's history, rather than from information related to the message being forwarded.

C. TSTP Forwarding Algorithm

To prevent the distortion coefficient from causing messages to be forwarded to an incorrect destination, the TSTP Forwarding Algorithm (Algorithm 1) ensures that all messages that a node receives and are queued for transmission have the Greedy Forwarding Property: all messages relayed will make positive spatial progress (in an Euclidean space) towards the destination. The Greedy Forwarding Property can be written as $\forall j \forall i \{m_j i \in Q_i | D_i < D_j\}$, meaning that each message $m_j i$ from node j overheard by node i will be stored in node i's transmission queue Q_i if and only if the Euclidean distance D_i from i to the the message's destination is smaller than the Euclidean distance D_j from node j to the same destination.

Algorithm 1 TSTP Greedy Forwarding Algorithm

```
procedure GreedyForward(msg)
    isQueued ← false
    for each m_Q ∈ Q_i do
        if msg.id = m_Q.id then            ▷ message was already queued
            isQueued ← true
            if EuclideanDistance(thisNode, msg.destination)
 > EuclideanDistance(msg.lastHop, msg.destination)
 then
                delete msg                 ▷ ignore the incoming message
            else
                Q.erase(m_Q)               ▷ message has been acknowledged
            end if
        end if
    end for
    if isQueued = false then
        if EuclideanDistance(thisNode, msg.destination) >
 EuclideanDistance(msg.lastHop, msg.destination) then
            delete msg                     ▷ ignore the incoming message
        else
            Q.insert(msg)                  ▷ queue message for transmission
        end if
    end if
end procedure
```

The Greedy Forwarding Algorithm handles four possible cases:

1) Case 1 (lines 6, 7): If the message is already queued and it is coming from a node more distant to the destination, it must be a retransmission attempt, and the message must be ignored (remaining on the queue for later transmission).
2) Case 2 (lines 8, 9): If the message is already queued and it is coming from a node closer to the destination, this means that this message has already made positive progress in relation to the receiving node, and thus it is interpreted as an acknowledgment and the message shall be dropped from the queue.
3) Case 3 (lines 14, 15): If it is a new message and it came from a node closer to the destination, it means that this node does not make positive progress towards the destination, and thus the message must be ignored.
4) Case 4 (lines 16, 17): If it is a new message and it came from a node more distant to the destination, it means that this node makes positive progress towards the destination, and this node is a relay candidate for the message. The message is queued for transmission.

We can prove that the Greedy Forwarding Algorithm ensures the Greedy Forwarding Property as follows:

Theorem 1: The Greedy Forwarding Algorithm will store in queue Q_i any given incoming messages $m_j i$ from node j overheard by node i if and only if $D_i < D_j$.

Proof: The proof is by contradiction. Suppose that $\exists i, \exists j, \exists m_j i \{m_j i \in Q_i | D_i \geq D_j\}$ and that every Q is only altered by Algorithm 1. $m_j i$ must have been included in Q_i by Case 4, because it is the only case that includes messages in the queue. But Case 4 only includes a given message $m_j i$ in Q_i if $D_i < D_j$, and since by assumption $D_i \geq D_j$, it is not possible that $m_j i$ was included in Q_i by Case 4. Therefore, since Case 4 is the only case that could possibly include $m_j i$ in Q_i, it must be true that $m_j i \notin Q_i$, reaching a contradiction. ∎

III. Simulation Assessments

In order to assess the scalability of TSTP's routing strategy and the applied metrics over the distortion coefficient, we performed simulations in OMNET++[1]. We consider three different metrics: the Euclidean metric (Equation 1), the effort metric (Equation 5), and the TTL metric (Equation 4). Parameters that were applied to every simulation are presented in Table I. The nodes were always deployed in a grid layout, with every node sending messages to the same destination node, located in the center of the grid.

The Delivery Ratio (DR), End-to-End Delay (EED) and Fairness Index (FI) were measured to assess TSTP's forwarding mechanism and the impact of each proposed metric. The Fairness Index [9] is a theoretical metric which reflects how fair is a monitored parameter. In our simulations, the FI was used to measure how balanced was the amount of messages that were relayed by the nodes, according to Equation 6.

[1]https://omnetpp.org/

TABLE I. Simulation Parameters

Parameter	Value
Transmission Power	1.0 mW
Radio Sensitivity	-85dBm
MAC Period (S)	150ms
Number of Nodes	10 to 100 (step 10)
Propagation Model	Free Space Model
Deployment	Grid

TABLE II. Message Frequency for Control WSN scenario

Nodes (%)	Period / TTL (s)
10	1
20	10
20	60
50	300

$$FI = \frac{(\sum msgforwarded)^2}{Nnodes \times \sum msgforwarded^2} \quad (6)$$

where $msgforwarded$ is the number of message relayed, and $Nnodes$ is the number of relay nodes. FI is in the interval $[0, 1]$. The highest the value of FI, the more balanced the number of relayed messages is.

When a message's TTL expires before reaching the final destination, there are two options: drop the message immediately or set $\alpha = 0$, such that it will have the highest forwarding priority. The first treatment is appropriate to networks where it is not guaranteed that there is a greedy path between sender and destination. The second option might be used to boost delivery ratio on networks with a known topology. Since the grid layout makes sure that there is always a greedy path between any two nodes, we are able to simulate both cases. In Figures 2, 3, 4, 5, "w/ drop" refers to the first case.

A. Control WSN Scenario

The first simulated environment intends to depict a somewhat realistic traffic scenario for a control WSN. Each node generates a message periodically, with fixed period and TTL per node of either 1, 10, 60 or 300 seconds. Table II shows the proportion of nodes with each period and TTL.

Figure 2 shows that boosting priority of expired messages is able to keep Delivery Ratio at 100% up until a density limit. As all nodes are sending their own messages and relaying messages from others nodes, at around 60 nodes the demand for network resources increases beyond the point supported by the low-power MAC, causing high contention, frequent collisions, and a high rate of messages losses. By dropping expired messages, the network can be sustained for a higher density of nodes, with the cost of missing around 5% of messages that were within the network capacity in the less dense scenarios. We can also see that both metrics increase delivery ratio slightly when compared to pure Euclidean distance, and as expected, TTL delivers the most messages in most cases.

On the End-to-End Delay (EED) measurements (Figures 3, 4) we can point out a few observations. The first one is that as the network size increases, the EED increases too. This happens because the farther the nodes are from the destination, the more hops on average the messages need to travel. The second observation is that as the network density increases beyond 60 nodes, the message delays acutely

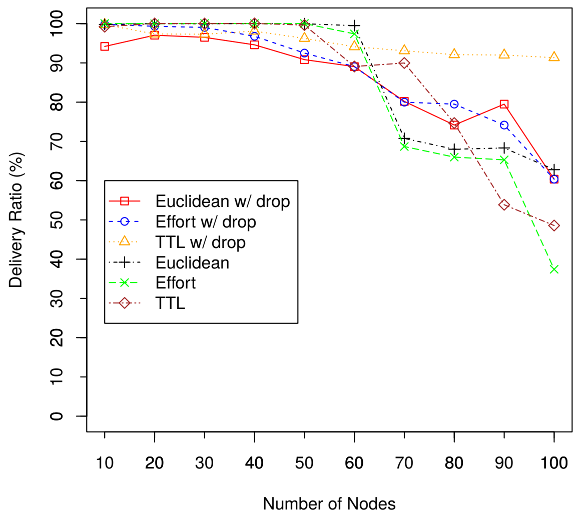

Fig. 2. Delivery Ratio for the control WSN scenario.

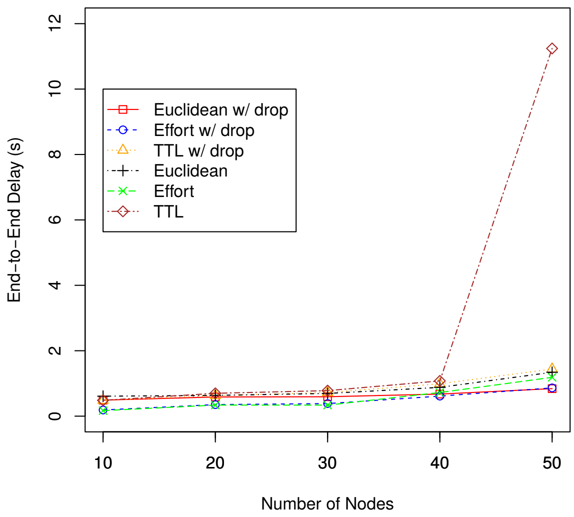

Fig. 4. Focus on End-to-End Delay up to 50 nodes for Control WSN scenario.

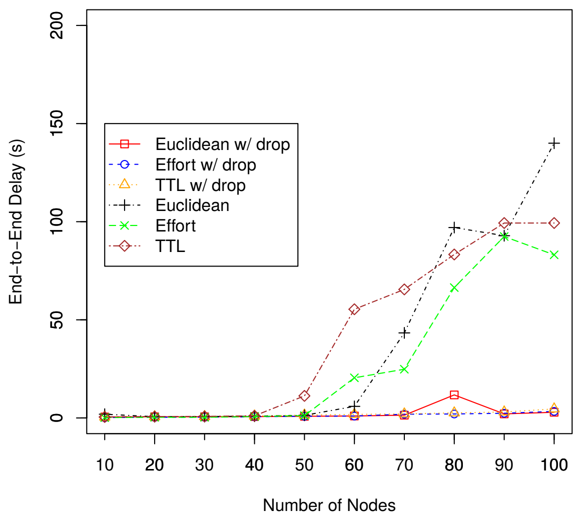

Fig. 3. End-to-End Delay for the Control WSN scenario.

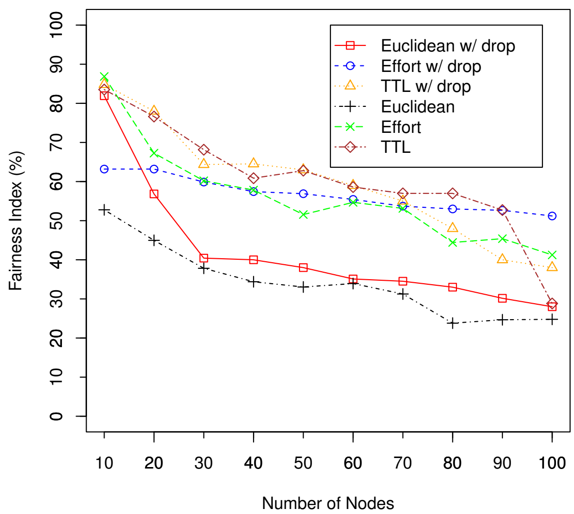

Fig. 5. Fairness Index for Control WSN scenario.

increase, indicating a network capacity saturation point. We attribute this fact to frequent collisions when a vast number of messages have the highest possible priority. Figure 4 presents a subset of the results from Figure 3, so we can see the EED effects when the network is not saturated. In this case, the EED is near 1 second for all simulations. The pure Euclidean and the effort metrics had the best performances regarding the EED. TTL might take more time to deliver a message because it often selects relay candidates that are not the closest possible to the destination, effectively increasing the number of hops. Moreover, dropping the expired messages results in smaller average EED, as expected.

The last measurements, shown in Figure 5, represent how fair was the network with each applied metric. As expected, the TTL and effort metrics reached the highest FI. This means that overall the network could better balance the work to relay messages with these metrics. It is possible to say that as the network shares the effort of relaying messages, it can in general achieve a higher lifetime. Another observation, which was also expected, is that the pure Euclidean metric presents a significantly more unfair effort to relay messages, even more as the network density increases, because nodes in an optimal path tend to be overused.

B. Busy Network Scenario

To better expose the impact of each metric, we perform another set of simulations with a more uniform and traffic-intense scenario. We use the same configurations, except for the message generation period: every node generates messages every 5 seconds. We repeat each simulation 25 times to account for random biases, and present the same measurements. In this environment where the network is operating closer to its limit, forwarding decisions have a bigger impact, and it is possible to see more clearly the advantages of each metrics.

Figure 6 shows that in the busy scenario, if the network is too sparse (10 to 20 nodes in this case), delivery ratio suffers. This is due to the fact that few neighbors will be available to forward data, which is specially harmful when every node

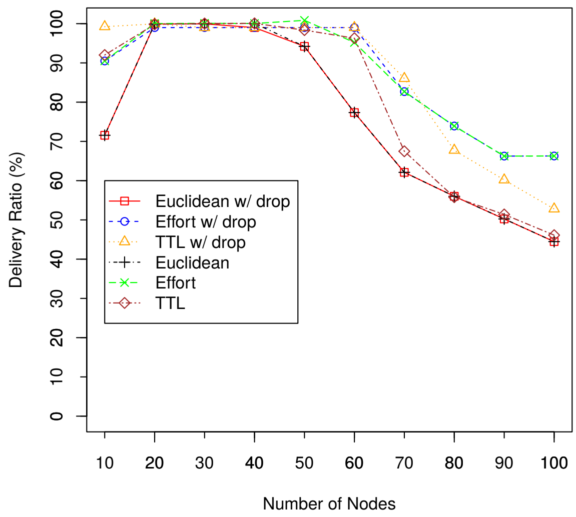

Fig. 6. Delivery Ratio for busy network scenario.

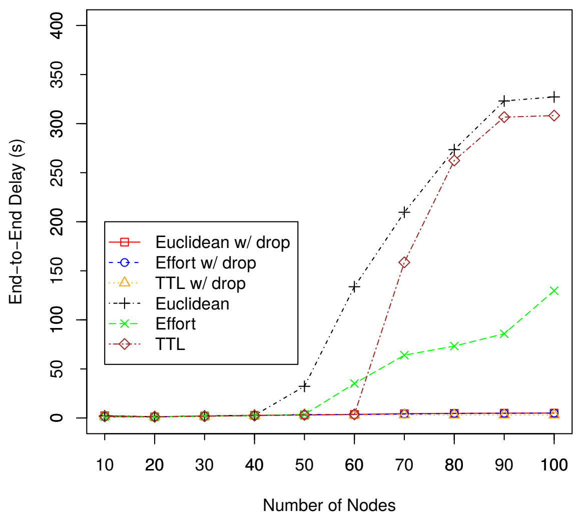

Fig. 7. End-to-End Delay for busy network scenario.

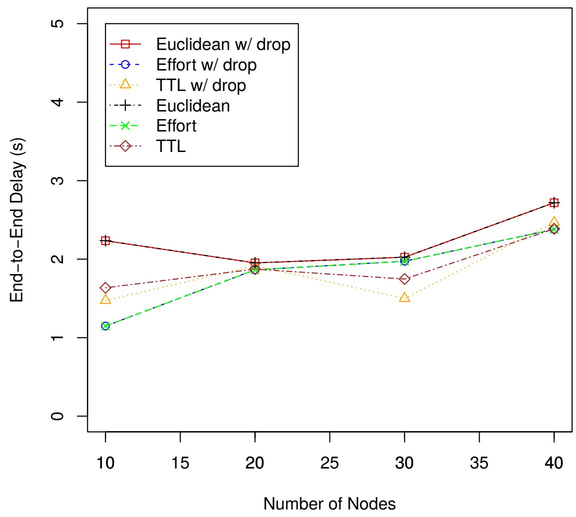

Fig. 8. Focus on End-to-End Delay up to 40 nodes for Control WSN scenario.

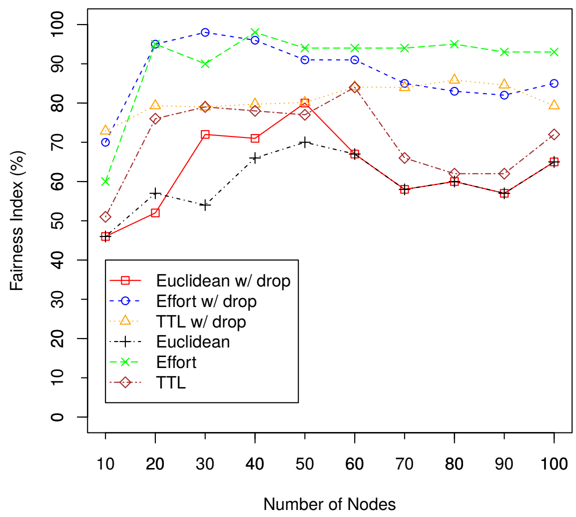

Fig. 9. Fairness Index for busy network scenario.

generates messages with similar frequency. Also, we can see that, before the saturation point, the TTL metric successfully performs the best at delivering messages before expiration, keeping the delivery ratio higher than 90% for up to a density of around 60 nodes. The effort metric is still superior to the Euclidean metric in terms of Delivery Ratio, because it tends to spread the traffic better between the nodes. In fact, for a more saturated network (75 to 100 nodes in this case), the better distribution of relays achieved by the TTL metric maintains the delivery ratio better than the other metrics.

The End-to-End Delay (Figures 7, 8) showed a behavior similar to the previous scenario (Figures 3, 4), but network saturation happens earlier. This time the TTL metric is successfully able to deliver packets faster than the other metrics.

As expected, the effort metric performed the best in most cases in terms of Fairness Index (Figure 9), successfully balancing the traffic by making nodes that are being used too often appear farther away. Again, the pure Euclidean metric overuses nodes in optimal paths, making the traffic less distributed. The TTL metric is an average case, and actually performed better than the effort metric for some specific cases on highly saturated networks.

IV. Related Work

TSTP MAC [6] is an adaptation and implementation of Receiver-Based MAC (RB-MAC) [7] under the TSTP context. RB-MAC defines the cooperative, distance-based geographic forwarder selection and passive acknowledgment mechanisms used initially in TSTP MAC. The authors of RB-MAC analyze the protocol in terms of energy consumption and latency, comparing it to 1-hopMAC, where senders define a specific receiver at each hop. They show that RB-MAC is in general more energy efficient and causes less latency because its anycast nature makes it much more resilient to lossy links [7]. Furthermore, not defining a receiver allows for dynamic and diverse contention metrics, although the authors do not investigate their impact as we do in the present work. In previous work [10], we implemented both RB-MAC and B-MAC under a Configurable MAC framework in the Embedded Parallel Operating System (EPOS), and showed that RB-MAC far outperforms B-MAC for varying channel conditions. In another work [1], we also analyzed the impact of the latency caused by RB-MAC in TSTP.

ADHOP (Ant-based Dynamic Hop Optimization Protocol) [8] is a routing algorithm based on Ant-Colony Optimizations (ACO) that target mobile, small-size and low-cost platforms, consuming small amounts of memory and processing power. In previous work [11], we replaced ADHOP's original latency-based pheromone evaporation heuristics with energy-aware ones based on battery charge and estimated node lifetime. Simulations compared the energy-aware versions of ADHOP to its original version, to AODV and to AOER. Results showed that the proposed approaches are able to balance network load among nodes, resulting in a lower number of failures due to battery depletion. The idea of pheromone evaporation heuristics in ADHOP is translated to the concept of spatial distortion in the geographic context of the present work.

Other MAC protocols, such as CMAC [12], use the same distance-based offset routing idea as TSTP and acknowledge that different routing metrics may be used instead of geographic distance. The authors do not study the impact of these metrics, however.

V. Final Remarks and Future Work

In this work, we introduced the idea of spatial distortion to the Trustful Space-Time Protocol, exploring different information present in TSTP messages to adjust geographic routing performance based on different parameters.

We ran simulations to study the impact of two different metrics on the network in terms of delivery ratio, fairness index, and end-to-end delay, under varying node densities. The results show that the TTL-aware spatial distortion is successfully able to increase delivery ratio, and the metric aware of number of transmissions is able to better spread the transmission of messages throughout different nodes of the network.

As future work, we intend to investigate the impact of more metrics, such as battery level and successful forward ratio; to study the impact of combining two or more metrics to distort Euclidean space; and to explore ways to incorporate a recovery mode to route messages across void regions, where a path exists between source and destination, but not a *greedy* path.

References

[1] D. Resner and A. A. Fröhlich, "Design Rationale of a Cross-layer, Trustful Space-Time Protocol for Wireless Sensor Networks," in *20th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2015).*, Luxembourg, Luxembourg, Sep. 2015, pp. 1–8. [Online]. Available: http://www.lisha.ufsc.br/pub/Resner_ETFA_2015.pdf

[2] V. Srivastava and M. Motani, "Cross-layer design: a survey and the road ahead," *Communications Magazine, IEEE*, vol. 43, no. 12, pp. 112–119, Dec 2005.

[3] B. Fu, Y. Xiao, H. Deng, and H. Zeng, "A survey of cross-layer designs in wireless networks," *Communications Surveys Tutorials, IEEE*, vol. 16, no. 1, pp. 110–126, First 2014.

[4] D. Resner, A. A. Fröhlich, and L. F. Wanner, "Speculative Precision Time Protocol: submicrosecond clock synchronization for the IoT," in *21th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA 2016). To appear*, Berlin, Germany, Sep. 2016.

[5] D. Resner and A. A. Fröhlich, "Key Establishment and Trustful Communication for the Internet of Things," in *4th International Conference on Sensor Networks (SENSORNETS 2015)*, Angers, France, Feb. 2015, pp. 197–206. [Online]. Available: http://www.lisha.ufsc.br/pub/Resner_SENSORNETS_2015.pdf

[6] D. Resner and A. A. Frohlich, "TSTP MAC: A Foundation for the Trustful Space-Time Protocol," in *14th IEEE/IFIP International Conference on Embedded and Ubiquitous Computing (EUC 2016). To appear*, Paris, France, Aug. 2016.

[7] M. Akhavan, T. Watteyne, and A. Aghvami, "Enhancing the performance of RPL using a Receiver-Based MAC protocol in lossy WSNs," in *IEEE ICT*, Ayia Napa, Cyprus, May 2011, pp. 191–194.

[8] A. M. Okazaki and A. A. Fröhlich, "ADHOP: an Energy Aware Routing Algorithm for Mobile Wireless Sensor Networks," in *IEEE SENSORS 2012*, Taipei, Taiwan, Oct. 2012. [Online]. Available: http://www.lisha.ufsc.br/pub/Okazaki_Sensors_2012.pdf

[9] R. Jain, D.-M. Chiu, and W. R. Hawe, *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*. Eastern Research Laboratory, Digital Equipment Corporation Hudson, MA, 1984, vol. 38.

[10] R. V. Steiner, M. R. Akhavan, A. A. Fröhlich, , and A. H. Aghvami, "Performance Evaluation of Receiver Based MAC Using Configurable Framework in WSNs," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Shanghai, China, Apr. 2013, pp. 884–889. [Online]. Available: http://www.lisha.ufsc.br/pub/Steiner_WCNC_2013.pdf

[11] A. Hoeller, P. Oliveira, and A. A. Fröhlich, "Dynamic Adjustment of Transmission Power of Mobile Ad-Hoc Wireless Sensor Networks," in *Brazilian Symposium on Computing Systems Engineering*, Niterói, Brazil, Nov. 2013, pp. 167–168. [Online]. Available: http://dx.doi.org/10.1109/SBESC.2013.41

[12] S. Liu, K. W. Fan, and P. Sinha, "Cmac: An energy efficient mac layer protocol using convergent packet forwarding for wireless sensor networks," in *2007 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, June 2007, pp. 11–20.