# Early Forest Fire Detection with Sensor Networks: Sliding Window Skylines Approach

Krešimir Pripužić, Hrvoje Belani, Marin Vuković

University of Zagreb, Faculty of Electrical Engineering and Computing,
Department of Telecommunications, Unska 3, HR-10000 Zagreb, Croatia
{kresimir.pripuzic, hrvoje.belani, marin.vukovic}@fer.hr

**Abstract.** Wireless sensor networks are widely used in environmental applications, like forest fire detection. Although forest fires occur relatively rarely, their number is increasing in Europe in the last years, so their manifestation must be early detected in order to prevent higher damages. To minimize needless communication between the sensor nodes for this usage, new data suppression technique using sliding window skylines is described in this paper. We experimentally evaluate our algorithm for continuous sliding windows skylines computation, and show its usability in practice.

**Keywords:** Data suppression, forest fire detection, sensor network, sliding window skylines, experimental evaluation.

## 1 Introduction

Sensor network technology is considered as one of the key technologies for 21st Century [1]. It is a network of spatially distributed autonomous devices with various sensors that cooperatively monitor certain conditions (e.g. physical or environmental) at different locations. Communication between sensors is wireless in most cases. A wireless sensor network (WSN) is the term for Low-Rate Wireless Personal Area Networks employing no fixed infrastructure and having communication links less than 10 meters in length and sensors centered on a subject or individual or in the targeted area [2].

WSN find their spread usage in a variety of applications: military, environmental, health, etc. The amount of data sent over the network can be very low, and the message latency can be on the order of minutes. Yet, costs must be low, and power consumption must be low enough for the entire network to last an entire required timeframe (e.g. season). These low-data-rate applications involve sensing of one form or another and require short-range links without a preexisting infrastructure on site, except network of sensors. Sensors can monitor conditions at different locations, such as temperature, relative humidity, wetness, lightning conditions, smoke, atmospheric pressure, wind speed and direction, etc [3].

According to the application type and design requirements, there are two basic types of WSNs: the one for rare event detection (e.g. forest fire detection or intrusion detection), and the other for periodic data gathering (e.g. temperature monitoring or biosensors). Although forest fire occurs relatively rarely, its detection must be early manifested in order to prevent higher damages. For network nodes to live long enough to fulfill their purpose, supply batteries consumption must be minimized. Hence, needless communication between the nodes must be lowered using different data aggregation and suppression techniques. New data suppression technique using sliding window skylines is described in this paper.

The next section discusses applicability of WSNs for forest fire detection, their advantages comparing to other solutions (e.g. satellite imagery) and issues that must be dealt with in order to improve reliability of the solutions and make readings of fire detection more accurate. The third section describes implementation of algorithm that uses sliding window skylines and simulation based on the case study of early forest fire detection. The fourth section presents the algorithm implementation and its simulation in WSNs usage for forest fire detection. The fifth section evaluates related work and the final section gives conclusion and future work.

## 2 Wireless Sensor Networks for Forest Fire Detection

As stated before, various applications deploy large number of small and inexpensive sensor nodes with one or more sensors attached and a fewer number of intermediate nodes, which aggregate and/or suppress sensing data forwarding them to the sink. Nowadays, WSNs are widely applied in environmental applications, such as habitat monitoring, agriculture research, earthquake monitoring, traffic control and fire detection, because of the nature of such applications that involve people, assets and environment in events of disasters, accidents and other needs of a today's society.

Wild fires, including forest and plant fires, are uncontrolled fires occurring in wild and rural areas which can cause significant damage to natural and human resources [4]. Common causes of forest fires are lightning strikes, human carelessness, and exposure of fuel to extreme heat and aridity.

In the last years heat waves in Europe caused magnified number of forest fires with devastating outcomes. Croatia also belongs to countries with increased risk of summer forest fires, especially on Dalmatian coast and islands in Adriatic Sea. For example, in period from January 1st to December 31st 2007 there were 8945 fires registered in Croatia, and 5455 of them (61%) were plant and forest fires that burnt the area of 67992 hectares [5]. Comparing these data with the ones for 2006, it can be seen that number of fires has increased for 25.7%, number of plant fires for even 52.6%, and the burnt area has increased 3.6 times. Burnt area index, given in hectares per fire, has increased even 2.4 times, which shows the enlarged threat of every fire occurred for landscape, human and animal life in affected areas. Therefore, early detection and suppression of forest fires is crucial for restriction of their propagation.

Most of existing forest fire detection systems rely on the satellite imagery. These approaches are limited due to the weather conditions (e.g. clouds) that can seriously decrease the detection accuracy. These issues become irrelevant when implementing forest fire detection using WSNs.

One of the most important systems aspects of a WSN for forest fire detection is its lifetime [6], because the fire detection network must operate for a very long period of time in order to detect such a comparatively rare event. Occurring of forest fires must not be confused with sensor battery exhaustion, as well as antennas being reoriented in the wrong direction by falling branches, curious animals, wind, etc. Another critical issue for such application is field coverage, since the WSN must identify the event quickly and accurately. In order to do so each node must be given a unique identification number (ID) which is associated with a node's precise position in the WSN or globally, with the use of GPS. With knowledge of the node ID and specific coverage area it is possible to precisely locate the source of the fire.

Fires can differ in size, shape, growth, frequency, and intensity. Nevertheless, all these parameters become irrelevant if it is possible to notice the fire at its very beginning. However, for early fire detection, coverage area of single node should be as low as possible; if the coverage area is too wide the sensor node will report fire once the fire reaches the sensor, which could be too late. Small size coverage areas would require more sensors which could be too expensive, so it is necessary to reach the optimum size of coverage area, which is out of the scope of this paper.

The sensors clustering using a distributed protocol, their physical distribution along the targeted area and data routing problems are outside the scope of this paper.

## 3 Sliding Window Skylines

Suppose that each sensor in our network measures a certain number of parameters to which we will refer as attributes. These attributes form multi-dimensional attribute space. Therefore, we assume that every sensor reading is a point in the attribute space.

For any two values of an attribute (e.g. temperature), it is easy to say which of the values is better indicator of forest fire (one with the higher temperature). In the case of sensor readings, which are multi-dimensional, it is not so easy to say which reading is better indicator of forest fire. For example, let us take a look at Figure 1, where some readings in two-dimensional attribute space are shown. We have two attributes: temperature and wind speed. We want to know which readings are better indicators of forest fire than the others. This is typical example of ranking objects by more than one criterion. In our example, a reading is better indicator than another (i.e. we will say that it dominates) if it has larger temperature and higher wind speed. The top readings are the set of readings which are not dominated by any other reading. They are shown as gray dots in the figure. We see that they form line which bounds all other readings. This line is usually called the skyline, while the top readings are called skylines.

Each sensor in our network periodically measures attributes. For early detection of forest fire we do not have to collect all these readings at sink. Actually, at some point in time, it is sufficient to know skylines, because all other readings are bounded with them. To save power, we propagate only the skylines to the sink. At each inner sensor node on path to the sink, we aggregate both its own and received skylines, and send these aggregated skylines to next node on the path. Thereby, at the sink, we will receive skylines that will be aggregates of the whole WSN, such that all readings of all sensors will be bounded with them. This is an effective way of sensor readings aggregation, which reduces power consumption of WSN, because local processing and storage at node consumes less energy than transmitting data over the radio [19].

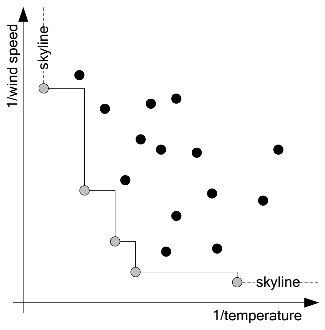

**Figure 1 - Skyline of Sensor Readings**

At the sink, we want to continuously monitor the environment, and therefore we have to know only most recent skylines, because the older ones do not reflect a present state of the monitored environment. For all readings (and skylines) we define lifetime in the form of time window. Every reading older than a size of the window will be deleted from the network. If the size is very little, every reading will be skyline in the window, and this will result in too many transmitted readings. On contrary, if the size is too large, aggregated skylines will not adequately reflect the present state. In practice, this size highly depends on dynamism of the phenomenon of interest (i.e. forest fire in our case).

## 4 Algorithm

At each sensor node we maintain an index of unexpired skylines and readings dominated by them. We propagate every change in the skylines to next sensor node on the path to the sink. In the rest of this section we present our algorithm for continuous maintaining of sliding window skylines. This algorithm is run at every sensor node, once for every of its own and received readings. The synchronization between nodes is not needed, because every node on the path has its own clock and for each propagated skyline it adds time spent on it. In this way, any node on the path may delete expired readings from its index.

```
method add new reading
  check oldest skyline
  foreach skyline
    if (skyline dominates reading)
      add reading to skyline
      return
    else if (reading dominates skyline)
      remove skyline
  add reading to skylines and propagate it.
```

The upper method adds new reading to the index. First we check if oldest skyline is expired. After that we go through present skylines and check if the reading is dominated by any of them. If this is the case, we add it to the set of readings dominated by the first such skyline we encountered. Otherwise, we remove from the index every skyline dominated by it, add it to the skylines and propagate it to next node on the path. We have to keep all younger readings that are dominated by any skyline (or any other reading) in the index, because they are potential skylines in future. Readings dominated by any younger reading (or skyline) cannot become skyline in future, and we may delete them from index. It is important to notice that we do not have to add a reading to all sets of all skylines that dominate it, because it may become skyline only if all such skylines are expired.

The following method adds a reading to the set of readings dominated by some other reading (i.e. parent). It is almost identical to the upper method, except we do not check if the parent is expired and we do not propagate reading after adding it to parent's children, because it is not a skyline. We do not have to check if the parent is expired, because it is not a skyline, and therefore it is younger than unexpired skyline which dominates it.

```
method add reading to parent
  foreach child of parent
    if (child dominates reading)
      add reading to child
      return
    else if (reading dominates child)
      remove child
  add reading to children of parent.
```

As we said before, each time we add new reading we have to check if the oldest skyline is expired. If it is expired we have to remove it and add all of its children from the index. The following two methods do this job. To improve performances of our algorithm we keep skylines (and children of other readings) in red-black trees sorted by their time of expiry.

```
method check oldest skyline
  if (oldest skyline is too old)
    remove oldest skyline from skylines
    foreach child of oldest skyline
      add child.

method add reading
  foreach skyline
    if (skyline dominates reading)
     if (skyline is younger than reading)
       foreach child of reading
         add child
       return
     else
       add reading to skyline
       return
  add reading to skylines and propagate it.
```

## 5 Experimental Evaluation

In this section we examine our algorithm and index structure on the performances of sliding window skyline computation. Following well-established methodology set by previous research on skyline algorithms [12, 13, 14] we choose to use the following three data distributions: independent, correlated and anti-correlated. In this section we want to see simulation runtimes for different cardinality (i.e. the number of readings) and different sizes of time windows. Shorter runtime means less processing at every sensor node in the network, and therefore smaller consuming of battery power which results in longer lifetime of the network.

The following setup was used in experiments: We created synthetic datasets and vary the cardinality from 10 thousands to 1 million. We fixed dimensionality to 2, and used 2 different sizes of the time window. The first size was 5000, and the second was 10 percent of the cardinality. Experiments were run in Java on a PC with 2.8 GHz Pentium Processor and 1GB of main memory.

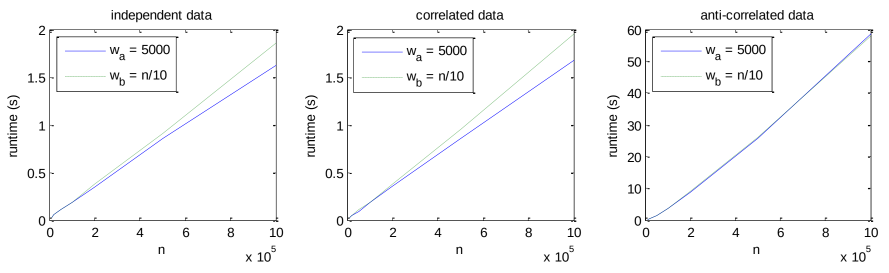

**Figure 2 - Results of Experiments**

In Figure 2 we can see the results of the experiments, where n is the cardinality. We conclude that there is no significant difference in runtime for the independent and correlated datasets. We see that for larger cardinalities there is slight increase in the runtime for window $w_b$, which has the size of 10 percent of the cardinality. This is expected, because skylines live longer in the case of larger window sizes, and therefore the number of readings in the index is larger. For the anti-correlated dataset, we see that the runtime is more than one order of magnitude longer. This is expected, because many readings are skylines in the anti-correlated dataset.

Morse at al. [14] did an extensive evaluation of different algorithms for continuous skyline computation. For the independent and correlated datasets the runtime of our algorithm is one order of magnitude smaller (0.1 s comparing to their 1 s for cardinality 5000), while for the anti-correlated dataset our algorithm is equally fast to their algorithm. They are using $R^*$-trees and Quadtrees for indexing of skylines (and other readings). We suppose that our approach is faster for continuous skyline computation over sliding windows because we store skylines (and other readings) directly to the index as its nodes, while Morse at al. are storing them as end-nodes (i.e. leaves) of used structures. On the other hand, the memory consumption should be lower in our approach, because of the same reason.

## 6 Related Work

We have recognized two different topics of related work for our subject: usage of wireless WSNs in fire detection systems, and skylines computation in WSNs.

Some papers exploit the essence of artificial neural networks to perform simple calculations at many organized single nodes in order to conduct complex data processing [7]. This approach also, they claim, reduces communication overhead and energy consumption. Some recent, also domestic, works combine video and network sensor monitoring [8] [9], but their approach doesn't analyze the importance of data aggregation and suppression from sensors. They rely on algorithms for image analysis and looking of visual signs of forest fires, particularly forest fire smoke during the day and forest fire flames during the night. Some solutions employ the usage of mobile agents [8] [10] for efficient network exploration and fire detection, but the communication paths seem overused while agents tracking fire, which exhausts the sensor supplies (e.g. batteries). Some enhanced supply solutions exploit solar energy.

Skyline computation in WSNs has received much attention recently [15, 16, 17]. Kwon at al. [15] propose an algorithm for in-network processing of skyline queries. Chen at al. [17] use advanced approach for continuous in-network skyline computation that employs hierarchical thresholds at the nodes. Xin at al. [16] propose an energy-efficient algorithm for continuous skyline computation. There are also many other paper related to continuous skyline computation, which are not related to WSNs [12, 14, 18]. We compared our algorithm with [14] in Section 5.

## 7 Conclusion

This paper proposes enhanced algorithm for continuous skyline computation in order to suppress collected data from wireless sensors that are monitoring environment for potential forest fires. In experimental evaluation, we showed that our approach is faster than other algorithms for continuous skyline computation over sliding windows.

Because of the increased risk of forest fires in Croatia, especially during summer seasons, early detection and suppression of forest fires is crucial, in order to restrict their propagation. According to the Law of fire protection in the Republic of Croatia, National Protection and Rescue Directorate and its Fire Fighting Sector are obligated to propose Activity Program of special measures in fire protection on a yearly basis. One of the main goals of the program is development and implementation of new fire protection systems. This paper therefore represents useful and applicable contribution in that direction.

## Acknowledgements

This work was carried out within the research project "Content Delivery and Mobility of Users and Services in New Generation Networks", supported by the Ministry of Science, Education and Sports of the Republic of Croatia.

## References

1. Chong, C., Kumar, S.: Sensor Networks: Evolution, Opportunities and Challenges. In: Proceedings of the IEEE, Vol.91, No.8, pp. 1247--1256. IEEE Press, New York (2003)
2. Callaway, E.H.: Wireless Sensor Networks: Architectures and Protocols. Auerbach Publications, CRC Press LLC, Boca Raton (2004)
3. Akyildiz, I. A., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A Survey of Sensor Networks. IEEE Communications Magazine 40, 8, pp. 102--114. IEEE Press (2002)
4. Hedeeda, M.: Forest Fire Modeling and Early Detection using Wireless Sensor Networks. Technical report CMPT2007, Faculty of Applied Sciences, Simon Fraser University, Canada (2007)
5. Firefighter Gazette, No 3, Croatian Firefighter Association, Zagreb (2008) (In Croatian)
6. Tanenbaum, A.S., Gamage, C., Crispo, B.: Taking Sensor Networks from the Lab to the Jungle. IEEE Computer, Vol.39, Issue 8, pp. 98--100. IEEE Computer Society (2006)
7. Yu, L., Wang, N., Meng, X.: Real-Time Forest Fire Detection with Wireless Sensor Networks. In: Proceedings of International Conference on Wireless Communications, Networking and Mobile Computing, Vol.2, pp.1214--1217. IEEE Press, New York (2005)
8. Stipanicev, D., Bodrozic, Lj., Stula, M.: Environmental Intelligence Based on Advanced Sensor Networks. In: Proceedings of 14th International Workshop on 2007 IWSSIP&EC-SIPMCS. Maribor, Slovenia (2007)
9. iForestFire – intelligent Forest Fire monitoring system, Faculty of Electrical Engineering, Machine Engineering and Naval Architecture, University of Split, http://iforestfire.fesb.hr
10. Fok, C.-L., Roman, G.-C., Lu, C.: Efficient Network Exploration and Fire Detection using Mobile Agents in a Wireless Sensor Network. ONR-MURI Review, Baltimore, MD (2004)
11. Bekara, C., Laurent-Maknavicius, M., Bekara, K.: SAPC: A Secure Aggregation Protocol for Cluster-Based Wireless. In: Springer's IFIP MSN 2007 proceedings, Lecture Notes LNCS 4864. pp.784--798. The 3rd International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2007). Beijing, China (2007)
12. Lin X., Yuan Y., Wang W., Lu, H.: Stabbing the Sky: Efficient Skyline Computation over Sliding Windows. In: Proceedings of the 21st International Conference on Data Engineering (ICDE'05), pp.502--513. IEEE Computer Society (2005)
13. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. In: Proceedings of the ACM SIGMOD International Conference on Management of Data. San Diego, California (2003)
14. Morse, M., Patel, J. M., Grosky, W. I.: Efficient Continuous Skyline Computation. In: Proceedings of the 22nd International Conference on Data Engineering. IEEE Computer Society, Washington DC (2006)
15. Kwon, Y., Choi, J.-H., Chung, J.-D., Lee, S.K.: In-Network Processing for Skyline Queries in Sensor Networks. In: IEICE Transactions on Communications, E90-B(12): pp. 3452--3459. Oxford University Press (2007)
16. Xin, J., Wang, G., Chen, L., Zhang, X., Wang, Z.: Continuously Maintaining Sliding Window Skylines in a Sensor Network. In: Proceedings of the DASFAA 2007, pp. 509--521. Bangkok, Thailand (2007)
17. Chen, H., Zhou, S., Guan, J.: Towards Energy-Efficient Skyline Monitoring in Wireless Sensor Networks. In: Proceedings of the EWSN 2007, pp. 101--116. SpringerLink (2007)
18. Tao, Y., Papadias, D.: Maintaining Sliding Window Skylines on Data Streams. IEEE Transactions in Knowledge Data Engineering, pp. 377--391. IEEE Computer Society (2006)
19. D. Zeinalipour-Yazti, D., Kalogeraki, V., Gunopulos, D., Mitra, A., Banerjee, A., Najjar, W.: Towards In-Situ Data Storage in Sensor Databases. In: Bozanis, P., Houstis, E.N. (eds.) Advances in Informatics. LCNS, vol. 3746, pp. 36–46. Springer, Berlin (2005)