Designing decentralized spatial algorithms for geosensor networks

M. Duckham[1], D. Nussbaum[2], J-R. Sack[2], N.Santoro[2]

[1]Department of Geomatics, University of Melbourne, Victoria 3010, Australia
Telephone: +61 3 8344 63935
Fax: +61 3 9347 2916
Email: mduckham@unimelb.edu.au

[2]Department of Computer Science, Carleton University, Ottawa K1S 5B6, Canada

1. Introduction

Over recent decades, research in the field of geocomputation has developed an arsenal of algorithms for geographic information processing and analysis. With few exceptions, these spatial algorithms almost universally assume that all the relevant geographic data are readily available to an algorithm (for example stored in one or more centralized spatial information system, like GIS or spatial database, accessible to the algorithm either directly or via a digital communication network). By contrast, this paper examines the design of *decentralized* spatial algorithms, where multiple spatial information systems each with only partial knowledge of the relevant geographic data, must collaboratively process data to generate the required answer.

Research into decentralized spatial algorithms is motivated by the development of new technologies, in particular *geosensor networks* (GSN). A geosensor network is type of wireless sensor network (wireless network of miniaturized sensor-enabled computers, called nodes) that monitors geographical phenomena in the environment (Nittel et al. 2004, Zhao and Guibas, 2004). Each node in a geosensor network can sense information about its immediate geographic environment (including parameters like temperature, light, CO2, humidity, soil moisture, etc.); and communicate with other nodes in close spatial proximity. However, geosensor networks are uniquely resource-constrained information systems, particularly with respect to energy resources. As a result, GSN have limited energy resources for communication (the most energy-intensive operation for a node). Limited communication resources make it highly inefficient to communicate raw data from a GSN to a centralized silo for processing. Instead, efficient algorithms require geographic data be processed *in the network*, relying on collaborative computation between nearby nodes with no centralized control.

Unfortunately, designing decentralized algorithms is notoriously difficult, even more so in the case of GSN where the environment and node locations, may both vary spatially and temporally. The key challenge facing any decentralized spatial algorithm is how to reliably achieve the desired *global* spatial behavior through the specification of local *node* behaviors (cf. Estrin et al., 2000). In this context, this paper sets out a new approach to designing decentralized spatial algorithms suitable for operation in GSN. The approach draws upon established tools and techniques for (decentralized) algorithm design, but extends and adapts them for use in a spatiotemporal context.

2. Algorithm design process

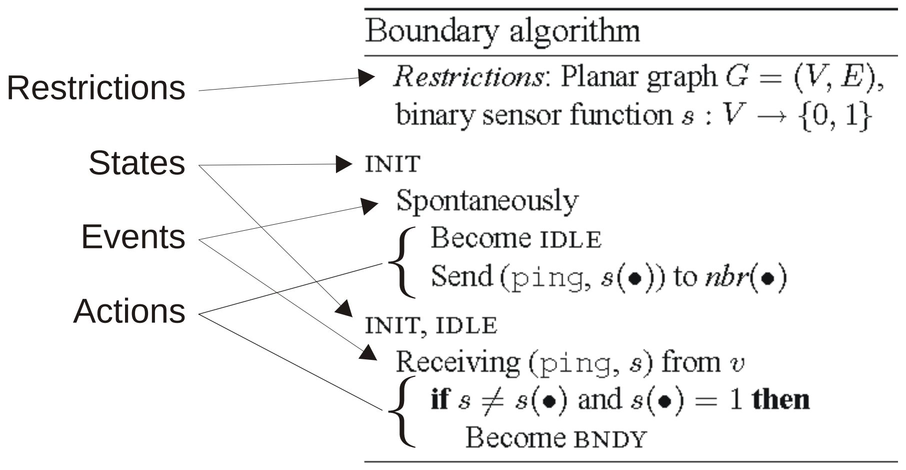
The decentralized algorithm design process proposed in this paper iterates over two distinct phases. First, the *specification phase* sets out in a structured way the computational procedure and interactions each individual system component will engage in. The specification stage is founded on a technique developed by Santoro (2007), but is extended with (quantitative and qualitative) representations of the spatial location of nodes; and representations of the spatiotemporal changes both in the geographic environment and the locations of nodes. Second, the *analysis phase* critically examines the specification, identifying problems and limitations, faults and weaknesses. Depending on the results of the analysis, the process then iterates until no further changes are needed, and the algorithm adequately solves the problem.

2.1 Specification

The decentralized spatial algorithm specification technique used is founded on four key structures

1. *restrictions* on the environment in which the algorithm operates, including:
    - non-spatial restrictions on what data can be sensed by nodes;
    - spatial restrictions on what information a node is assumed to be able to sense about their absolute and/or relative location;
    - temporal restrictions on whether nodes are assumed to be static or move over time, and whether sensed data changes over time;
    - uncertainty restrictions on how reliable communication, sensing, positioning, and computation are assumed to be; and
    - network restrictions on the structure of the communication network (e.g., whether the network is planar or non-planar, any other network connectivity or topology restrictions that may exist).
2. *events* that occur to nodes, such as the receipt of a messages from a neighbor, comprising:
    - receipt of a message, sent by a neighboring node;
    - a triggered event, such as a scheduled alarm or periodic sensor reading; or
    - a spontaneous impulse, external to the system.
3. *actions* that a node can perform in response to the different events that occur. Actions must be atomic sequences of operations that cannot be interrupted by other events.
4. *states* for a node, which allow nodes to retain knowledge of previous interactions.

An example of a simple decentralized algorithm for locally detecting the boundary of a region is shown in figure 1. In the example algorithm, each node communicates its current sensed value (in this simplified case, 1 or 0) to its immediate one-hop neighbors. Any node that is both inside the region (senses 1) and has a one-hop neighbor that is outside the region (senses 0) can locally decide it is at the boundary, without requiring any centralized control. While extremely simple, boundary detection is an important component of more sophisticated algorithms, such as area computation, and topological relationships between regions.



```
Boundary algorithm
Restrictions: Planar graph $G = (V, E)$,
binary sensor function $s : V \to \{0, 1\}$
INIT
    Spontaneously
        Become IDLE
        Send (ping, $s(\bullet)$) to $nbr(\bullet)$
INIT, IDLE
    Receiving (ping, $s$) from $v$
        if $s \neq s(\bullet)$ and $s(\bullet) = 1$ then
            Become BNDY
```

Figure 1. Example decentralized boundary algorithm.

2.2 Analysis

The analysis phase comprises three distinct components:

- An *adversarial analysis*, where the designer adopts the role of an adversary, whose objective is to find as many scenarios as possible where the algorithm fails in some way, including generating incorrect results, executing inefficiently, or failing to terminate.
- A *computational analysis*, which aims to determine the *communication complexity* both for the algorithm overall, and for individual nodes across network.
- An *execution analysis*, which aims to explore the emergent, dynamic properties of the algorithm, through the interactions between individual nodes, for example using *sequence diagrams*.

In the adversarial analysis, the designer probes the limitations of the algorithm, for example by introducing uncertainty into the sensed values, or examining the effects of unreliable communication between nodes. In the computational analysis, the examination focuses on the communication resources required by the algorithm. For example, the algorithm in figure 1 has an overall linear communication complexity $O(n)$ (in total n messages are sent, where n is the size of the network), but a constant time load balance $O(1)$ (each node sends exactly one message). Such analyses allow direct comparison of the computational resource requirements different algorithms. In the execution analysis, the designer investigates the dynamic behavior of the algorithm. For example, figure 2 shows a sequence diagram for a 1D arrangement of nodes (the limitations of graphic depiction on flat paper restrict sequence diagrams to 1D spatial arrangements), showing the events and state changes resulting from the boundary algorithm in figure 1.

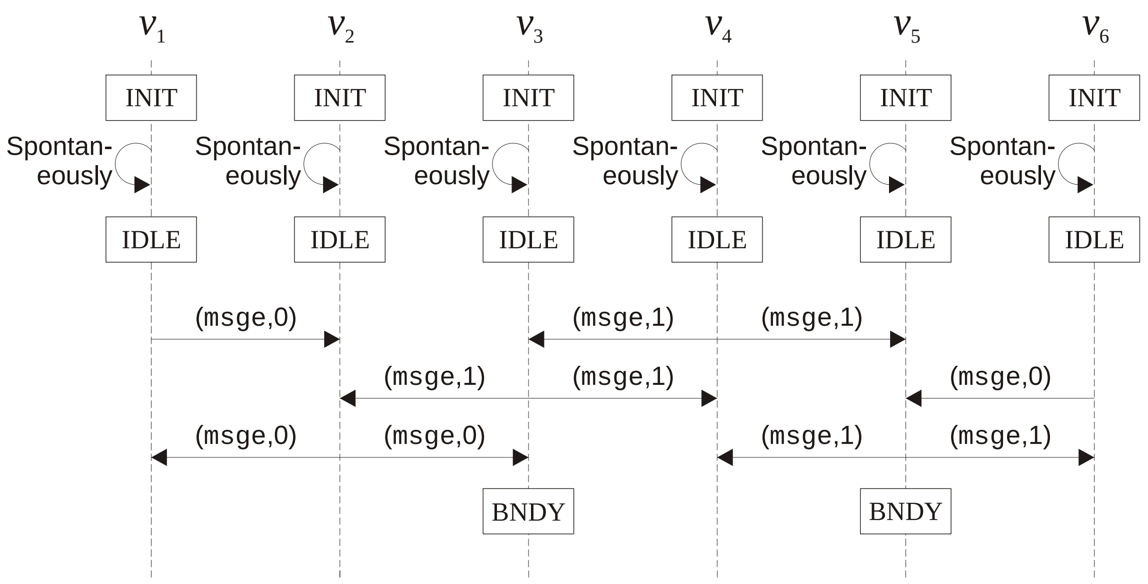
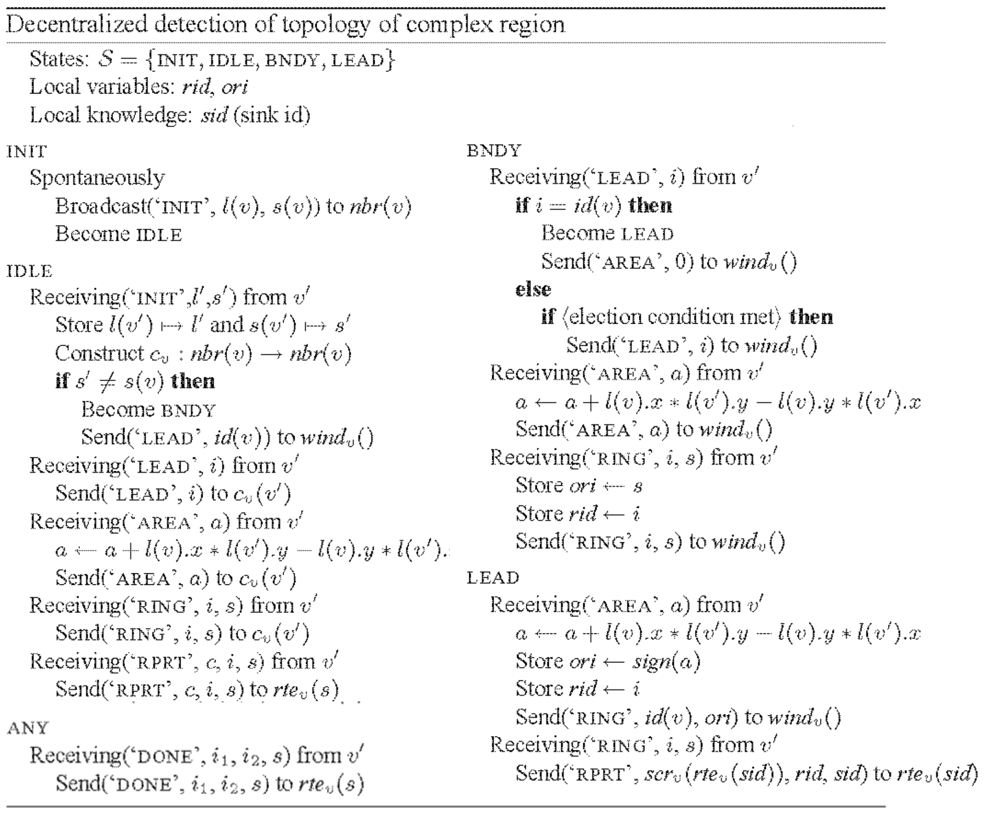


Figure 2. Example sequence diagram for seven nodes operating boundary algorithm in figure 1, where s={(v_1,0), (v_2,0), (v_3,0), (v_4,1), (v_5,1), (v_6,1), (v_7,0),} (i.e., where v_3, v_4, v_5 are inside region).

3. Example design

The full paper explores the efficacy of the decentralized algorithm design process through the development of a sophisticated decentralized algorithm. The algorithm, shown in figure 3, is capable of determining the topology of a complex areal object, potentially comprising multiple sub-regions, islands, and holes (after a model by Worboys and Bofakos, 1993). The example demonstrates the utility of the approach in designing highly complex decentralized algorithms. In the light of these experiences, the conclusions of the full paper look forward more broadly at the increased role for decentralized spatial computing in the near future as technologies like GSN become more mature.



```
Decentralized detection of topology of complex region
  States: $S = \{\text{INIT}, \text{IDLE}, \text{BNDY}, \text{LEAD}\}$
  Local variables: $rid$, $ori$
  Local knowledge: $sid$ (sink id)

INIT
  Spontaneously
    Broadcast('INIT', $l(v)$, $s(v)$) to $nbr(v)$
    Become IDLE

IDLE
  Receiving('INIT',$l'$,$s'$) from $v'$
    Store $l(v') \mapsto l'$ and $s(v') \mapsto s'$
    Construct $c_v : nbr(v) \to nbr(v)$
    if $s' \neq s(v)$ then
      Become BNDY
      Send('LEAD', $id(v)$) to $wind_v()$
  Receiving('LEAD', $i$) from $v'$
    Send('LEAD', $i$) to $c_v(v')$
  Receiving('AREA', $a$) from $v'$
    $a \leftarrow a + l(v).x * l(v').y - l(v).y * l(v').$
    Send('AREA', $a$) to $c_v(v')$
  Receiving('RING', $i$, $s$) from $v'$
    Send('RING', $i$, $s$) to $c_v(v')$
  Receiving('RPRT', $c$, $i$, $s$) from $v'$
    Send('RPRT', $c$, $i$, $s$) to $rte_v(s)$

ANY
  Receiving('DONE', $i_1$, $i_2$, $s$) from $v'$
    Send('DONE', $i_1$, $i_2$, $s$) to $rte_v(s)$

BNDY
  Receiving('LEAD', $i$) from $v'$
    if $i = id(v)$ then
      Become LEAD
      Send('AREA', 0) to $wind_v()$
    else
      if ⟨election condition met⟩ then
        Send('LEAD', $i$) to $wind_v()$
  Receiving('AREA', $a$) from $v'$
    $a \leftarrow a + l(v).x * l(v').y - l(v).y * l(v').x$
    Send('AREA', $a$) to $wind_v()$
  Receiving('RING', $i$, $s$) from $v'$
    Store $ori \leftarrow s$
    Store $rid \leftarrow i$
    Send('RING', $i$, $s$) to $wind_v()$

LEAD
  Receiving('AREA', $a$) from $v'$
    $a \leftarrow a + l(v).x * l(v').y - l(v).y * l(v').x$
    Store $ori \leftarrow sign(a)$
    Store $rid \leftarrow i$
    Send('RING', $id(v)$, $ori$) to $wind_v()$
  Receiving('RING', $i$, $s$) from $v'$
    Send('RPRT', $scr_v(rte_v(sid))$, $rid$, $sid$) to $rte_v(sid)$
```

Figure 3. Decentralized algorithm for computing the topology of a complex areal region, with islands and holes (detailed explanation in full paper).

4. References

Estrin, D., Govindan, R., and Heidemann, J. 2000. Embedding the Internet: Introduction. *Communications of the ACM*, 43(5): 38–41.

Nittel, S., Stefanidis, A., Cruz, I., Egenhofer, M., Goldin, D., Howard, A., Labrinidis, A., Madden, S., Voisard, A., and Worboys, M., 2004. Report from the First Workshop on Geo Sensor Networks, *ACM SIGMOD Record*, 33(1).

N. Santoro, 2007. *Design and Analysis of Distributed Algorithms*. New Jersey: Wiley.

Worboys, M. F. and Bofakos, P., 1993. A canonical model for a class of areal spatial objects. In *Proc. Third International Symposium on Advances in Spatial Databases* (SSD'93). Berlin: Springer, 36–52.

Zhao, Z. and Guibas, L. J., 2004. Wireless Sensor Networks—An Information Processing Approach. San Francisco, CA: Morgan Kaufmann Publishers.