A filtering technique for n-way stream joins in wireless sensors networks

Submitted: 10/10/2018
Revised: 02/11/2018
Accepted: 07/11/2018

Boubekeur Djail[*]
Walid Khaled Hidouci[**]
Malik Loudini[***]

Abstract

Purpose – The join operations between data streams need more time and request more energy than traditional joins. In wireless sensor networks, energy is a critical factor. The survival of the network depends on this energy, thus it is necessary to consider, for this type of queries in such networks, the reduction of the sensors' energy consumption. While works that have been done to treat n-way join operations between data streams are rare so far, we propose a technique, named NSLSJ (N-way Stream Local Semi-Join) to perform this type of join operations. The principal aim is to considerably reduce the consumed energy.

Methodology/approach/design – The technique 'N-way Stream Local Semi-Join (*NSLSJ)* proposed in this paper is based on an in-network execution, and on filtering *tuples* strategy for an important gain in energy.

Findings – Compared to NSLJ and Sens-Join techniques, NSLSJ shows better performances in the realized tests as it consumes less energy.

Keywords: Joins query, wireless sensor networks, query processing, communication cost.

Introduction

A wireless sensor network is a wireless network using sensors that are characterized by their low cost, limited computing and storage capacity. Each sensor records the captured data as a local data table. Local table records constitute distributed tables; one table per type of records (Yao e Gehrke, 2003). Traditional relational operators are used for querying these databases.

[*]Researcher at the *Laboratoire de la Communication dans les Systèmes Informatiques* of the *Ecole Nationale Supérieure d'Informatique*, BP 68M, 16309, Oued-Smar, Alger, Algérie (http://www.esi.dz). Assistant Professor at *Ecole Supérieure en Sciences Appliquées d'Alger*. E-mail: b_djail@esi.dz.
[**]Professor of Computer Science at the Ecole Nationale Superieure d'Informatique (ESI), in Algiers, Algeria. E-mail: wk.hidouci@gmail.com.
[***]Professor of Automatic Control at the *Ecole Nationale Supérieure d'Informatique (ESI)*, in Algiers, Algeria. E-mail: m_loudini@esi.dz.

DJAIL, B; HIDOUCI, W. K.; LOURINI, M. *A filtering technique for n-way stream joins in wireless sensors networks*. **Law, State and Telecommunications Review**, Brasilia, v. 11, no. 1, p. 119-132, May 2019. DOI: https://doi.org/10.26512/lstr.v11i1.24853

Joins represent queries that are performed in wireless sensor networks by applications that require data from several nodes. In general, joins use a lot of energy. Joins on data streams need a lot more energy than joins on static tables.

Because data transmission consumes more energy than data processing in the node (Zhao e Guibas, 2004), it is necessary to decrease the transmitted data quantity during a join query to reduce the energy consumed by the sensors. Thereby, our aim is to propose solutions for minimizing energy consumed during a join execution, by reducing data transmission. Most of the existing techniques in this area have essentially targeted binary joins. Works on n-way joins are very rare.

We present in this paper an energy-efficient technique for n-way join queries between data streams in sensor networks.

General features of join queries in wireless sensor networks

Definitions

Let us consider two tables R and S.

Join of R and S is a query that concatenates the tuples of R and those of S, which verify a specified condition (join predicate). When the join predicate uses the equality operator, the join query is called equijoin. In wireless sensors networks, join queries are usually performed between two geographical regions.

An n-way join is the join performed between n tables, where n>2.

Semi-join of two tables R and S is the join performed between one of the two tables and the projection on an attribute of the other table.

Joins implementations in wireless sensors networks

A join query can be executed at sink. The tuples of tables are previously being transferred from nodes. This implementation is called extern-join.

A second implementation is when join query is executed at nodes in the network. This is the in-network implementation.

The extern-join implementation introduces an excessive consumption of energy due to the high amount of transmitted data. The in-network implementation, by contrast, does not involve high energy consumption, as the number of messages transmitted is reduced. It is the most efficient implementation.

Join classes in the wireless sensors network

In wireless sensors network, a join query can be executed between sensors of a unique region, or between sensors of multiples regions. In this last case join query is more common and known as inter-region join.

On the other hand, a join can be performed between static tables and is called one execution join, or between streams of data, in a continuous way, and is called continuous join. A continuous join that runs at regular intervals is said periodic join.

Related works

With the first proposed techniques to execute join queries, in wireless sensor networks, no filtering of data tuples is performed (Madden *et al.*, 2003; Yao e Gehrke, 2003; Bonfils e Bonnet, 2004; Abadi *et al.*, 2005; Chowdhary e Gupta, 2005; Pandit e Gupta, 2006; Mihaylov *et al.*, 2008; 2010). The main disadvantage is that the energy consumption is very high because of the large number of transmitted messages.

However, with recent techniques, joins queries are executed based on filtering tuples to eliminate records which do not participate in the result of the join query. So, the consumed energy will be more reduced. These filtering techniques have been designed mainly to deal with binary join queries (Coman e Nascimento, 2007) (Yu *et al.*, 2006) (Min *et al.*, 2011) (Yang *et al.*, 2007) (Stern *et al.*, 2010) (Lai *et al.*, 2008) (Lai *et al.*, 2010). Few works have considered n-way join queries.

Stern *and al.* (Stern *and al.*, 2009) proposed SENS-join, a technique to process all join types: binary and n-way joins. The join query is executed at the sink in several steps: First, the join attributes are transferred to the base station, in order to determine a filter, which is later broadcast in the network. Each site extracts the relevant tuples and transfers them to the sink, where finally the join query is performed.

SENS-join introduces too high energy consumption because of the high number of tuples to be transmitted to the well.

NLJ (N-way Local Join) (Djail *et al.*, 2016b) and NLSJ (N-way Local Semi-Join) (Djail *et al.*, 2016a) are the techniques that we proposed to execute n-way join queries between static tables.

NSLJ (N-way Stream Local Join) (Djail *et al.*, 2018) is a suggested technique to treat continuous n-way join queries in wireless sensors networks. NSLJ performs a join query locally at each step No filtering process is used by this technique, so its performances are bad.

For better efficiency in performing n-way join queries in wireless sensors network, we propose a new technique which uses semi-join to filter no-matching tuples. This technique is described in the following.

N-way Stream Local SEMI-Join (NSLSJ) technique

NSLSJ principle

In this paper, we present N-way Stream Local Semi-Join (NSLSJ), a technique to treat n-way join queries between data streams. Our aim is to efficiently decrease the consumed energy.

With NSLSJ, we are interested in executing continuous inter-region joins having syntax like this:

SELECT S1.attributes, S2.attributes,…,Sn.attributes
FROM S1, S2, …, Sn
WHERE predicate(S1) AND predicate(S2) … AND predicate(Sn)
AND join-expression (S1.join-attributes , S2.join-attributes ,…, Sn.join-attributes)

An example where this join type can be performed is the control of road traffic through several transit regions. A query that we can write is:

SELECT V1.V_Id, V1.time, V2.time, V3.time
FROM V1, V2, V3
WHERE (V1.time IN i1) and (V2.time in i2) and (V3.time in i3) and (V1.V_Id = V2. V_Id) and (V2. V_Id= V3. V_Id)

where:

- V1, V2, V3 are the relations (data streams)
- i1, i2 and i3 indicate the time ranges in which the vehicles crossed regions 1, 2 and 3, respectively.

In this example, the query selects the Id of a vehicle (the same value in each table), the times at which the vehicle passed in each region.

NSLSJ is based on in-network execution. Each join query is executed in a succession of intermediate joins. Each of these sub-joins is performed in a network node. The left linear trees technique was also adopted in order to define the execution sequence of the intermediate joins (Steinbrunn *et al.*, 1993). This order of execution is more accurately determined by considering the geographical positions of the participating regions at the join query, going from the first to the last region, by selecting at each step the nearest zone as the next zone.

Since we consider joins between data streams, we have adopted the principle of the 'distributed one-way stream join processed at destination' technique, proposed in (Tran e Lee, 2010). This technique uses, at each execution, tuples windows of every data stream. Each intermediate-join is performed at the node receiving end result.

The detailed description of NSLSJ technique indicates a three-phase execution:

Phase 1.

Initially, the query is diffused by the sink to the root node of each region. A location routing protocol such as GPSR : Greedy Perimeter Stateless Routing for wireless networks (Karp e Kung, 2000) (Ratnasamy *et al.*, 2002) is used to guarantee the reception of the query message by the concerned nodes. Additionally, each root node receives the tuples from the nodes of the region that it represents. For this, we assume that each node knows its position and the positions of its neighbors, by GPS : Global Positioning System or by localization algorithms (Savvides *et al.*, 2004).

Phase 2.

At the root node of each region, is maintained a window of tuples that were received from other nodes of the region. The window is noted W_{2i} for the i^{th} node. To avoid transmitting a projection of one tuple at every execution, we transmit to i^{th} region Si a projection noted K_{1i} of a set of tuples, noted B_{1i}.

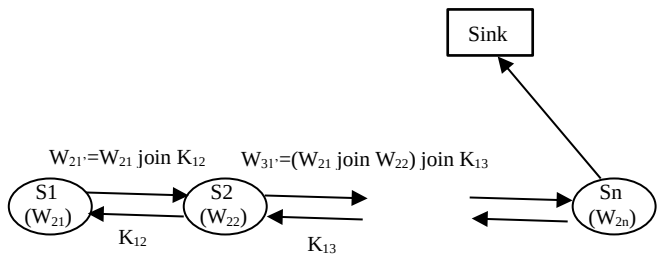
The following process is then performed:

A projection of B_{12}, K_{12}, is transmitted by region S1 to S2. A semi-join is performed between the current window and K_{12}. The result $W_{21'}$= (W_{21} join K_{12}) is then sent to the S2 region. In this region, the join operation between $W_{21'}$ and the window W_{22} maintained in this region is executed .

The projection of B_{13} (K_{13}) received in this region, will be used to determine the semi-join with the calculated result. The result of this semi-join is forwarded to the S3 region. The process is repeated between each stream Si and the following stream until the final result is determined at the last node Sn (Figure 1).

Phase 3.

The last result, at region Sn, is transmitted to the sink.



W_{2i} is the window maintained at the region Si.

K_{1i} is the projection of the window W2i on the join attributes.

Figure 1 – N-way Stream local Semi-Join (NSLSJ) principle.

NSLSJ example

We present in the following example an illustration of NSLSJ principle. We consider a join query between tree streams.

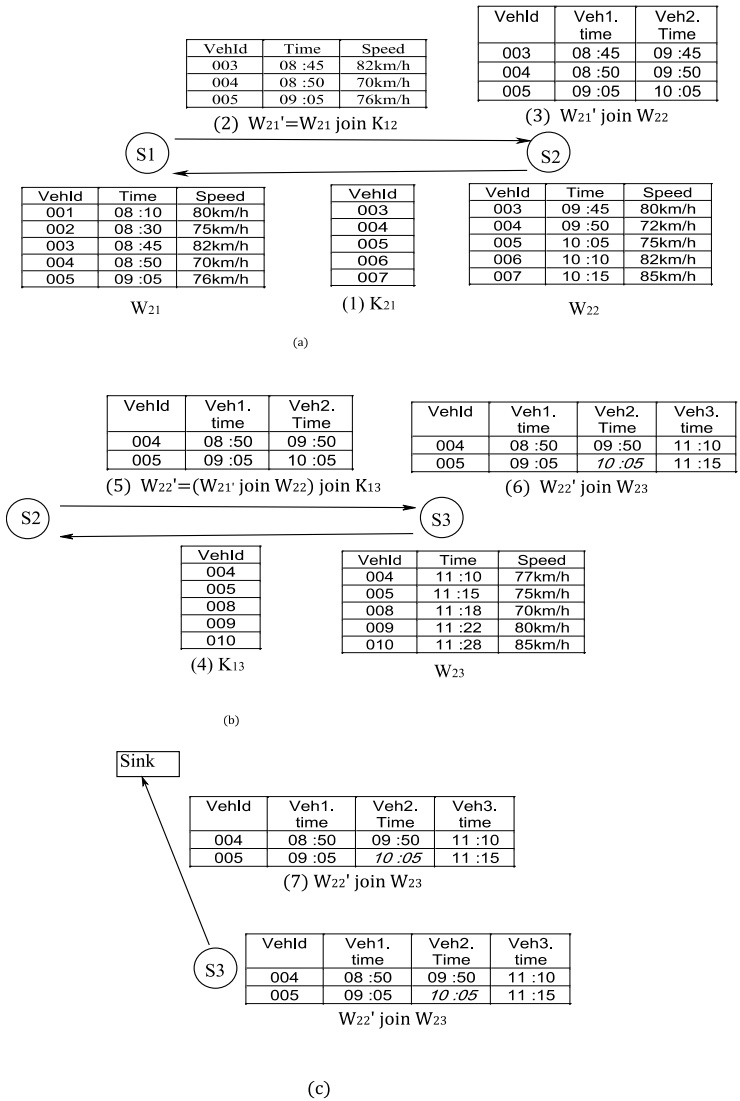


Figure 2 – An example of NSLSJ execution.

First, the join operation is executed between a stream region S1 and the nearest stream region S2. This phase is executed in four steps (Figure 2 (a)):

Step 1: A projection K_{12} of a set of S2 tuples is transmitted to the S1 region.

Step 2: A semi-join operation is performed between K_{12} and a window W_{21} maintained at the S1 stream region.

Step 3: The result of the semi-join operation $W_{21'}$ is forwarded to the S2 stream region.

Step 4: A join operation is accomplished between $W_{21'}$ and a window W_{22} of the S2 stream region.

In the second phase, the join operation is performed between the S2 stream region and the S3 stream region. The same steps as the first phase is executed (Figure 2 (b)).

At the end, the final result is forwarded to the sink. (Figure 2 (c)).

Performance analysis

Experiment environment

We accomplished our tests on the NS3 simulator.

We assumed the following:

- a tuple size equals 40 bytes,
- a column size equals 20 bytes,
- a result tuple size equals 30 bytes,
- a message size is similar to a tuple size.

We determined the communication following selectivity factors of the intermediates joins in the ranges $[10^{-5},10^{-4}]$ and $[10^{-4}, 10^{-3}]$. The values of selectivity factors are generated randomly.

Note that the selectivity factor of a join is the ratio of the tuples number of the join result by the product of the tuples number of the first relation by that of the second relation. The selectivity factor expresses the volume of the join result.

Two simulations were performed: one for 3 streams and another for 5 streams. In each simulation case, we compare our technique (NSLSJ) with our previously proposed technique (NSLJ) and with the SENS-join technique. We consider for this the number of transmitted messages as a measure of efficiency. The best technique will present the low number of diffused messages during its execution.

Experiment results

With values of the range [10^{-5}, 10^{-4}] for selectivity factors, (Chart. 1 and Chart. 2), 'N-way Stream Local Semi-Join' shows better performances than NSLJ and SENS-join whether with 3 streams or 5 streams. The number of transmitted messages is about 105 messages for NSLSJ, 102.5 for SENS-join and up to 120 for NSLJ during all the interval. Note that in this range, selectivity factors values are very low; less than 10^{-4}.

In the range [10^{-4}, 10^{-3}] of selectivity factors, (Chart. 3 and Chart. 4), NSLSJ continues to offer the best performance, but not in the same way than the previous range. With high values of selectivity factors, NSLSJ tend to decrease in performance. SENS-join keeps the same performances for both selectivity factor ranges.

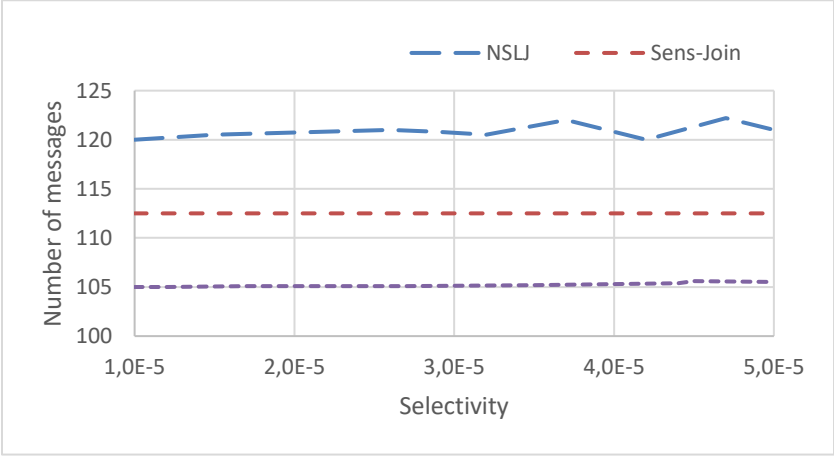


Chart 1 – Communication cost (as number of messages) *3 streams in the interval [10^{-5}, 10^{-4}]*

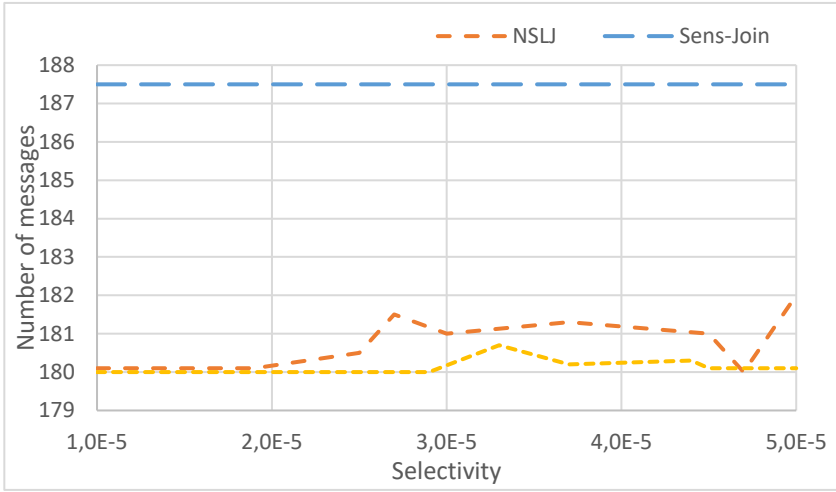


Chart 2 – Communication cost (as number of messages) for 5 streams in the interval [10^{-5}, 10^{-4}]

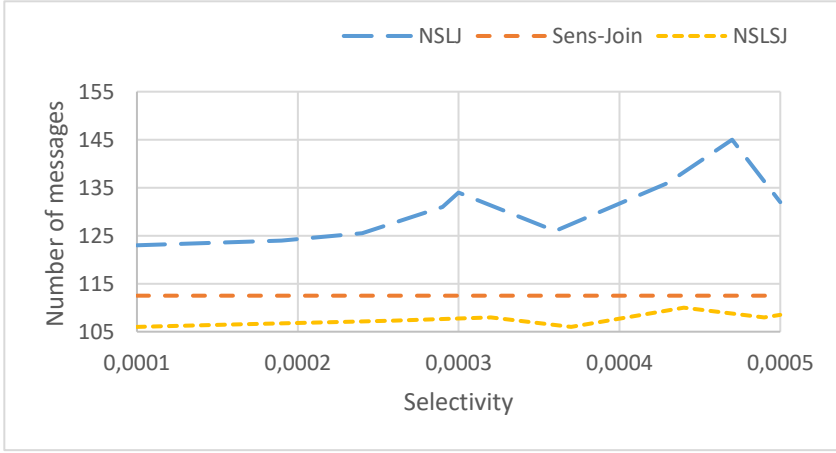


Chart 3 – Communication cost (as number of messages) for 3 streams in the interval [10^{-4}, 10^{-3}]

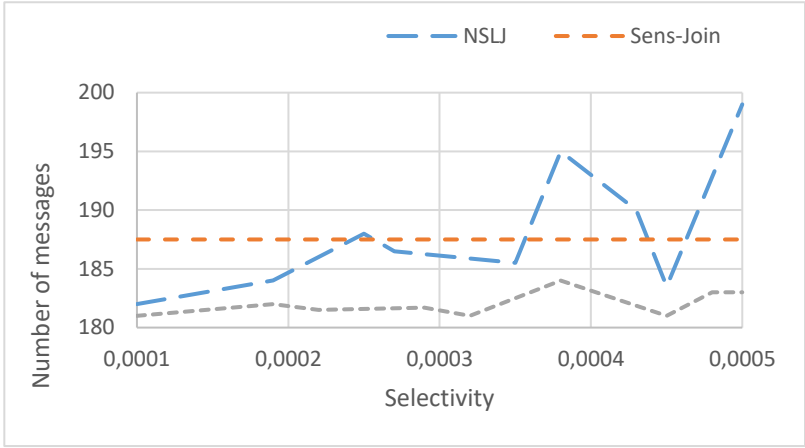


Chart 4 – Communication cost (as number of messages) for 5 streams in the interval [10^{-4}, 10^{-3}]

Discussion

For low values of selectivity factor, NSLSJ offers the best performance, given that it executes in in-network to minimize the quantity of transmitted messages to the sink and because, additionally, it uses the semi-join principle to more filter tuples of concerned tables.

SENS-join performs less than NSLSJ since it executes join query at the sink, whence it inducts a high energy consumption, because of the great quantity of transmitted messages. The filters determined and used by SENS-join remain insufficient.

For high values of selectivity factor, the two techniques show bad performances. In this case, the best way is to execute extern-join.

Conclusion

In this paper, we presented NSLSJ, an energy-efficient technique for n-way stream join execution.

In order to guarantee the best performance, NSLSJ use semi-join principle to filter the tuples that participate to the join query, and then to reduce the number of tuples transmitted during its execution. The in-network principle adopted additionally permits decreasing the quantity of transmitted messages at each step of the execution.

NSLSJ offers the more energy efficiency in comparison with SENS-join in the case of low values of selectivity factor.

In future works, we will try to adapt other solutions proposed for binary joins in sensor networks (such as synopsis join (Yu *et al.*, 2006)), in order to further reduce the number of transmissions in the network and allow better gain in sensors energy. We will also extend our solution to treat specific join queries. Recent works in this domain are those of Mo and al. (Mo *et al.*, 2014) for spatial queries and Kang and al (Kang, 2015) for iceberg joins.

References

ABADI, D. J.; MADDEN, S.; LINDNER, W. Reed: Robust, efficient filtering and event detection in sensor networks. Proceedings of the 31st international conference on Very large data bases, 2005, VLDB Endowment. p.769-780.

BONFILS, B. J.; BONNET, P. Adaptive and decentralized operator placement for in-network query processing. Telecommunication Systems, v. 26, n. 2-4, p. 389-409, 2004. ISSN 1018-4864.

CHOWDHARY, V.; GUPTA, H. Communication-efficient implementation of join in sensor networks. International Conference on Database Systems for Advanced Applications, 2005, Springer. p.447-460.

COMAN, A.; NASCIMENTO, M. A. A distributed algorithm for joins in sensor networks. Scientific and Statistical Database Management, 2007. SSBDM'07. 19th International Conference on, 2007, IEEE. p.27-27.

DJAIL, B.; HIDOUCI, K. W.; LOUDINI, M. N-way Local SemiJoin : A Filtering Technique for N-Way Joins in Wireless Sensors Networks. Journal of Electronic Systems, v. 6, n. 1, p. 7-16, 2016a. ISSN 2278 – 6538.

DJAIL, B.; HIDOUCI, K. W.; LOUDINI, M. A technique for n-way joins in wireless sensor networks. Database Systems Journal, v. 7, n. 2, p. 3-9, 2016b. ISSN 2069-3230.

DJAIL, B.; HIDOUCI, K. W.; LOUDINI, M. An energy-efficiency technique for n-way stream joins in wireless sensor networks. Nature & Technology Journal., v. Vol. A: Fundamental et Engineering Sciences, 18 p. 09-15, 2018.

KANG, H. In-Network Processing of an Iceberg Join Query in Wireless Sensor Networks Based on 2-Way Fragment Semijoins. Sensors, v. 15, n. 3, p. 6105-6132, 2015.

KARP, B.; KUNG, H.-T. GPSR: Greedy perimeter stateless routing for wireless networks. Proceedings of the 6th annual international conference on Mobile computing and networking, 2000, ACM. p.243-254.

LAI, Y.-X.; CHEN, Y.-L.; CHEN, H. PEJA: Progressive energy-efficient join processing for sensor networks. Journal of Computer Science and Technology, v. 23, n. 6, p. 957-972, 2008. ISSN 1000-9000.

LAI, Y.; LIN, Z.; GAO, X. SRJA: Iceberg Join Processing in Wireless Sensor Networks. 2010 2nd International Workshop on Database Technology and Applications, 2010, IEEE. p.1-4.

MADDEN, S. et al. The design of an acquisitional query processor for sensor networks. Proceedings of the 2003 ACM SIGMOD international conference on Management of data, 2003, ACM. p.491-502.

MIHAYLOV, S. R. et al. A substrate for in-network sensor data integration. Proceedings of the 5th workshop on Data management for sensor networks, 2008, ACM. p.35-41.

MIHAYLOV, S. R. et al.. Dynamic join optimization in multi-hop wireless sensor networks. Proceedings of the VLDB Endowment, v. 3, n. 1-2, p. 1279-1290, 2010. ISSN 2150-8097.

MIN, J.-K.; YANG, H.; CHUNG, C.-W. Cost based in-network join strategy in tree routing sensor networks. Information Sciences, v. 181, n. 16, p. 3443-3458, 2011. ISSN 0020-0255.

MO, S. et al. Multi-attribute join query processing in sensor networks. Journal of Networks, v. 9, n. 10, p. 2702-2712, 2014. ISSN 1796-2056.

PANDIT, A.; GUPTA, H. Communication-efficient implementation of range-joins in sensor networks. International Conference on Database Systems for Advanced Applications, 2006, Springer. p.859-869.

RATNASAMY, S. et al. GHT: a geographic hash table for data-centric storage. Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications, 2002, ACM. p.78-87.

SAVVIDES, A. et al. Localization in sensor networks. In: (Ed.). Wireless sensor networks: Springer, 2004. p.327-349.

STEINBRUNN, M.; MOERKOTTE, G.; KEMPER, A. Optimizing join orders. Citeseer, 1993.

STERN, M.; BÖHM, K.; BUCHMANN, E. Processing continuous join queries in sensor networks: a filtering approach. Proceedings of the 2010 ACM

SIGMOD International Conference on Management of data, 2010, ACM. p.267-278.

STERN, M.; BUCHMANN, E.; BÖHM, K. Towards efficient processing of general-purpose joins in sensor networks. 2009 IEEE 25th International Conference on Data Engineering, 2009, IEEE. p.126-137.

TRAN, T. M.; LEE, B. S. Distributed stream join query processing with semijoins. Distributed and Parallel Databases, v. 27, n. 3, p. 211-254, 2010. ISSN 0926-8782.

YANG, X. et al. In-network execution of monitoring queries in sensor networks. Proceedings of the 2007 ACM SIGMOD international conference on Management of data, 2007, ACM. p.521-532.

YAO, Y.; GEHRKE, J. Query Processing in Sensor Networks. CIDR, 2003. p.233-244.

YU, H.; LIM, E.-P.; ZHANG, J. On in-network synopsis join processing for sensor networks. 7th International Conference on Mobile Data Management (MDM'06), 2006, IEEE. p.32-32.

ZHAO, F.; GUIBAS, L. J. Wireless sensor networks: an information processing approach. Morgan Kaufmann, 2004. ISBN 1558609148.

DJAIL, B; HIDOUCI, W. K.; LOURINI, M. *A filtering technique for n-way stream joins in wireless sensors networks*. **Law, State and Telecommunications Review**, Brasilia, v. 11, no. 1, p. 119-132, May 2019. DOI: https://doi.org/10.26512/lstr.v11i1.24853