2006

SPLAI: Computational Finite Element Model for Sensor Networks

Ruzana Ishak

Shadaruddin Salleh

Stephan Olariu
Old Dominion University

Mohd.Ismail Abdul Aziz

Original Publication Citation

Ishak, R., Salleh, S., Olariu, S., & Aziz, M. I. A. (2006). SPLAI: Computational finite element model for sensor networks. *Mobile Information Systems, 2*(1), 77-92.

Mobile Information Systems 2 (2006) 77–92
IOS Press

SPLAI: Computational finite element model for sensor networks

Ruzana Ishak[a], Shaharuddin Salleh[b], Stephan Olariu[c] and Mohd.Ismail Abdul Aziz[d]
[a]*Department of Mathematics, Universiti Teknologi Malaysia, 54100 Jalan Semarak, Kuala Lumpur, Malaysia*
E-mail: ruzana@citycampus.utm.my
[b]*Department of Mathematics, Universiti Teknologi Malaysia, 81310 Johor Bahru, Malaysia*
E-mail: ss@mel.fs.utm.my
[c]*Department of Computer Science, Old Dominion University, Norfolk, Virginia 23529, USA*
E-mail: olariu@cs.odu.edu
[d]*Department of Mathematics, Universiti Teknologi Malaysia, 81310 Johor Bahru, Malaysia*
E-mail: m_ismail@mel.fs.utm.my

Abstract. Wireless sensor network refers to a group of sensors, linked by a wireless medium to perform distributed sensing task. The primary interest is their capability in monitoring the physical environment through the deployment of numerous tiny, intelligent, wireless networked sensor nodes. Our interest consists of a sensor network, which includes a few specialized nodes called processing elements that can perform some limited computational capabilities. In this paper, we propose a model called SPLAI that allows the network to compute a finite element problem where the processing elements are modeled as the nodes in the linear triangular approximation problem. Our model also considers the case of some failures of the sensors. A simulation model to visualize this network has been developed using C++ on the Windows environment.

Keywords: Wireless sensor network, corona, wedges, routing and finite element method

1. Introduction

Recent advances in micro-electro-mechanical systems (MEMS) have led to the rapid development of micro-sensors. Such sensors are generally equipped with data processing and communication capabilities. Wireless sensor networks are a new class of ad hoc networks that are expected to find increasing deployment in coming years. A sensor network is a collection of tiny disposable and low-power devices. A sensor node is a device that transforms a sensed attribute into something that provides information about its vicinity. Each sensor in the network sends such collected data, usually via radio transmitter, to a command center as a sink either directly or through a data concentration center as a gateway.

Sensor networks have been proposed for a wide variety of application areas. Some examples are such as intrusion detection and tracking, where sensors are deployed along the border of a battlefield to detect classify and track intruding personnel and vehicles. There also a lot of interest in the environmental monitoring, where specialized sensor nodes that are able to detect temperature changes and smoke can be deployed in high-risk of a forest, to give early warning of forest fires. In traffic analysis, traffic sensor network can monitor vehicle traffic on a highway or a congested part of a city. In security, surveillance

sensor networks can be used to provide security in an art gallery or shopping mall. At the network layer of a sensor network, the main aim is to find ways for energy-efficient route setup and reliable relaying of data from the sensor nodes to the sink. So, a special multihop wireless routing protocol between the sensor nodes and the sink are needed to maximize the lifetime of the network. Problems encountered in effective implementation of the sensor networks include coverage area, data routing and power efficient.

In this paper, we discuss the data routing issues by considering the limited energy supply in sensor networks. The issues involve three phases: Interest Request, Message Transmission and Data Computation. We propose SPLAI, or *Sensor Protocols for 2D-Linear Approximation Information*, which is a dissemination algorithm based on the data-centric approach for modeling the finite element solution to sensor networks. Our routing technique in SPLAI requires finding the spanning tree and transmitting messages to the processing element by finding the shortest paths to optimize the routing. We also consider the self-organizing features of the network in the case when some nodes fail. The network adapt to the changes by updating the current information to the rest of the nodes. The computational model discussed in SPLAI consists of the two-dimensional (2D) linear triangular approximation of the finite element method.

The paper is organized into four sections. In Section 1, we review the application and implication of routing issues in sensor networks. Section 2 describes the routing protocol problem that considers the limited energy supply constraint. Several data-centric routing approaches are discussed in Section 3. We describe a method for locating the dynamic coordinates of the sensor nodes in the network in Section 4. This is followed by some discussion on SPLAI in Section 5. We further describe the development of SPLAI involving the dissemination method using Prim's Algorithm and data computation using the linear triangular approximation of the finite element method in Section 6. Section 7 is the summary and conclusion.

2. Problem formulation

The problem is stated as follows: Given a set of sensors and a few specialized nodes termed as processing elements (PEs) in a sensor network, how do the PEs react to the data provided by the sensors? Two sub-problems are associated here: first is locating the position of all the sensor nodes relative to the PEs in a process called *training*, and second is computing the data supplied by the sensors once training has been completed. The PEs in this context are assumed to have unlimited communication capability for transmitting and receiving messages to/from the sensors, and enough memory and processing capability to perform tasks such as mathematical calculations.

The sensor network in our discussion is modeled as an undirected graph. In this model, a node in the graph represents a sensor while an edge between two nodes represents the communication link between the sensors. The absence of a link between a pair of nodes means the two nodes are not within their transmission range.

Our case study in this paper involves the development of a finite element model for computing the temperature in a convex hull that is bounded by sensors in a sensor network. This problem involves finding a practical method for disseminating and transmitting messages in sensor networks. We also explore the problem of approximating a value at new location within the network. Disseminating involves the transmission of messages to a group of nodes in the network, which is a problem of broadcast through a spanning tree. Transmitting involves the transmission between a pair of nodes, which is from sensor node to the processing element. A term, approximating is the computation of a message to predict messages at new location within the network. Our contribution consists of a protocol for routing and improves its performances through the development of our model, SPLAI.

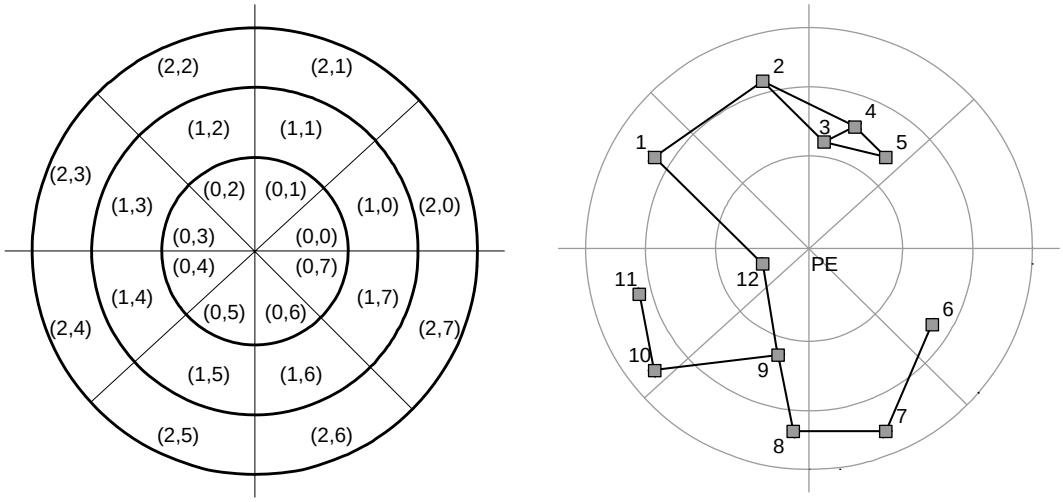

Fig. 1. Dynamic coordinates from PE in the form of coronas and wedges (left) and the location of the sensor nodes (right).

3. Overview of sensor network routing protocols

Advances in wireless communications and electronics have fostered the development of relatively in-expensive and low-power wireless sensor nodes. The sensor nodes are extremely small in size and communicate un-tethered in short distances. These small devices are incorporated with sensing, data processing and communicating components, to collect data, monitor equipment, and transmit information, which leverages the idea of sensor networks. Each sensor node in these networks operates autonomously with no central point of control in the network and communicates using infrared devices or radios.

The main goal in conventional wireless networks such as mobile ad-hoc network or MANET is to provide high quality of service and high bandwidth efficiency when mobility exists. In contrast, in a sensor network conservation of energy is considered to be important than the performance of the network. Therefore, the current routing protocols designed for traditional networks cannot be used directly in a sensor network due to the following reasons [1]:

1. Sensor networks are data-centric, where data is requested based on certain attributes.
2. Application specific, where the requirements of the network will change with the application.
3. Aggregated data in the case that adjacent nodes may have similar data.
4. Since the model is data-centric, each sensor is assumed to have non-unique id. Additionally, large number of nodes in the network implies large ids, which might be substantially larger than the actual data being transmitted.

To comply with the above requirements, new routing schemes have been proposed. Routing protocols must select the best path to minimize the total power and to maximize the lifetime of all nodes. Due to such differences, many new algorithms have been proposed in the problem of routing data in sensor networks. Most of these routing mechanisms consider the characteristics of sensor nodes along with the application and architecture requirements [2].

Routing protocols in sensor networks can be classified as data-centric, hierarchical or location-based. Data-centric protocols are query-based and depend on the naming of desired data, which helps in eliminating many redundant transmissions. The sink sends queries to certain regions and waits for

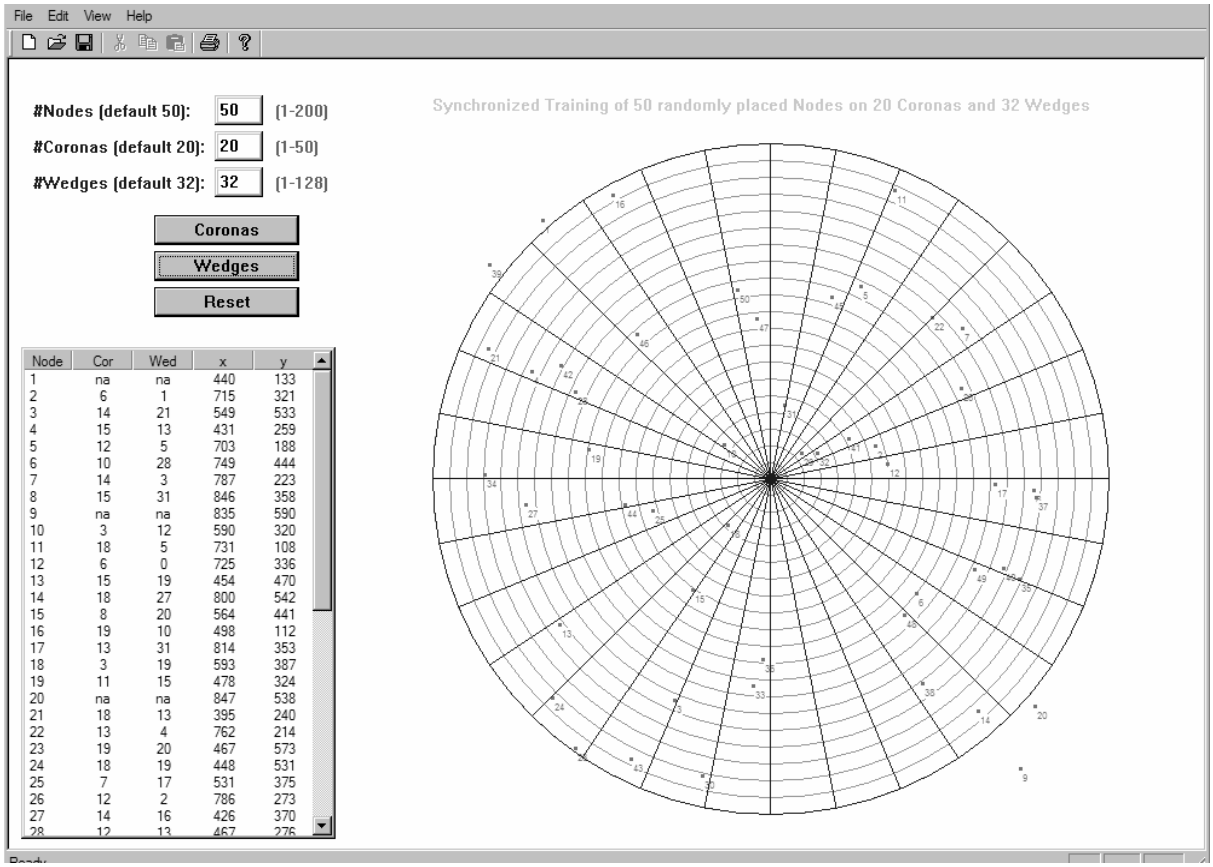

Fig. 2. Training of 50 nodes using 20 coronas and 32 wedges in SPLAI.

data from the sensors located in the selected regions. Since data is being requested through queries, attribute-based naming is necessary to specify the properties of data.

Some established data-centric routing protocols are *Flooding and Gossiping, Sensor protocols for information via negotiation* (SPIN) *and Directed Diffusion*. Flooding and Gossiping are two classical mechanisms to relay data in sensor networks without the need for any routing algorithms and topology maintenance. In flooding, a node wishing to disseminate a piece of data across the network starts by sending a copy of this data to all of its neighbors. Each sensor receiving a data packet broadcasts it to all of its neighbors except the node from which it just received the data. This process continues until the packet arrives at the destination or the maximum number of hops for the packet is reached. Gossiping is a slightly enhanced version of flooding where the receiving node sends the packet to a randomly selected neighbor, which picks another random neighbor to forward the packet to and so on [2]. SPIN considers data negotiation between nodes in order to eliminate redundant data and save energy [7]. Directed Diffusion has been developed and has become a breakthrough in data-centric routing [8].

A different approach from the hierarchical protocols is achieved by clustering the nodes so that cluster heads can do some aggregation and reduction of data in order to save energy. As mentioned in [1–3], cluster formation is typically based on the energy reserve of sensors and sensor's proximity to the cluster head. The hierarchical routing approaches for sensor networks include LEACH (Low-Energy Adaptive Clustering Hierarchy), PEGASIS (Power-Efficient GAthering in Sensor Information Systems) and TEEN

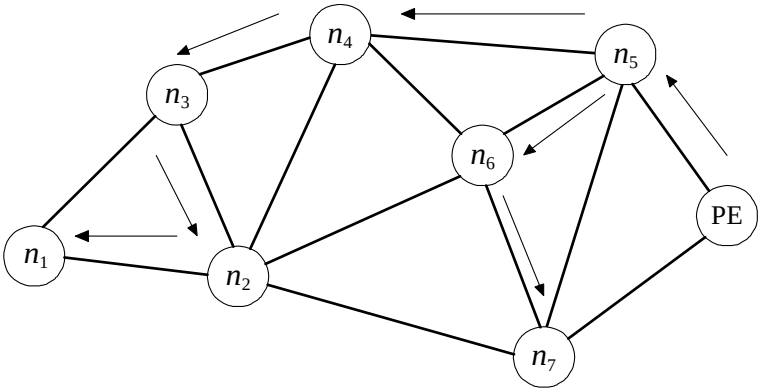

Fig. 3. A route initiated by PE in Interest Request Phase using MST.

(Threshold sensitive Energy Efficient sensor Network protocol). Another approach uses the location-based protocols, which utilize the information on the nodes location to relay data to the desired regions rather than the whole network. In most cases, location information is needed in order to calculate the distance between two particular nodes so that energy consumption can be estimated. MECN (Minimum Energy Communication Network), GAF (Geographic Adaptive Fidelity) and GEAR (Geographic and Energy-Aware Routing) are among the location-based protocols in sensor networks [1,3].

4. Training the sensor nodes

A wireless sensor network is a self-organized network which does not depend on external help in establishing itself. The position of each node in the network is determined through a process called *training*. Basically, training involves a repeated transmission of signals with varying strengths and directions from the PE (sink) to the sensor nodes. One difficulty in locating the nodes is the fact that the Cartesian or the xy-coordinate system is not practical for implementation as the exact location of each node may not be traced. It is not wise to assume each node has a position locating device such as the GPS as the sensor network as a whole is self-organized. Therefore, an approximation based on a dynamic coordinate system will be more appropriate and practical in its implementation. Dynamic coordinate, in this context, refers to a coordinate system relative to the position of the sink. Training is said to be successfully completed once all the nodes that lie within the transmission range to the sink have been assigned with the dynamic coordinates.

We assume only one processing element is used in this model. This processing element has enough computing power and memory to train the sensor nodes, and to perform all the necessary computations from the data gathered from the nodes. In addition, each sensor node is assumed to be awake at all time to respond to the signals from PE during the training. This assumption may not describe the real scenario of the sensor nodes which are in sleep most of the time for saving energy. However, this assumption is necessary in the simulation in order to focus to the computational aspect of the model.

Our model is based on the work of Haseebulla [6] which implements the coronas (c) and wedges (w), coordinate system. In this coordinate system, a *corona* is an annular area whose center is at PE. A *wedge* is a sector measured from the $x-$axis in the anti-clockwise direction with its center at PE. The origin of the coordinate system, or the *sink*, is the starting coordinate at (0,0) which lies in the first sector in the

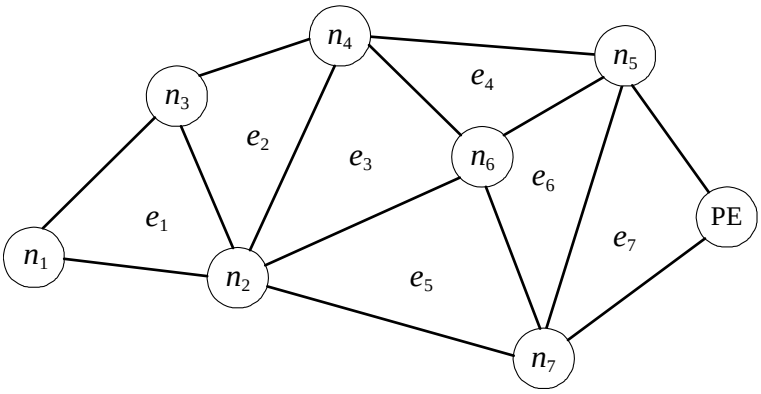

Fig. 4. Sensor network with seven nodes and one processing element.

innermost circle. Figure 1 (left) shows a simple corona-wedge coordinate system with three coronas and eight wedges with their assigned coordinates.

Training the nodes in the network involves a successive transmission of signals from PE to the nodes. The strength of a signal corresponds directly to the corona number: the weakest indicates Corona 0 while the strongest is Corona $C-1$, where C is the number of coronas in the network. In addition, for training the nodes according to the wedge number, PE transmits its strongest signals successively in angular, anti-clockwise directions starting from the $x-$axis. PE determines the corona and wedge numbers of a node only when a transmitted signal receives a respond from the node. If no respond is received then the node is assumed to be outside of the transmission range and the node is said to be untrained.

The first phase of the training involves the coronas only. For C coronas, c_i for $i = 0, 1, 2, \ldots, C-1$, the training starts from the innermost corona, c_0. The PE transmits its weakest signal in the circular region identified as Corona 0. Upon receiving the beacon, the nodes belonging to this corona immediately respond and they are instantly identified to belong to this corona. PE repeats by increasing the transmission strength to the second level, and the nodes that respond to the signal are those that may belong to Corona 0 or 1. However, the nodes in Corona 0 have been identified earlier and, therefore, their complements are assigned to Corona 1. The same process is repeated until the last corona, c_{C-1}, which corresponds to the maximum signal strength. It is obvious from this approach that only nodes that are within the transmission range will respond to the signals from PE. Those that don't respond are considered the outliers, and they are assumed to be outside of the network.

The next phase of training involves the wedges. Training is achieved by having the PE transmitting its maximum signals in directional positions according some predetermined angles measured from Wedge 0. In general, a network with W equal-width wedges requires a directional transmission of $2\pi/W$ as its angle. For example, the directional angle for a network with eight wedges is $2\pi/8$, as illustrated in Fig. 1 (left).

Figure 1 (right) shows 12 nodes scattered randomly within the transmission range of PE. The figure shows the result from training using three coronas and eight wedges. Each node has a limited transmission range as illustrated as edges in the graph. Note that the dynamic coordinates may not be unique as a coordinate location may not be attached to one node only. As shown in the figure, node 1 is located at (2,3) while nodes 3, 4 and 5 share the same coordinate, $(1, 1)$.

Assuming the origin of the Cartesian coordinates at the sink, the position at (c, w) can be transformed

to the Cartesian coordinates (x, y) using the following relationships:

$$x = c\cos\frac{2\pi}{W}w, \tag{1a}$$

$$y = c\sin\frac{2\pi}{W}w. \tag{1b}$$

Conversion to real coordinates using Eqs (1a) and (1b) is very useful especially in cases where the real location of the sink is known using a measuring device such as the GPS.

The training of nodes into the dynamic coordinates is implemented visually in SPLAI. Figure 2 shows a training simulation of 50 sensor nodes that are scattered randomly, which are grouped into 20 coronas and 32 wedges in SPLAI. Their location in (c, w) are determined through a series of repeated transmissions from the sink (PE), and these values are converted into their corresponding (x, y) values using Eqs (1a) and (1b).

5. Routing model

Once training is completed, the next step is to form a routing mechanism to allow communication among the sensor nodes. This step is important as a value sensed by a node needs to be passed to PE to be used in the computational stage. PE requires all data read from the nodes in the network before performing the necessary computation. However, it is not possible for a node to transmit its value directly to PE as it has a short and limited transmission capability. In order to transmit its value to PE, the node will have to form a routing path through other nodes until it reaches PE.

Basically, routing between the nodes and PE in the network involves finding the shortest paths between pairs of points in the graph. This approach requires finding the shortest path from every sensor node to PE, which may take a considerable computing time and overhead. Another practical approach is to compute the minimum spanning tree of the graph with PE as its root. This approach has the advantage of eliminating redundant paths that may result from the shortest paths.

Two modes of routing for allowing communication in the sensor network are discussed in SPLAI. First is the *Interest Request Phase* which is a message transmission from the sink (PE) to the sensor nodes using a routing path based on the minimum spanning tree of the network. This path is necessary to enable PE to send request for values to the nodes. The other mode is the *Message Transmission Phase* which is a message transmission on the sensed values from the nodes to PE.

5.1. Interest request phase using the minimum spanning tree

In SPLAI, we adopt the second approach of the training, that is, using the minimum spanning tree for routing from PE to the nodes. The training requires computing the minimum spanning tree of the graph with its root at PE using the Prim's algorithm. The task of finding the minimum spanning tree is performed by PE once the information about the network is known. The spanning tree is important in this context as it represents the interest request phase for PE as the processing unit needs to know the optimum path for communicating with all the sensor nodes in the network.

The main idea behind SPLAI is outlined as follows: PE disseminates an interest query packet for the values to each node in the network using the minimum spanning tree path. Upon receiving the interest

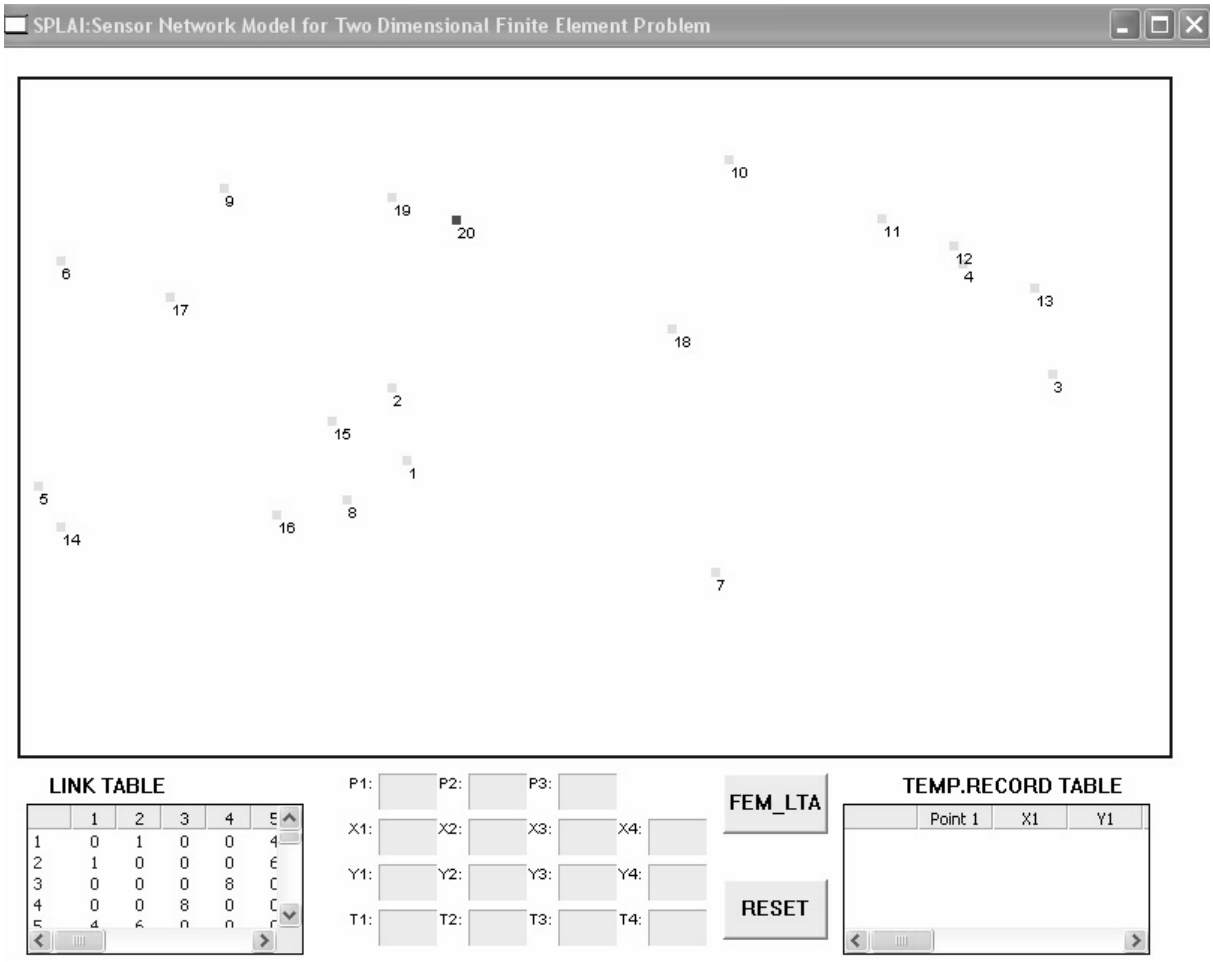

Fig. 5. Illustrating SPLAI model interface.

packets, each node in the network starts periodically sending its temperature data or measurements back to the processing element using the same path. Since many or all of the nodes will be sending their data at the same time, it would be more efficient to combine these readings into a single packet. As data flows, it can be aggregated along the way. PE stores the data and performs the necessary computation to evaluate the temperature at any point based on the finite element technique.

SPLAI includes tasks such as training the nodes, routing for data communication and performing computation on the sensed data. The model consists of a data-centric algorithm as it involves the dissemination of data from the sensor nodes, and its transmission to PE. Upon receiving the required data, PE stores them in its memory before applying the necessary computational steps.

Figure 3 shows a sensor network consisting of a processing element (PE) and seven randomly distributed sensor nodes, n_i for $i = 1, 2, \ldots, 7$, scattered in a two-dimensional area Q. Each sensor node, n_i, has the corona-wedge coordinates of (c_i, w_i) which is transformed to (x_i, y_i) using Eqs (1a) and (1b). Each sensor has a sensing range of R_i, that is, it can sense a metric such as temperature (or pressure and sound in a similar situation), that is within a distance of R_i from n_i. The figure also shows message routing path from PE to the nodes on the minimum spanning tree (shown as arrows).

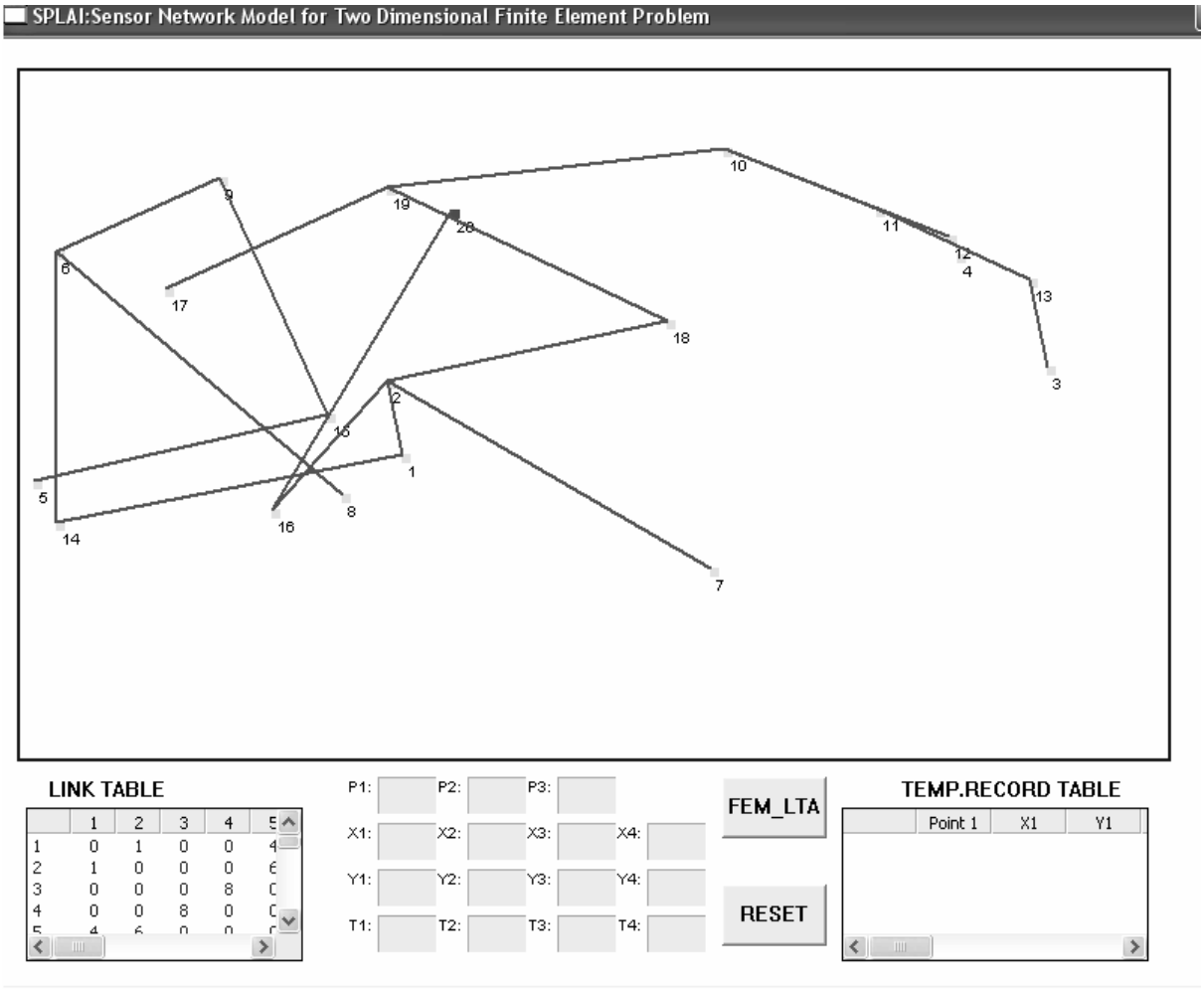

Fig. 6. Illustrating the interest request phase.

The sensor network behaves like an *ad hoc network* where the nodes can both receive and transmit messages. Each node in the network has a transmitter and a receiver to perform these functions to act as a router. To facilitate message transmission, a routing protocol called DSDV, or *Destination Sequenced Distance Vector*, is adopted. In this protocol, a node updates its routing and neighbor information proactively whenever new information is received.

In our work, G represents a network of sensor nodes which is assumed to be connected and undirected in this model. The dissemination method of an interest query packet in the network is a graph-broadcasting model of sensor nodes. A crucial problem in this case is to find the minimum spanning tree of the graph with the root as the sender. This problem is solved using the Prim's algorithm, which is represented in its visual form in [11].

The algorithm begins with an empty graph, T. The strategy in the Prim's algorithm consists of building a tree in T one node at a time starting from any node in the graph. From the starting node, a link with the minimum weight to its neighboring node is added to T. Next, the shortest link emanating from the two nodes becomes the minimum link and added into T. This added link must not form a circuit in T,

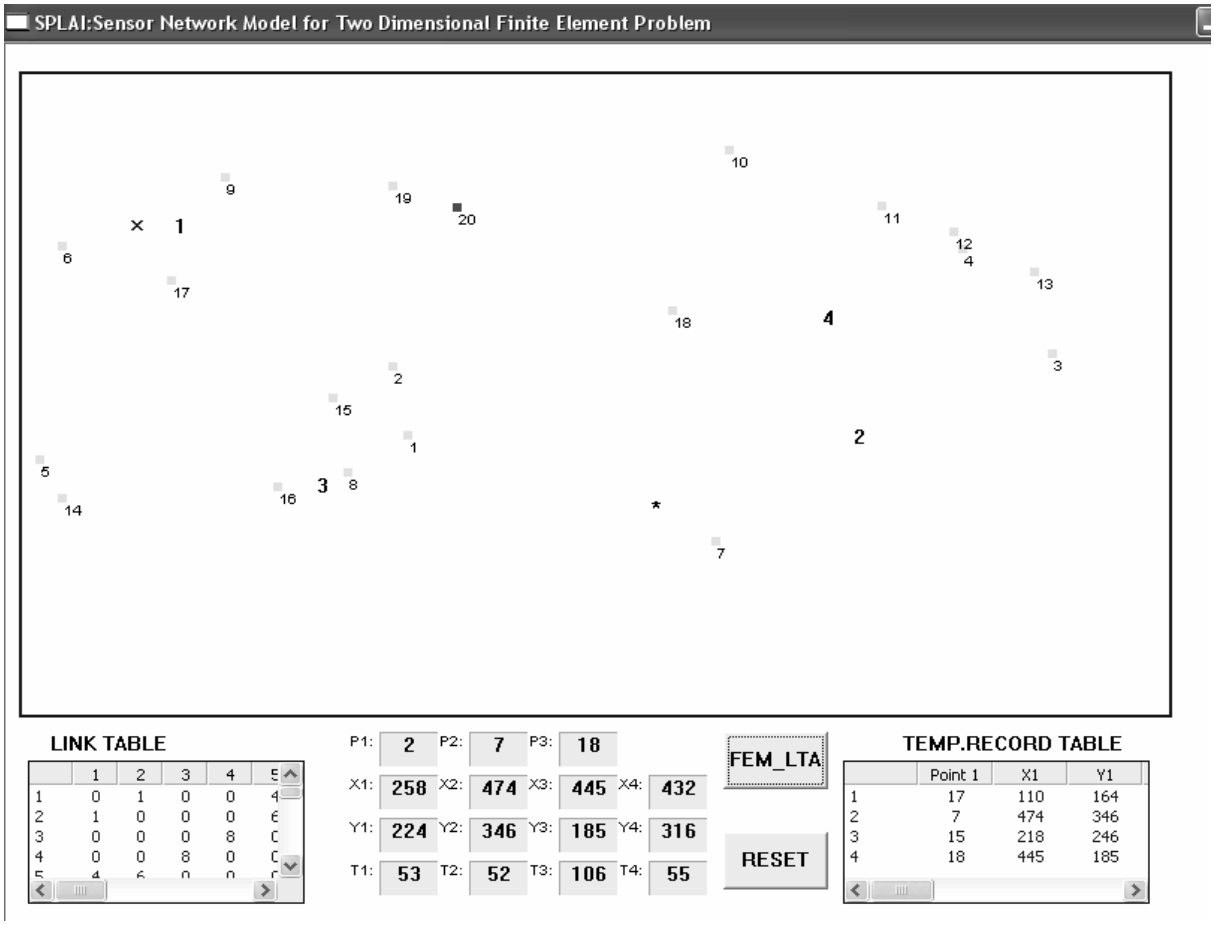

Fig. 7. Illustrating the data computation phase.

otherwise the move is rejected. The step repeats from the second node to the third, and subsequently until all the nodes have been included.

The following steps summarize the Prim's algorithm for finding the minimum spanning tree of a graph:

```
Read the information on the weighted graph G;
Create an empty graph, T;
Choose the first node, and add into T;
do until T becomes a spanning tree of G
  Choose the minimum link, originating from the node;
  if the new link does not have a circuit
    Add the link and its new node into T;
  else
    Reject the move;
```

Table 1
Routing table structure for n_2

Destination	Next	h_n	s_n
n_1	1	1	n_1-160
n_2	2	0	n_2-200
n_3	3	1	n_3-666
n_4	4	1	n_4-76
n_5	6	2	n_5-888
n_6	6	1	n_6-550
n_7	6	2	n_7-444
PE	6	3	PE-250

5.2. Message transmission phase

The next step upon receiving the interest query packet of temperature value from the processing element is the message transmission phase. Each sensor node aggregates their sensed values to PE using the route determined from the DSDV routing protocol [9]. DSDV provides the shortest paths of all pairs of nodes in the network. In DSDV, each node has a table for updating its routing information with its adjacent nodes, and this allows the simultaneous updates of the values all the way to the sink (PE). PE receives this vital information, and use these values for performing the subsequent computational steps.

Table 1 shows the routing information of the node n_2 in the graph from Fig. 3 using SPLAI. In a basic distance vector routing protocol, every host maintains a routing table containing the distances from it to all possible available destinations. In the table, *destination* refers to the address of the destination node, while *next* is the next hop along the path of the current node to get to the destination. h_n is the total number of hops required to get to the destination. The sequence number originated from destination is represented by s_n.

In DSDV, the number of hops is the metric for computing the path from the source to its destination. DSDV extends the basic Bellman-Ford and the Floyd-Warshall algorithms by attaching a sequence number to each route table entry. A route with a higher recent sequence numbers is always preferred as the basis for making the forwarding decision. In this case, the nodes can distinguish the stale routes from the new ones, and this avoids the formation of routing loops in the network. The message transmission phase is based on the DSDV routing protocol using shortest paths outlined by the Floyd-Warshall algorithm. The algorithm, as described in its visual form in [11], is summarized as follows:

1. Each sensor node is referred to its routing table entries.
2. A sensor node sends its temperature data to the *next* sensor node that is its immediate neighbor in the network.
3. The receiving sensor node then sends the value to its *next* sensor node until it reaches the PE.
4. If broken links occur in the network, the nearest sensor node will advertise a route table by assigning a metric of ∞ to its table. In this case, every sensor node in the network updates its table. Any new information will not be sent to the node in the next transmission.
5. The sequence number is then increased by 1 for every broken link detected.

The source in our model is PE, which then disseminates interest packets to other nodes in the network. PE stores this value and shares the temperature data with other nodes.

6. Computational model using finite element method

In our model, PE is capable of performing numerical computations on the data provided by the sensor nodes. This step is performed once training has been completed, and the network has been successfully organized. The computational model is useful in many numerical applications such as in approximating the temperature of any point inside the network using the finite element method. We discuss the problem of computing the temperature of the points inside the network using the linear triangular approximation (LTA) technique in finite element method [4]. The linear triangular approximation method computes the temperature in any point inside a triangle by taking the temperatures at the vertices of the triangle as the finite elements.

We describe our computational model using an example of a network with one processing element and seven sensor nodes, as shown in Fig. 3. Figure 4 shows the triangular elements formed from the graph in Fig. 3. There are seven areas in the bounded region of the network, marked as e_i for $i = 1, 2, \ldots, 7$.

One of the main constraints in a sensor network is its limited energy supply. The network needs to limit its activities that consume energy. To achieve this objective, only one PE is used for the computations, while the sensor networks supply the necessary data. Upon user query, PE performs the computation to approximate the value at other location, which is out of distance of R_i from n_i using the values at three other node or locations.

Figure 4 shows the finite element model formed from the sensor nodes which are scattered in the plane. Each sensor node reads the temperature at its location then transmits this value to PE through the shortest path route. The temperature field within an element is given by

$$\theta = \lambda_1\theta_1 + \lambda_2\theta_2 + \lambda_3\theta_3, \text{ or } \theta = \lambda\theta^{(e)}, \tag{2}$$

where θ_i is a temperature and λ_i is the shape function at n_i. This is conveniently represented in terms of the pair α, β as follows:

$$\lambda = [\alpha, \beta, 1 - \alpha - \beta], \tag{3}$$

$$\lambda_1 + \lambda_2 + \lambda_3 = 1. \tag{4}$$

The shape function describes how the temperature varies in each element. Using the isoparametric representation, the displacements inside the element are now written using the shape functions and the nodal values of the unknown displacements field. We have

$$u = \lambda_1 q_1 + \lambda_2 q_3 + \lambda_3 q_5, \tag{5a}$$

$$v = \lambda_1 q_2 + \theta_2 q_4 + \theta_3 q_6, \tag{5b}$$

where q_1, q_2, q_3, q_4, q_5 and q_6 show two directions for each node in the triangle, and u and v are the nodal values of the unknown displacements. The above equation can be written as

$$x = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3, \tag{6a}$$

$$y = \lambda_1 y_1 + \lambda_2 y_2 + \lambda_3 y_3. \tag{6b}$$

Table 2
Element Connectivity

Element (e)	(x_1, y_1)	(x_2, y_2)	(x_3, y_3)
1	2	3	1
2	2	4	3
3	6	4	2
4	6	5	4
5	7	6	2
6	7	5	6
7	8	5	7

Table 3
Information received by PE based on DSDV

Node	Location, (x_i, y_i)	Temp., θ	Element, e
n_1	(2,4)	100	e_1 =(4,8)
n_2	(6,8)	80	e_2 =(6,12)
n_3	(4,12)	120	$e_3 = (10, 14.5)$
n_4	(10,16)	200	$e_4 = (14, 10)$
n_5	(16,14)	140	$e_5 = (12, 6)$
n_6	(14,9)	60	$e_6 = (15, 10.5)$
n_7	(16,4)	35	$e_7 = (17, 6)$
PE	(22,8)	23	

Substituting the value of λ from Eq. (3) into Eqs (6a) and (6b) produce

$$x = (x_1 - x_3)\alpha + (x_2 - x_3)\beta + x_3, \tag{7a}$$

$$y = (y_1 - y_3)\alpha + (y_2 - y_3)\beta + y_3, \tag{7b}$$

where $(x_1, y_1), (x_2, y_2)$ and (x_3, y_3) are the coordinates of n_1, n_2 and n_3, respectively. Using the notation, $x_{ij} = x_i - x_j$ and $y_{ij} = y_i - y_j$, Eq. (5) becomes

$$x = x_{13}\alpha + x_{23}\beta + x_3, \tag{8a}$$

$$y = y_{13}\alpha + y_{23}\beta + y_3. \tag{8b}$$

The values of λ_1, λ_2 and λ_3 are obtained by solving Eqs (8a) and (8b). It follows that the temperature at location (x, y) in each element in the network can be calculated by substituting these values into Eq. (2).

Table 2 shows the connectivity of the elements in the network. As depicted in the table, most standard finite element codes use the convention of going around the element in a counterclockwise direction to avoid calculating a negative area.

Table 3 is an illustration of the network in Fig. 3, which shows the temperature, location and element of each node. The temperature at e_5 is computed by referring to the information given in Tables 2 and 3. By substituting the readings from these tables into Eqs (8a) and (8b) at this node we obtain the following simultaneous equations:

$$10\alpha + 8\beta + 6 = 12$$

Table 4
Temperature approximation at the finite elements

e_i	1	2	3	4	5	6	7
θ_i	99.9	130	174	77.648	54.524	89.5	37.416

Table 5
Link Table for 19 nodes and the PE

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	0	1	0	0	4	0	4	8	8	0	0	0	0	1	6	10	2	8	4	3
2	1	0	0	0	6	5	2	4	8	0	0	0	0	4	5	2	3	1	8	3
3	0	0	0	8	0	0	0	0	0	0	7	7	4	0	0	0	0	0	0	0
4	0	0	8	0	0	0	0	0	0	5	9	9	5	0	0	0	0	6	0	0
5	4	6	0	0	0	8	0	10	9	0	0	0	0	5	1	5	9	0	0	0
6	0	5	0	0	8	0	0	0	2	1	0	0	0	1	3	5	9	0	6	0
7	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0
8	8	4	0	0	10	2	0	0	4	0	0	0	0	7	6	6	6	6	4	7
9	8	8	0	0	9	1	0	4	0	0	0	0	0	10	3	5	7	0	2	2
10	0	0	0	5	0	0	0	0	0	0	1	4	6	0	0	0	0	9	3	7
11	0	0	7	9	0	0	0	0	0	1	0	6	3	0	0	0	0	5	0	0
12	0	0	7	9	0	0	0	0	0	4	6	0	7	0	0	0	0	4	0	0
13	0	0	4	5	0	0	0	0	0	6	3	7	0	0	0	0	0	7	0	0
14	1	4	0	0	5	1	0	7	10	0	0	0	0	0	5	9	8	0	0	0
15	6	5	0	0	1	3	0	6	3	0	0	0	0	5	0	3	10	6	5	9
16	10	2	0	0	5	5	0	6	5	0	0	0	0	9	3	0	7	0	3	1
17	2	3	0	0	9	9	0	6	7	0	0	0	0	8	10	7	0	0	1	2
18	8	1	0	6	0	0	9	6	0	9	5	4	7	0	6	0	0	0	1	5
19	4	8	0	0	0	6	0	4	2	3	0	0	0	0	5	3	1	1	0	5
20	3	3	0	0	0	0	0	7	2	7	0	0	0	0	9	1	2	5	5	0

$$-4\alpha + \beta + 8 = 6$$

Solving the above equations, we get $\alpha = 0.5238$ and $\beta = 0.09525$. It follows that Eqs (4) and (5) produce $\lambda_1 = 0.5238$, $\lambda_2 = 0.09525$ and $\lambda_3 = 0.38095$. Finally, we obtain the temperature of e_5, as follows:

$$\theta = \lambda\theta^{(e)} = \begin{bmatrix} 0.5238 & 0.09525 & 0.38095 \end{bmatrix} \begin{bmatrix} 35 \\ 60 \\ 80 \end{bmatrix} = 54.525.$$

Applying the above steps to all other elements in the network we get the results as shown in Table 4. This phase provides instant information including the case whereby one or more sensors in the network are weak or not functioning properly.

In general, the routing and computation protocols in SPLAI for a sensor network using one processing elements can be summarized as follows:

1. The processing element, PE broadcasts an interest query packet to get the temperature data in the region by following the path specified by the minimum spanning tree.
2. When the packets reach the sensor nodes, they aggregate their temperatures to their nearest neighbors based on their routing table entries until the message reaches PE.
3. PE is responsible for receiving, storing, processing and sharing the temperature data with other networks. In the processing, PE can perform the computation to approximate the temperature of any location even if it is out of range from s_i.

Table 6
Indicating the result shown in the TEMP.RECORD TABLE

	P1	X1	Y1	T1	P2	X2	Y2	T2	P3	X3	Y3	X4	Y4	*T4
1	17	110	164	83	9	146	92	33	6	37	140	113	119	43
2	7	474	346	52	3	699	215	119	18	445	185	567	266	89
3	15	218	246	89	16	181	308	37	8	228	298	208	300	54
4	18	445	185	106	13	687	158	71	10	483	73	546	183	98

7. SPLAI simulation model

SPLAI was developed using the C++ programming language running on the Windows environment. Figure 5 shows the interface of our SPLAI model network for massively deployed sensor nodes. The sensor nodes are scattered randomly in a rectangular area with a transmission range of 250 units. We describe this model using a case study using 19 sensor nodes and one PE. Figure 5 shows this scenario where node 20 is the processing element, or the sink, while the other nodes numbered from 1 to 19 are the sensor nodes. SPLAI also displays the link table and temperature record table.

Table 5 shows the link table between the sensor nodes based on their distance apart from the case in Fig. 5. The values shown in the table provide a vital information about routing on the interest request phase from PE to the nodes and the message transmission phase from the nodes to PE.

Figure 6 shows the minimum spanning tree in the Interest Request Phase initiated by the PE to perform the path in the network before the sensors start sending and receiving the data. The tree is constructed using the Prim's algorithm. This phase provides an important step in having PE to present its request for values to the sensor nodes.

Once the values are received from the sensor nodes, PE becomes ready to perform the computational steps for computing the temperatures at the elements. Figure 7 shows the data computation phase in SPLAI for approximating the temperatures using the linear triangular approximation of the finite element method. The location of the nodes is indicated as bold number 1, 2, 3, 4 and * in the simulation.

Table 6 shows the results in the form of approximated temperature values of the 19 elements in the network using the finite element method.

8. Conclusion and future work

In this paper, we propose SPLAI as a tool for disseminating, transmitting and computing in a sensor network. Our protocol is based on the data-centric approach involving the creation of dynamic coordinates for each node before its real location is determined. The disseminating problem has been modeled as a spanning tree based on Prim's Algorithm, while the transmission is a table-driven shortest path protocol based on the DSDV algorithm. One obvious advantage from our approach is SPLAI considers the problem of limited energy supply among the sensor nodes through the computation phase of the finite element method using the linear triangular approximation (LTA) approach.

The computational model in SPLAI is based on one processing element. Obviously, it is possible to extend the work by incorporating more than one processing element. Two or more processing elements can be used to enhance the training and routing methods as well as transforming the computational step into a parallel model. The use of multiple processing elements has the advantage of speeding up training and routing which contributes in the energy constraint scenario of the sensor network. Energy saving is a crucial factor in a sensor network which has to be considered seriously before the network can be deployed. This requirement is an open problem that will definitely attract further research into this area.

References

[1] D.P. Agrawal and O.A. Zeng, *Introduction to Wireless and Mobile Systems*, Brooks/Cole-Thomson Learning, USA, 2003.
[2] K. Akkaya and M. Younis, A Survey on routing protocols for wireless sensor networks, *Ad Hoc Networks Elsevier Computer Science Publications* (2003).
[3] I. Akylldiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, A survey on sensor networks, *IEEE Communications Magazine* **40**(8) (2002), 102–114.
[4] T.R. Chandrupatla and A.D. Belegundu, *Introduction to Finite Elements in Engineering*, Third Edition. Prentice Hall, 2002.
[5] G. Chartrand, Oellerman and R. Ortrud, *Applied and Algorithmic Graph Theory*, McGraw-Hill,Inc., 1993.
[6] Haseebulla Khan, Localization in wireless networks, MSc. Thesis, Department of Computer Science, Old Dominion University, Oktober 2005.
[7] W. Heinzelman, J. Kulik and H. Balakrishnan, *Adaptive Protocols for Information in wireless Sensor networks*, Fifth ACM/IEEE mobicom Conference, Seattle, WA, 1999.
[8] C. Intanagonwiwat, *Directed Diffusion: A Scalable and robust communication paradigm for sensor networks*, ACM Mobicom, 2000.
[9] C.E. Perkins and B. Bhagwat, *Highly dynamic destination sequenced distance vector routing*(*DSDV*) *for mobile computers*, Proceedings of SIGCOMM'94, 1994, 234–244.
[10] M.E. Royer and C.K. Toh, A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks, *IEEE Personal Communications* (1999).
[11] S. Salleh, A.Y. Zomaya, S. Olariu and B. Ssnugi, Numerical simulations and case studies using Visual C++, *Net, Wiley-Interscience*, New York, 2005.
[12] R.J. Wilson, *Introduction to Graph Theory*, Longman Group Edited, 1985.

Ruzana Ishak is a Lecturer of Mathematics Department at Univ. Teknologi Malaysia, Kuala Lumpur. She received the B.Sc degree in Computational Maths & CAGD from Univ. Sains Malaysia, Penang in 1993 and the M.Sc degree in Mathematics from Univ.Teknologi Malaysia, Johor Bahru in 2002. She is currently pursuing a Phd degree from Univ. Teknologi Malaysia in collaboration with CS Department, Old Dominion University, Virginia. Ruzana's dissertation is on wireless sensor networks.
Email: ruzana@citycampus.utm.my

Stephan Olariu is a tenured full professor in Computer Science at Old Dominion University. He is a world-renowned technologist in the areas of parallel and distributed systems, parallel and distributed architectures and networks. He was invited and visited more than 120 universities and research institutes around the world lecturing on topics ranging from parallel algorithms, to graph theory, to wireless networks and mobile computing, to biology-inspired algorithms and applications, to telemedicine, to wireless location systems, and sensor network applications. Professor Olariu is the Director of the Sensor Networks Research Group at Old Dominion University. He has published 200+ archival journal articles and 100+ conference papers.

Professor Olariu earned his BSc, MSc and Ph.D. (Computer Science) at the McGill University, Montreal, Canada. He has received an NSF Research Initiation award. He is an Associate Editor of "Networks" and and serves on the editorial board of "IEEE Transactions on Parallel and Distributed Systems" and "Journal of Parallel and Distributed Computing".

Prof. Olariu's current research interests are in the area of parallel and distributed systems, wireless networks performance evaluation and security.
Email: olariu@cs.odu.edu

Shaharuddin Salleh is currently Associate Professor in Computational Mathematics at the Department of Mathematics, Universiti Teknologi Malaysia. He received PhD at the same university in 1998. His research interests in the areas of computational mathematics include algorithm designs for VLSI routing methods, mobile computing and parallel computing. To his credit, Shaharuddin has published more than 50 papers for various international journals and proceedings, and three books.