IPP Hurray!

Technical Report

Localizing Objects in Large-Scale Cyber-Physical Systems

Björn Andersson and Shashi Prabh

Localizing Objects in Large-Scale Cyber-Physical Systems

Björn Andersson and Shashi Prabh

1

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail: bja@isep.ipp.pt, ksph@isep .ipp.pt

http://www.hurray.isep.ipp.pt

Abstract

We use the term Cyber-Physical Systems to refer to large-scale distributed sensor systems. Locating the geographic coordinates of objects of interest is an important problemin such systems. We present a new distributed approach to localize objects and events of interest in time complexity independent of number of nodes.

Localizing Objects in Large-Scale Cyber-Physical Systems[*]

Björn Andersson   Shashi Prabh

IPP-HURRAY Research Group, CISTER/ISEP, Polytechnic Institute of Porto, Portugal
E-mail: {bja, ksph}@ .isep .ipp .pt

Abstract

We use the term Cyber-Physical Systems to refer to large-scale distributed sensor systems. Locating the geographic coordinates of objects of interest is an important problem in such systems. We present a new distributed approach to localize objects and events of interest in time complexity independent of number of nodes.

1. Introduction

One of the most frequent problems faced by distributed sensor system designers is to find the location of an object. Consequently, this problem has been extensively studied previously, especially in signal processing and sensor networks research communities. In this paper, we address the problem of localizing objects in large-scale Cyber-Physical systems consisting of millions of nodes in time complexity that is independent of the number of nodes.

A distributed sensor system deployment with 1000 nodes has already been reported [4]. In order to scale such systems to millions of nodes, the cost of an individual node must be kept small. So far, single-chip sensor nodes have not been manufactured in such large quantities, and consequently, the cost of individual sensor nodes has remained high. There are indications however of a convergence of wireless sensor network and RFID technologies such that RFID tags are used more and more for sensing tasks [1, 10]. This is significant since RFID tags are used widely in industry and are produced in billions of units annually at very low unit-cost. Furthermore, they are increasingly becoming more "capable" with programmable microprocessor, memory, sensors and sometimes even energy-storage devices (such as a battery or a capacitor). This evolution will foster an increasing interest in large-scale sensor systems.

There exist a number of solutions for node as well as event localizations. A survey of localization literature can be found in [5]. The localization technique presented by Ledeczi et. al. has been demonstrated to be successful on wireless sensor networks [7]. The authors consider their solution better than the other existing localization techniques. This technique is based on matching of audio signatures of rifle shots on a centralized server. Unfortunately, such solutions have a time complexity of O(m) in a single broadcast domain, where m is the number of sensor nodes. In this paper, we present a distributed algorithm for finding the location of an object. All sensor readings can affect the outcome of the localization calculation and the time complexity is independent of the number of sensor nodes. Our solution takes advantage of a prioritized MAC protocol that is collision-free if the priorities are unique. The widely used CAN bus [6] is the base protocol upon which the prioritized MAC protocol for wireless networks, called WiDOM, is built [3].

The remainder of this paper is organized as follows. Section 2 presents details of the localization problem formulation. Section 3 presents a background on the communication system we assume. Section 4 presents our algorithm to solve the localization problem. Section 5 presents conclusions.

2. Problem Formulation

Given a set of objects, we want to determine the geographical location and time associated with them. We use the terms "objects" and "events" of interest interchangeably. Let m denote the number of nodes of the sensor network. Let $H(x,y,z,t)$ denote the hypothesis that an object was located at (x,y,z) at time t. Let $N(x,y,z,t)$ denote the number of nodes that accept the hypothesis $H(x,y,z,t)$. We do not make any assumption about the mechanism by which the hypotheses are accepted or rejected – our algorithm is neutral to such mechanisms. Algorithms used to accept or negate the hypothesis can be as simple as a binary "yes/ no" based on sensor reading threshold or it could be based on matching the pattern of a sequence of sensor readings to a set of known signature signals. The latter approach is taken for the sniper detection system [7], where sequences

---

of audio samples are gathered on a centralized node and the following optimization problem is solved: find x,y,z,t that maximize $N(x,y,z,t)$.

In this paper, we use a fully distributed algorithm to estimate $N(x,y,z,t)$ (see Section 4). This distributed algorithm exploits the communication system. We will design our solution under the following assumptions. We assume that nodes know their own location but they do not know the location of any other node. We assume that all nodes are in a single broadcast domain.

3. Communication

A key building block of our new solution is the innovative use of a prioritized medium access control (MAC) protocol. For this reason, it behooves us to understand it further. Section 3.1 discusses how a prioritized MAC protocol can be used to compute the minimum of sensor readings. This initial understanding helps us to also understand how to estimate COUNT (in Section 3.2) efficiently. The COUNT primitive will turn out to be crucial in our localization.

3.1. Prioritized MAC

Let us consider the simple application scenario as depicted in Figure 1a, where a node (node N_1) needs to know the minimum (MIN) temperature reading among its neighbors. A traditional approach would be that N_1 broadcasts a request to all its neighbors and then N_1 waits for the replies from all of its neighbors. Clearly, even with a perfect a time division multiple access, this approach, the execution time is of the order of the number of neighbor nodes, i.e., $O(m)$.

Consider now that instead of using their priorities to access the medium, nodes use the value of its sensor reading as priority. For the simplicity sake, we assume that the sensor readings are coded as n-bit unique integers. Starting with the most significant bit first, let each node send the sensor reading bit-by-bit. Let us consider that the channel implements a logical-AND of the transmitted bits and for each transmitted bit and nodes read the resulting AND value in the channel (something straightforward in the wired medium). Finally, suppose that if a node reads '0' and is transmitting a '1', it stops transmitting (Figure 1b). Then, at the end of the transmission of n bits, the "observed" value in the channel will correspond to the MIN. It is as if all m temperature readings were transmitted in parallel (Figure 1c). Observe that for this case of computing MIN, there is no need for message payload.
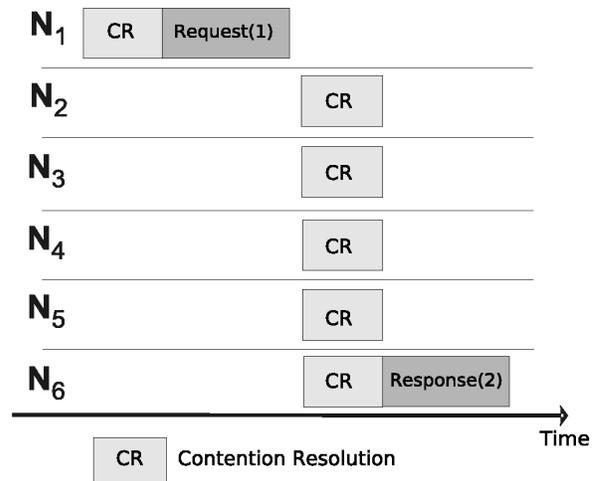
The medium access control (MAC) protocols that exhibit this logical AND behavior are known as Dominance (or, Binary-Countdown) protocols [9]. In the implementations of this protocol (e.g., the Controller Area Network or,
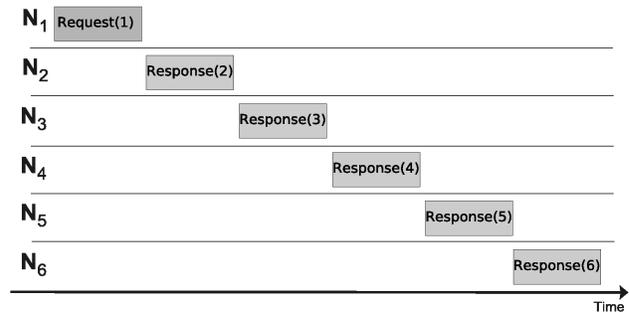
(a) Naïve algorithm

(b) CAN arbitration
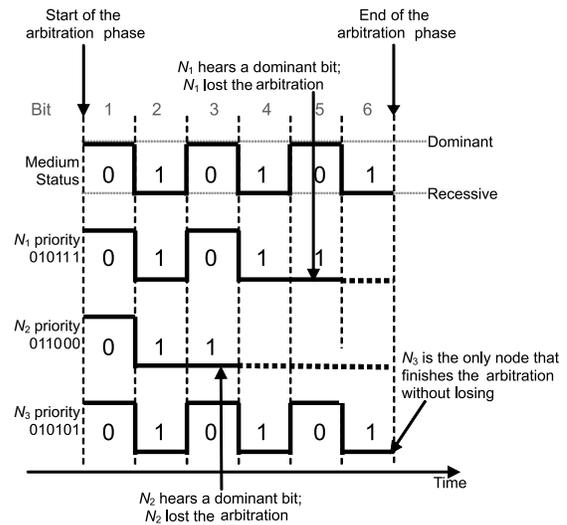
CR Contention Resolution

(c) WiDOM

Figure 1. Illustrations: (a) (#Nodes -1) responses as a result of the MIN request of N_1, (b) Bit-wise arbitration, (c) Only one response is generated for the same request.

CAN), messages have a unique contention field, which typically corresponds to a priority that is used to resolve the contention for channel access. After the completion of contention resolution phase, the node having message with the highest priority is granted channel access.

Current wireless transceivers can not transmit and receive data simultaneously. Therefore, for wireless networks, a node "transmitting a 1" performs carrier sensing only. All nodes "transmitting a 0" transmit simultaneously. Thus, those nodes that were "transmitting a 1" (i.e., doing carrier sensing) and sense a '0' being transmitted lose the tournament for channel access, and stop competing for channel access any further. Only those nodes that transmitted a '0' or did not lose in the round i, take part in the round $i + 1$ of the tournament.

In the following section, we apply this idea to get an estimate of number of nodes (having a certain property) in the network. We call this problem COUNT.

3.2. COUNT

The intuition behind our method of estimation of COUNT is as follows: If the contention field is a non-negative random number obtained at run-time, then the probability that the minimum value of the contention field is 0 approaches 1 as the number of nodes get very large. However, if there are only few nodes, then it is highly unlikely that the minimum among the random values is zero. From this observation, one can see that it is possible to estimate the number of nodes by computing the MIN of the random numbers.

The pseudo code of the algorithm for estimating the number of nodes is shown in Algorithms 1 and 2. The main algorithm (Algorithm 1) assumes that all computer nodes start their execution simultaneously and uses a global boolean variable $active$ as input, indicating if the node should be considered in the COUNT operation. When performing Algorithm 1, all nodes have $active$ equal to TRUE and proceed in following way. First, on line 8, the algorithm generates a random number in the range $[0, MAXV]$, then all nodes send their random number. When nodes call `send_empty`, the priority of the winner is returned and hence all nodes know the minimum random number (line 12). This is performed k times. The line 17 uses a function, shown in Algorithm 2 to compute the estimation of the number of nodes based on the minimum numbers obtained on line 12. The `if` statement on line 14 deals with the unlikely event that one of the minimum random numbers is $MAXV$. The design of the function in Algorithm 2 can be explained in terms of maximum-likelihood estimation. The reader can find more details on COUNT in [2].

Algorithm 1 Estimating COUNT (the number of nodes)

Require: All nodes start Algorithm 1 simultaneously.

1: active - a global boolean variable indicating if the node is considered in the COUNT
2: **function** *nnodes*(j : integer, x : array[1..k] of integer) **return** a real
3: r : array[1..k] of integer
4: x : array[1..k] of integer
5: q : integer
6: **for** $q \leftarrow 1$ **to** k
7:    **if** ($active$ = TRUE) **then**
8:       $r[q] \leftarrow random(0, MAXV)$
9:    **else**
10:       $r[q] \leftarrow MAXV$
11:    **end if**
12:    $x[q] \leftarrow$ `send_empty`($r[q]$)
13: **end for**
14: **if** ($\exists q : x[q] = MAXV$) **then**
15:    $est\_nodes \leftarrow 1$
16: **else**
17:    $est\_nodes \leftarrow ML\_estimation(x[1], x[2], ..., x[k])$
18: **end if**
19: **return** est_nodes *// the estimation of COUNT*

Algorithm 2 Function `ML_estimation`

Require: The division of two integers (as is done in line 6) returns a real number.

1: **function** *ML_estimation*(x : array[1..k] of integer) **return** an integer
2:   v : array[1..k] of real
3:   $sumv, q$: integer
4:   $sumv \leftarrow 0$
5:   **for** $q \leftarrow 1$ **to** k
6:     $v[q] \leftarrow ln\left(\frac{1}{1 - \frac{x[q]}{MAXV}} \right)$
7:     $sumv \leftarrow sumv + v[q]$
8:   **end for**
9:   **return** $\lceil \frac{k}{sumv} \rceil$
10: **end function**

4. Localization Algorithm

In Section 2, we presented the problem formulation where finding x,y,z,t corresponds to maximizing $N(x,y,z,t)$. Computing $N(x,y,z,t)$ at a specific point is however expensive in terms of communication. But we can use the COUNT primitive that we presented in Section 3.2. Let $N_{EST}(x,y,z,t)$ denote the *estimate* of $N(x,y,z,t)$. Our goal is now to find x,y,z,t such that $N_{EST}(x,y,z,t)$ is maximized. Since the function $N_{EST}(x,y,z,t)$ is "noisy," more than one maximizer for $N_{EST}(x,y,z,t)$ may exist. If $N_{EST}(x,y,z,t)$ is a sufficiently good estimate of $N(x,y,z,t)$ then these maximizers will be close if they represent the same event. In the case of multiple events, then multiple maximizers exist, and consequently, multiple hypotheses can be accepted. Observe that this is a problem of unconstrained optimization. It is known that pattern searching algorithms work well for such problems [8].

Let us reiterate our findings so far. We have found that a suitable way of designing a scalable localization algorithm is to (i) find x,y,z,t such $N_{EST}(x,y,z,t)$ is maximized, (ii) $N_{EST}(x,y,z,t)$ can be computed with the COUNT algorithm described in Section 3.2 and (iii) pattern searching is appropriate for solving the problem (i).

Algorithm 3 is based on these findings. The idea is that some node puts forward a hypothesis. It could be any node. However, nodes close to the events are better candidates since such a choice is likely to lead to a short search. All other nodes accept or reject the hypothesis and the node that puts forward the hypothesis counts the number of nodes that accepts the hypothesis. If the number of accepting nodes is not sufficiently large, other hypotheses are also considered and if one of them is accepted by more sensor nodes then the new hypothesis becomes the current hypothesis. The procedure is repeated until the hypothesis is accepted.

5. Conclusions

We have presented an algorithm for finding the location of an object. The algorithm has the virtue of having a time-complexity that is independent of the number of sensor nodes. Questions of finding the best way of doing the hypothesis verification and search, efficient localization of many objects and implementation of this algorithm on existing hardware are open.

References

[1] http://www2.lifl.fr/pops/senseid2007/.

[2] B. Andersson, N. Pereira, and E. Tovar. Estimating the number of nodes in wireless sensor networks. Technical Report TR-060702, IPP-HURRAY group, Institute Polytechnic Porto, available at http://www.hurray.isep.ipp.pt/privfiles/hurray-tr-060702.pdf, 2006.

[3] B. Andersson, N. Pereira, and E. Tovar. Widom: A dominance protocol for wireless medium access. *IEEE Transactions on Industrial Informatics*, 3(2):120–130, 2007.

[4] A. Arora et al. Exscal: Elements of an extreme scale wireless sensor network. In *Proc. of the 11th IEEE Intl. Conf. on Embedded and Real-Time Computing Systems and Applications (RTCSA'05)*, pages 102–108, Washington, DC, USA, 2005. IEEE Computer Society.

[5] J. Bachrach and C. Taylor. *Localization in sensor networks*. Wiley, 2005.

[6] Bosch GmbH, Stuttgart, Germany. *CAN Specification, ver. 2.0*, 1991.

[7] A. Ledeczi, A. Nadas, P. Volgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dora, K. Molnar, M. Maroti, and G. Simon. Countersniper system for urban warfare. *ACM Trans. Sen. Netw.*, 1:153–177, 2005.

[8] R. M. Lewis, V. Torczon, and M. W. Trosset. Why pattern search works. In *Technical Report*, Baltimore, Maryland, 1998.

[9] A. K. Mok and S. Ward. Distributed broadcast channel access. *Computer Networks*, 3:327–335, 1979.

[10] J. Want. Enabling ubiquitous sensing with RFID. *IEEE Computer Magazine*, 37:84–86, 2004.

Algorithm 3 Localization algorithm – pseudo code

Require: All nodes start Algorithm 3 simultaneously.

1: Some node, say node N_i broadcasts the hypothesis $H(x, y, z, t)$

2: Nodes decide to accept or reject the hypothesis

3: Node N_i counts the number of nodes that accepted the hypothesis using the technique described in Section 3.2

4: **if** a sufficiently large number of nodes accepted the hypothesis **then** go to 7.

5: **else**

6: **Do hypothesis search in the neighborhood of** $(\mathbf{x}, \mathbf{y}, \mathbf{z}, \mathbf{t})$**.** For example, consider:
   $H(x \pm \Delta_x, y \pm \Delta_y, z \pm \Delta_z, t \pm \Delta_t)$ where Δ's represent search neighborhood size.
   if none of the exploratory moves had a higher N_{EST} **then** increase the neighborhood size and search again **else**
   let x_e,y_e,z_e,t_e denote the coordinate among the exploratory moves with the highest N_{EST}.
   x := x_e; y := y_e; z := z_e; t := t_e
   increase the search neighborhood size
   go to step 4

7: The values x,y,z,t is the location sought.

8: Case: **Multiple events.** If there exists a node N_k that can form another hypothesis with sufficiently different variables from previously accepted hypotheses, then N_k broadcasts its hypothesis and the algorithm starts from step 1. The algorithm terminates when no sufficiently different new hypothesis can be accepted.