Population Estimation for Resource Inventory Applications over Sensor Networks

Jiun-Long Huang

Department of Computer Science,
National Chiao Tung University, Hsinchu, Taiwan
jlhuang@cs.nctu.edu.tw

Abstract. The growing advance in wireless communications and electronics makes the development of low-cost and low-power sensors possible. These sensors are usually small in size and are able to communicate with other sensors in short distances wirelessly. A sensor network consists of a number of sensors which cooperates with one another to accomplish some tasks. In this paper, we address the problem of resource inventory applications, which means a class of applications involving population calculation of a specific species or object type. To reduce energy consumption, each sensor only reports the number of sensed objects to the server, and the server will estimate the object number according to the received reports of all sensors. To address this problem, we design in this paper a population estimation algorithm, called algorithm Estimation, to estimate the object numbers. Several experiments are conducted to measure the performance of algorithm Estimation. The experimental results show that algorithm Estimation is able to obtain closer approximations of object numbers than prior algorithms.

Keywords: Sensor network, resource inventory, population estimation.

1 Introduction

The growing advance in wireless communications and electronics makes the development of low-cost and low-power sensors possible. These sensors are usually small in size and are able to communicate with other sensors in short distances wirelessly. A sensor network [1] consists of a number of sensors which cooperates with one another to accomplish some tasks. Sensors can be deployed either in a random or in a predetermined manner. Since being self-organized, sensors are able to form a sensor network automatically. Due to the characteristics of wireless communication and configuration-free deployment, sensor networks are suitable for various application areas including inventory management, product quality monitoring and disaster area monitoring [1][5]. Hence, sensor networks have attracted a significant research attention, including hardware and operating system design [8][13], localization [2][10], data aggregation methods [3][6][9] and applications of sensor networks [11][14].

X. Jia, J. Wu, and Y. He (Eds.): MSN 2005, LNCS 3794, pp. 175–184, 2005.

The authors in [7] described one kind of applications[1], resource inventory applications, which uses sensor networks to calculate object numbers. By resource inventory, it means a class of applications involving population calculation of a specific species or object type. To calculate population of objects, a number of sensors are deployed in a plane and each sensor is able to sense the number of objects within its sensing region. These sensors then form a sensor network and users are able to query the number of objects sensed by the sensor network via a server. For the sake of simplicity, in this paper we use "object number" to indicate the number of objects sensed by a sensor network. Since sensors are usually powered by batteries, energy conservation becomes an important issue in the design of sensor networks. To reduce energy consumption, the authors in [7] suggested that each sensor only reports the number of sensed objects to the server[2], and the server will estimate the object number according to the received reports of all sensors.

Since the sensors may be deployed randomly, the sensing regions of these sensors may be overlapped with one another. This phenomenon results in the difficulty of obtaining the exact object number due to the reason that one object may be sensed by more than one sensor. Fortunately, knowing the ranges of object numbers is still useful enough in many applications [7]. As a consequence, the authors in [7] proposed an energy-conserved scheme which is able to obtain the lower bounds and the upper bounds of the exact object numbers. For convenience, we name the scheme proposed in [7] as scheme SDARIA (i.e., the acronym of the title of [7]). Although being shown to be able to conserve much energy than other schemes [7], the lower bounds and the upper bounds obtained by scheme SDARIA are not informative. In our experiments, the upper bounds obtained by scheme SDARIA are around $170\% \sim 250\%$ of the exact numbers, while the lower bounds are around $60\% \sim 80\%$ of the exact numbers. Such high error rates make users not able to get enough information about the exact object numbers.

In view of this, we design in this paper a population estimation scheme, called algorithm Estimation, to estimate the object numbers. Specifically, algorithm Estimation first partitions the plane into several disjoint grids, and identifies the full and partial grids of each sensor. Algorithm Estimation then estimates the object number of each grid, and finally estimates the overall object numbers according to the estimated object number of each grid. Several experiments are conducted to measure the performance of algorithm Estimation. The experimental results show that algorithm Estimation is able to obtain closer approximations of object numbers than scheme SDARIA.

The rest of this paper is organized as follows. Section 2 gives a description of resource inventory applications and an overview of scheme SDARIA. The design of algorithm Estimation is given in Section 3. Section 4 shows the performance study of algorithm Estimation. Finally, Section 5 concludes this paper.

[1] Interested readers can refer to [7] for more examples of resource inventory applications.

[2] The detailed architecture of resource inventory applications is given in Section 2.

2 Preliminaries

2.1 Related Work

The system architecture of resource inventory applications proposed in [7] is shown in Figure 1. To calculate the number of objects, a number of sensors are deployed in a plane and each sensor is able to sense the number of objects within its sensing region. These sensors then form a sensor network and users are able to query the number of objects via a server. For energy conservation, each sensor only reports the number of sensed objects to the server when the server requests all sensors to report their sensing status. Similar as [7], we assume that the server is able to get the sensing regions of all sensors. The sensing region of each sensor can be obtained from manual measurement by human or automatic measurement by sensors when they are equipped with GPS [4].

Basically, scheme SDARIA comprises the following two phases.

- *Data aggregation phase:* In data aggregation phase, each sensor reports the number of sensed objects to the server via the sink. After receiving the reports of all sensors, the server transforms the received reports into the corresponding snapshot graph $G = \{V, E\}$. The transformation procedure is as follows. Each sensor is modelled as a vertex and there is an edge between two sensors if these two sensors' sensing regions are overlapped with each other. Figure 2 shows an example of the transformation between the reports and the corresponding snapshot graph.
- *Population estimation phase:* In population estimation phase, the server estimates the object number according to the snapshot graph G. In scheme SDARIA, two algorithms are proposed to obtain the upper bound and the lower bound of the exact object number.

Experimental results in [7] showed that the data aggregation method used in scheme SDARIA is able to greatly reduce the power consumption of sensors. Therefore, in this paper we adopt the data aggregation method used in scheme SDARIA and focus on population estimation phase. In scheme SDARIA, upper

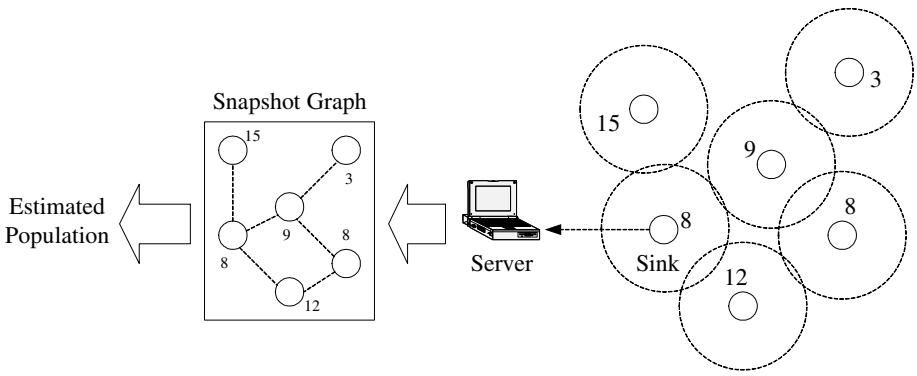


Fig. 1. System Architecture

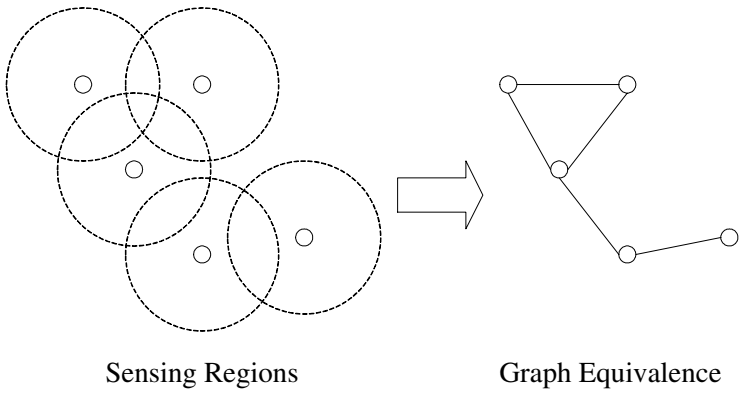


Fig. 2. Transformation between sensing reports and the corresponding snapshot graph

bound calculation is modelled as finding a subset of V which includes solely the vertices whose sensing regions cannot be replaced by the combinations of other vertices' sensing regions. For the interest of space, we do not describe the details of the upper bound calculation algorithm in this section. Interested readers can refer to [7] for details.

3 Design of Population Estimation Algorithm

Although we can obtain the lower bounds and the upper bounds of object numbers, respectively, by the lower bound and the upper bound calculation algorithms used in scheme SDARIA, these bounds do not give users enough information about the exact object numbers. For example, in our experiments, the obtained upper bounds are around $190\% \sim 210\%$ of the exact object numbers, and the lower bounds are around $60\% \sim 80\%$ of the exact object numbers. To address this problem, we propose a grid-based population estimation algorithm, called algorithm Estimation, to obtain close approximations of exact object numbers.

Before designing algorithm Estimation, we first partition the plane into several non-overlapped grids. Consider the sensing region of a sensor shown in Figure 3. A grid g is called the *full* grid of a sensor s if all the area of grid g is covered by the sensing region of sensor s. Similarly, a grid g is called the *partial* grid of a sensor s if only part of the area of grid g is covered by the sensing region of sensor s. A grid g is called the *overlapped* grid of sensor s if grid g is a full or a partial grid of sensor s. In Figure 3, the full and partial grids of the sensor are marked as 'F' and 'P', respectively.

To facilitate the estimation of the exact object numbers, we have the following assumptions.

1. The objects sensed by sensor s is uniformly distributed in the sensing region of sensor s.
2. The objects in a grid g is uniformly distributed in the area of grid g.

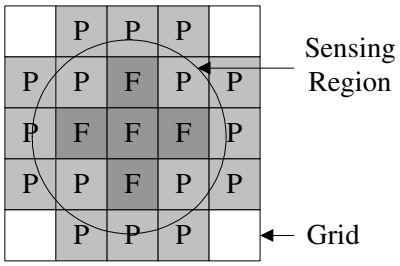


Fig. 3. Full and partial grids

3. On average, for a partial grid g of a sensor s, only half of the area of grid g is covered by the sensing region of sensor s.

Note that these assumptions are made only to guide the design of algorithm Estimation, and are not the limitations of algorithm Estimation. They will be relaxed in the experiments in Section 4.

The procedure of algorithm Estimation is as follows. Initially, each sensor is marked as UNSELECTED. Algorithm Grid-Estimation is an iterative algorithm and selects one sensor marked as UNSELECTED in each iteration. Consider the case that sensor s is selected. Denote the number of objects sensed by sensor s as $s.ObjNo$, and let $full$ and $partial$ be the sets of the full grids and the partial grids, respectively, of sensor s. By Assumption 3, we first calculate the *equivalent* number of full grids by considering one partial grid as $\frac{1}{2}$ full grid. Hence, the equivalent number of the full grids of sensor s is $|full| + \frac{|partial|}{2}$. By Assumption 1, $\frac{s.ObjNo}{|full|+\frac{|partial|}{2}}$ objects are expected in each full grid of sensor s. In addition, by Assumption 2 and Assumption 3, $\frac{s.ObjNo}{|full|+\frac{|partial|}{2}}$ objects are expected in each partial grid of sensor s. Hence, sensor s suggests that $\frac{s.ObjNo}{|full|+\frac{|partial|}{2}}$ objects are expected in each of its overlapped grids. Finally, sensor s is marked as SELECTED. The above procedure repeats until all sensors are marked as SELECTED. Since the expected object number in grid g may be suggested by several sensors, the expected object number in grid g is determined as the average of possible object numbers suggested by sensors. Finally, the estimated object number is determined as the summation of the expected object numbers of all grids. The algorithmic form of the proposed population algorithm is as follows.

Algorithm Estimation

1: **for** each grid g **do** /* Initialization */
2: $g.AvgObjNo \leftarrow 0$
3: $g.SensorNo \leftarrow 0$
4: **end for**
5: Mark all sensors as UNSELECTED
6: **while** (at least one sensor is marked as UNSELECTED) **do**

Pick one sensor which is marked as UNSELECTED
$full \leftarrow$ full grids of sensor s
$partial \leftarrow$ partial grids of sensor s
$expectation = \frac{ObjNo}{|full| + \frac{|partial|}{2}}$
for each grid g in $full \bigcup partial$ **do**
$g.AvgObjNo \leftarrow \frac{1}{g.SensorNo+1} \times (g.AvgObjNo * g.SensorNo + expectation)$
$g.SensorNo \leftarrow g.SensorNo + 1$
end for
Mark sensor s as SELECTED
end while
$total \leftarrow 0$
for each grid g **do**
$total \leftarrow total + g.AvgObjNo$
end for
return $total$

4 Performance Evaluation

4.1 Simulation Model

Similar as [7], the sensors are uniformly placed in to a 500×500m plane and the sensing radius of each sensor is set to 100m. We use GSTD tool [12] to generate the synthetic datasets used in this simulation. We synthesize the locations of objects by two distributions: uniform distribution and Gaussian distribution with standard deviation 50, and they are shown in Figure 4a and Figure 4b, respectively. Since focusing on population estimation phase, in data aggregation phase, we adopt the data aggregation method used in scheme SDARIA. In addition to algorithm Estimation, we also implement scheme SDARIA for comparison purposes. We implement both schemes in C++ and the simulation is executed in a PC with one Pentium III 500MHz CPU and 512MB memory. The default system parameters are listed in Table 1.

Since both schemes use the same data aggregation method, the energy consumption of the sensors in both schemes is the same. Hence, we take accuracy and execution time as the performance metrics of both schemes. Error rate, which is define as below, is taken to measure the accuracy of population estimation.

$$\text{Error rate} = \frac{\text{Estimated object number-Exact object number}}{\text{Exact object number}}$$


(a) Uniform


(b) Gaussian-50

Fig. 4. Datasets

Table 1. Default system parameters

Parameter	Value
Number of sensors (V)	30
Number of objects	2000
Sensing radius	100m
Grid size (Grid size ratio=2%)	2m

Table 2. Description of curves

Curve	Result
Lower bound	Lower bound obtained by scheme SDARIA
Estimation	Estimated object number
Upper bound	Upper bound obtained by scheme SDARIA

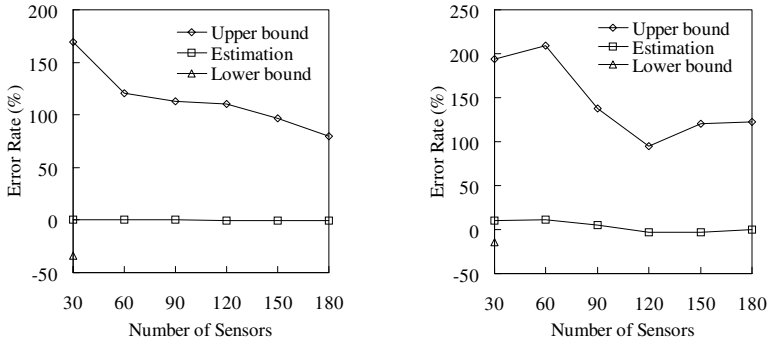
Note that error rate smaller than zero indicates that the estimated object number is smaller than the actual object number. The accuracy is higher when error rates are closer to zero. That is, the accuracy is high when the absolute values of error rates are small.

We now describe the meaning of the curves which will appear in the following experimental results. Lower bound represents the lower bounds obtained by scheme SDARIA. The lower bound calculation algorithms used in scheme SDARIA is a brute force-based algorithm to obtain the optimal solutions of maximal independent set problem [7]. Cureve Estimation represents the results of algorithm Estimation, while curve Upper bound represents the upper bounds obtained by scheme SDARIA. A summary of these curves as given in Table 2.

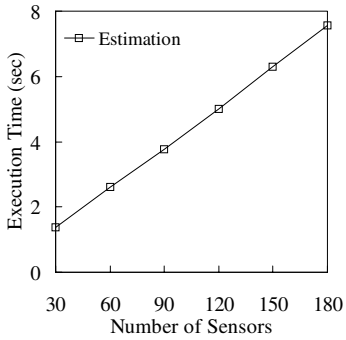
4.2 Effect of Sensor Number

This experiment is conducted to measure the effect of the number of sensors. Figure 5 shows the error rates and execution time of all algorithms with the number of sensors varied. The number of sensors is set from 30 to 180.

As observed in Figure 5a and Figure 5b, the error rates of upper bound calculation algorithm range from 80% to 205%. Note that the lower bound calculation algorithm used in scheme SDARIA is of high complexity, and hence, only the results with 30 sensors are shown in Figure 5a and Figure 5b. In our experiment, the execution time of the lower bound calculation algorithm is longer than six hours when the number of sensors is larger than 30. In addition, the error rates of algorithm Estimation are between -2.9% and 11.25% in this experiment. Since the absolute values of the error rates of algorithm Estimation are smaller than those of the lower bound and the upper bound calculation algorithms used in scheme SDARIA, algorithm Estimation is able to give users closer estimations than the lower bound and the upper bound calculation algorithms. Figure 5c



(a) Error Rate, Uniform (b) Error Rate, Gaussian-50



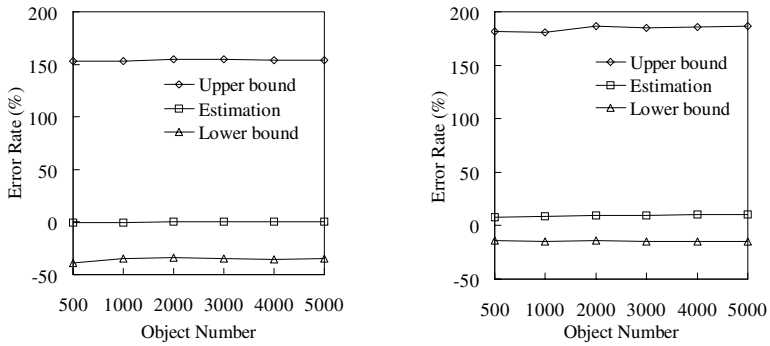
(c) Execution Time

Fig. 5. Effect of sensor number

shows the execution time of algorithm Estimation. The execution time of algorithm Estimation increases linearly as the number of sensors increases. This result agrees to the complexity analysis of algorithm Estimation in Section 3.

4.3 Effect of Object Number

This experiment is to investigate the effect of the number of objects. Figure 6 shows the error rates of all algorithms by setting object number from 500 to 5000. Since the execution time of all algorithms are not affected by object number and the distribution of objects, we only show the error rates of all algorithms in this subsection. As shown in Figure 6, the error rates of all algorithms are affected by the distribution of objects. Since being designed under the premise that objects are distributed uniformly, algorithm Estimation performs very well in dataset Uniform. It is also observed that the error rates of algorithm Estimation slightly increase as the number of objects increases. In this experiment, the error rates of algorithm Estimation increase from -0.81% to 0.1% and from 8% to 10.6%, respectively, in datasets Uniform and Gaussian-50. As shown in Figure 6, the error rates of the lower bound calculation algorithm in datasets Uniform and Gaussian-50 range from -33.94%



(a) Error Rate, Uniform (b) Error Rate, Gaussian-50

Fig. 6. Effect of object number

to -38.91% and from -14.2% to -15.1%, respectively. We observe that the absolute values error rates of the lower bound calculation algorithm become smaller when the distribution of objects becomes centralized.

5 Conclusion

In this paper, we addressed the problem of resource inventory applications over wireless sensor networks. To reduce energy consumption, each sensor reports only the number of sensed objects to the server, and the server will estimate the object number according to the received reports of all sensors. In view of this, we designed algorithm Estimation to estimate the object numbers. Several experiments were conducted to measure the performance of algorithm Estimation. The experimental results showed that algorithm Estimation was able to obtain close approximations of object numbers in reasonable execution time. In our experiments, the approximations of algorithm Estimation were around $95\% \sim 115\%$ of the exact object numbers. This result showed that algorithm Estimation is more suitable for practical use than prior schemes.

References

1. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A Survey on Sensor Networks. *IEEE Communications Magazine*, August 2002.
2. K. Chintalapudi, R. Govindan, G. Sukhatme, and Amit Dhariwal. Ad-Hoc Localization Using Ranging and Sectoring. In *Proceedings of the IEEE INFOCOM Conference*, March 2004.
3. J. Considine, F. Li, Kollios G, and J. Byers. Approximate Aggregation Techniques for Sensor Databases. In *Proceedings of the 20th IEEE International Conference on Data Engineering*, March-April 2004.
4. D. M. Doolin, S. D. Glaser, and N Sitar. Software Architecture for GPS-enabled Wildfire Sensorboard. In *TinyOS Technology Exchange*, February 2004.

5. S. D. Glaser. Some Real-World Applications of Wireless Sensor Nodes. In *Proceedings of SPIE Symposium on Smart Structures and Materials*, March 2004.
6. C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann. Impact of Network Density on Data Aggregation in Wireless Sensor Networks. In *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems*, July 2002.
7. T.-H. Lin and P. Huang. Sensor Data Aggregation for Resource Inventory Applications. In *Proceedings of the IEEE Wireless Communications and Networking Conference*, March 2005.
8. I. Locher, S. Park, M. Srivastava, A. Chen, R. Muntz, and S. Yuen. A Support Infrastructure for the Smart Kindergarten. *IEEE Pervasive Computing Magazine*, April-June 2002.
9. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: A Tiny Aggregation Service for ad hoc Sensor Networks. In *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, December 2002.
10. Y. Shang and W. Ruml. Improved MDS-Based Localization. In *Proceedings of the IEEE INFOCOM Conference*, March 2004.
11. G. Simon, M. Maroti, A. Ledeczi, G. Balogh, B. Kusy, A. Nadas, G. Pap, J. Sallai, and K. Frampton. Sensor Network-based Countersniper System. In *Proceedings of the 2nd ACM International Conference on Embedded Networked Sensor Systems*, November 2004.
12. Y. Theodoridis, J. R. O. Silva, and M. A. Nascimento. On the Generation of Spatialtemporal Datasets. In *Proceedings of the 6th International Symposium on Large Spatial Databases*, July 1999.
13. TinyOS Community Forum. http://www.tinyos.net/.
14. N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin. A Wireless Sensor Network for Structural Monitoring. In *Proceedings of the 2nd ACM International Conference on Embedded Networked Sensor Systems*, November 2004.