Collaborative and Distributed Computation in Mesh-Like Wireless Sensor Arrays*

Mitali Singh[1], Viktor K. Prasanna[1], Jose Rolim[2], and Cauligi S. Raghavendra[1]

[1] University of Southern California
Department of Computer Science
Los Angeles, CA 90089, USA
{mitali,prasanna,raghu}@usc.edu
http://pacman.usc.edu
[2] University of Geneva
Department of Informatics
Geneva 4, Switzerland
rolim@cui.unige.ch

Abstract. Sensor networks are being used for implementation of a large number of applications involving distributed and collaborative computation. Extensive research has focused upon design of time optimal parallel and distributed algorithms for two dimensional mesh connected computers (MCC). In this paper, we discuss a simple scheme for emulating the above algorithms for mesh-like sensor arrays. We show that a large set of parallel algorithms (see Property 1), that take time $T(n)$ on MCC of size n, can be implemented on a wireless sensor mesh of size n in time $O(r^2 + T(n/r^2).r^2)$. Here r represents the transmission range of the sensor nodes. We discuss implementation of algorithms for ranking and sorting using our techniques and analyze them for time and energy efficiency.

1 Introduction

Sensor networks can be considered as large scale dynamically configurable, distributed systems, where autonomous nodes (sensor nodes) collaborate among themselves to achieve a larger objective. Such networks have revolutionized data gathering and processing, and enabled a large range of applications such as unattended environment monitoring, traffic control, automatic target recognition, building vigilance, and hazard detection. In recent years, sensor networks have gained tremendous popularity in both the research community and the industry. Efforts are being made to design sensor nodes that are small, low cost and yet have large functionality built into them. Sensor applications often require deployment of nodes in inaccessible, remote areas. This makes energy management critical in sensor networks, where the functionality of the network is limited by

* This work is supported by the DARPA Power Aware Computing and Communication Program under contract no. F33615-C-00-1633.

the battery life of the nodes. Several research projects [9] [14] [16] [18] are focusing on design of energy efficient hardware and software for sensor applications.

We approach sensor networking from a parallel and distributed system's perspective and focus on developing algorithms for sensor networks using formal analysis. We observe that most of the general models of computation must be redefined in the context of sensor networks. Wireless communication and energy constraints are the two major factors that are responsible for the above. Moreover, while some sensor networks have regular topologies (such as traffic monitoring sensor nodes on cross-streets), most involve an ad hoc deployment of sensor nodes. Design of distributed algorithms for such networks is very challenging. However, we observe that a mesh like topology is intrinsic to densely populated sensor nodes uniformly distributed in a two dimensional plane (see Figure 2). The overheads for maintaining the topology are minimal if the network has been localized [4], synchronized [5] and each sensor node is aware of its location in an absolute or relative scale. We assume sensor nodes to be organized in mesh-like wireless sensor arrays and investigate design of time and energy efficient distributed algorithms for these systems.

A large number of distributed algorithms have been designed and analyzed in the past for time optimal implementation over two dimensional Mesh Connected computers (MCC) [7] [8] [10] [13]. We discuss a simple scheme for adapting these algorithms for implementation in wireless sensor networks. We observe that the efficiency of the algorithms is largely influenced by the transmission range of the sensor nodes. Larger range reduces the diameter of the network but also results in higher interference. This reduces the number of sensor nodes that can transmit concurrently. Moreover, larger range results in higher energy dissipation. We demonstrate that any algorithm with execution time $T(n)$ on MCC of size n can be emulated on a wireless mesh in time $O(n + T(n/r^2).r^2)$, where r is the transmission range and n is the total number of sensor nodes.

As an illustration, we discuss time optimal implementations of algorithms for sorting and ranking, and also analyze them for overall energy dissipation. We demonstrate that ranking can be performed in time $O(r^2 + \sqrt{n}/r.)$ with overall energy dissipation $O(n.r^2)$. Our algorithm for sorting executes in time $O(r^2 + r.\sqrt{n})$ and energy $O(n.r^2 + r.n\sqrt{n})$. We observe the results are asymptotically the same as the sorting algorithm for wireless networks discussed in [3].

The rest of the paper is organized as follows. Our model is presented in Section 2. The algorithms for ranking and sorting are discussed in Section 3. Finally, we conclude in Section 4.

2 Our Model

For our analysis, we consider a uniform distribution of sensor nodes over a two dimensional square plane of size $\sqrt{n} \times \sqrt{n}$ as illustrated in Figure 1. The sensor nodes communicate with each other over a single wireless channel with fixed transmission range r. We divide the plane into unit area cells such that each cell contains at least one sensor node. To ensure connectivity in the network

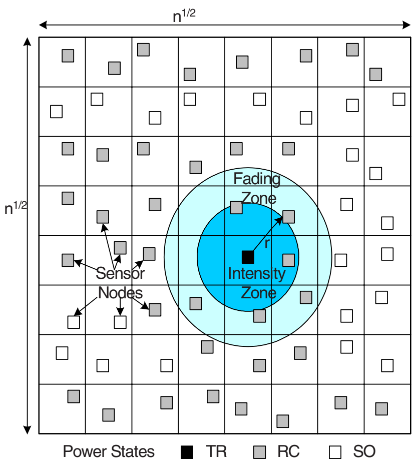


Fig. 1. Sensor network model

we assume $r \geq 1$. The sensor nodes, thus form a mesh like topology over the network. A similar topology is assumed by the Wireless Sensor Network (WSN) model described in [1]. Our model augments the WSN model with energy costs. Some of the key assumptions of our model are discussed below.

- **Network features:** We assume a homogeneous network comprising of n sensor nodes organized in a mesh like topology. Each sensor node has a unique id, and is aware of its relative position in the mesh. Energy efficient initialization schemes for wireless sensor networks are discussed in [11] [6]. The network is time synchronized. A sensor node has constant amount of memory denoted by $m > 1$, and **fixed** transmission radius $r \geq 1$.
- **Communication:** The sensor nodes communicate over a single wireless channel. The coverage area of a sensor with transmission range r, is defined as the distance till which it can be heard. All sensor nodes lying within distance r from the transmitting sensor are said to be in the *intensity zone* (see Figure 1) and are guaranteed to receive the transmission. The transmission signal strength reduces with distance as a function of r. We assume that it can be detected up to a distance $2r$ after which the signal power is below the reception level. The sensor nodes lying within distance $r < d \leq 2r$ lie in the *fading zone*. A COLLISION is said to occur if a sensor node lies within the coverage area of two or more transmitting sensor nodes. In any time step a sensor can receive or transmit one unit of data.
- **Power States:** A sensor node can be in three power states: transmit (TC), receive (RC), or switched off (SO) power state. A sensor can transmit in state TC and receive in state RC. No operation can be performed in state SO. Energy dissipation in state SO is considered to be negligible.
- **Communication Energy:** The communication energy of the network consists of the total transmission energy and the reception energy. Power dis-

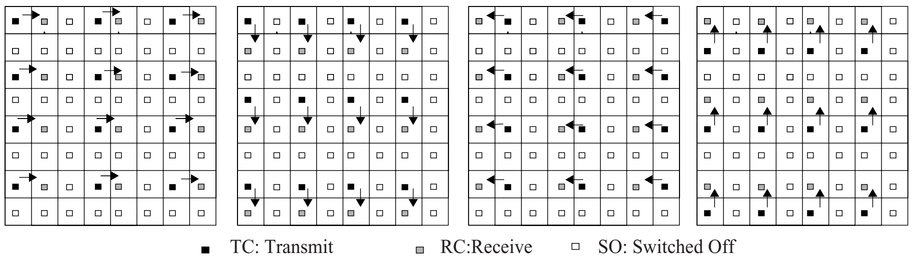


Fig. 2. Collision free transmission

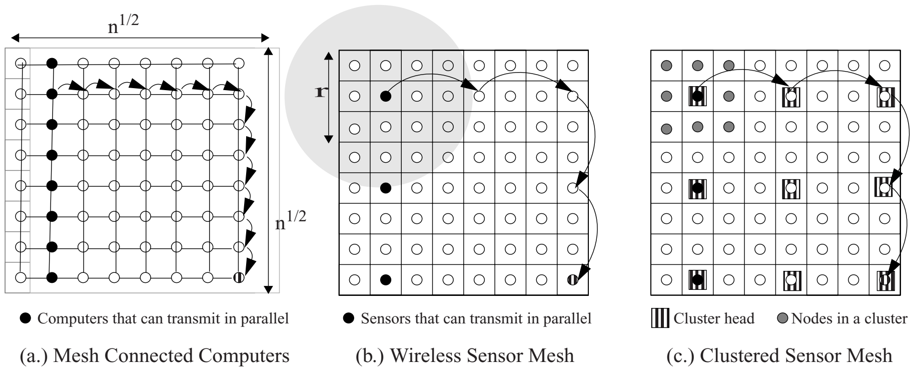
sipation at the transmitter has two components, P_{Tx} and P_{Rad}. The power dissipation at a receiver is given by P_{Rx}. Here, P_{Tx} and P_{Rx} represent the power dissipated in processing a (transmitted or received) packet. This value depends on the electronics of the radio and the type of radio components integrated (such as frequency synthesizers, mixers, modulators) in the module and is equivalent. P_{Rad} denotes the radiation power at the transmitter, which is a function of the transmission range r, and the environment-dependent path loss exponent α. Typically α lies between $1.7 - 4$ [17]. We assume $\alpha = 2$. We define one energy unit as the energy dissipated in receiving one unit of data. Transmission of one unit of data dissipates energy r^2

– **Overall Time and Energy:** Energy and time costs for communication are much larger than computation in state-of-the-art sensor systems [19]. Thus **we consider only communication time and energy** for analysis of our algorithms. We do not consider energy dissipation or time taken for computation or sensing of data at any sensor node.

3 Mesh Emulation

Extensive literature exists for time and work optimal distributed algorithms designed for mesh connected computers (MCC). We discuss a simple scheme for emulating the above algorithms on mesh-like sensor arrays. Our goal is to minimize overall execution time and energy. Sensor networks use wireless technology for communication. Thus, in order to minimize interference and collisions in the network, the transmission schedules must be defined such that no receiving sensor node is in the coverage area of more than one transmitting sensor node. The coverage area of a sensor node is determined by the transmission range r. For $r = 1$, the behavior of the network is very similar to MCC. Lemma 1 demonstrates that time taken for emulation of an MCC algorithm is of the same order. For $r > 1$, the network diameter is reduced, but interference is increased, and the performance bounds for emulating the algorithm are discussed in Theorem 1.

Lemma 1. *Any parallel algorithm for MCC with time complexity $T(n)$ can be emulated on sensor mesh array with $r = 1$ in time $O(T(n))$.*



n^1/2
n^1/2
r
● Computers that can transmit in parallel
● Sensors that can transmit in parallel
Cluster head
Nodes in a cluster
(a.) Mesh Connected Computers
(b.) Wireless Sensor Mesh
(c.) Clustered Sensor Mesh

Fig. 3. Communication in MCC and sensor mesh

Proof: In the MCC model, a processing element can receive or transmit to each of its adjacent neighbors at any given time if the neighbors are idle. In a wireless mesh, the broadcast medium imposes some restrictions. To ensure collision free communication, no receiver should be in transmission range of more than one sensor node at any given time. This limits the number of sensor nodes that can broadcast concurrently reducing the bandwidth of the network. However, it can be shown that any operation of the MCC can be simulated on a wireless mesh in 24 time steps. In each time step, the mesh is assigned a specific power configuration, which limits the functionality of a sensor node in that time step. A sensor can transmit only in time steps, where it is scheduled to be in state TR. Figure 2 illustrates four time steps emulating four direction of communication. In any of the 4 time steps only $n/6$ sensor nodes can transmit. Thus, to emulate an operation of the MCC, 24 configurations are sufficient. The other 20 configurations can be obtained by changing the power states of the illustrated sensor nodes in a cyclic manner. Thus, time complexity of the algorithm when implemented on a wireless mesh is $24.T(n)$. □

Next we analyze the scenario where each sensor node has a transmission range $r > 1$. Each sensor can reach $O(r^2)$ adjacent sensor nodes in a single hop. In MCC, a processing element can communicate with only 4 adjacent processing elements in a single transmission. The network diameter of a $\sqrt{n} \times \sqrt{n}$ mesh is given by $\sqrt{n}$ in the MCC model and is reduced to $\sqrt{n}/r$ in a wireless mesh. Figure 3(a.) shows the number of hops required to traverse a mesh in MCC and Figure 3(b.) illustrates the number of hops required in a wireless mesh of same size. However, the broadcast medium of the wireless mesh imposes restrictions on the number of sensor nodes that can transmit concurrently. The coverage area of a sensor node is $O(r^2)$. Only one sensor in this area can transmit at a time. This reduces the mesh bandwidth by $O(r^2)$, as shown in Figure 3. All black processing elements in the MCC can transmit to their right neighbor in the same time slot. However, in the wireless mesh, the transmitters must be interleaved to ensure that the intended listener can only hear one transmission at a time.

This motivates a clustering approach as illustrated in Figure 3(c.). The sensor nodes are divided into clusters of size r^2, with the central sensor chosen as the cluster head. A mesh of size $\sqrt{n}/r \times \sqrt{n}/r$ involving only the cluster heads is imposed on the network. Each sensor in a cluster only communicates with its cluster head. Only cluster heads take part in inter-cluster communication. The wireless mesh of size n where each sensor node has a single data element is transformed to a wireless mesh of size n/r^2 having r^2 data elements per sensor. Thus, an algorithm satisfying Property 1, with time complexity $T(n)$ on MCC of size n can be emulated on the cluster mesh in time $O(r^2 + T(n/r^2).r^2)$. We give a proof by construction below (Theorem 1).

Property 1. Problem Size Linearity: An algorithm implemented on MCC is said to have problem size linearity if $T^p(n) = O(p.T(n))$. $T(n)$ represents the time complexity of the algorithm implemented on MCC of size n, where each processing element has a single unit of data. $T^p(n)$ denotes the time complexity of the algorithm implemented on a mesh of size n, where each processing element stores $p \geq 1$ data elements.

Remark 1. We observe that a very large set of algorithms satisfy problem size linearity. These include algorithms for problems such as sorting, ranking, matrix multiplication, sum, prefix sum, permutation routing, and matrix transposition.

Theorem 1. *A parallel algorithm implemented on a $\sqrt{n} \times \sqrt{n}$ MCC, that has problem size linearity and execution time $T(n)$ can be implemented on the mesh of sensor nodes of size $\sqrt{n} \times \sqrt{n}$ with transmission range r, in time $O(r^2 + T(n/r^2) \times r^2)$. Here, r represents the transmission range of a sensor.*

Proof: The proof is by construction. Consider an algorithm implemented on an $\sqrt{n} \times \sqrt{n}$ mesh with time complexity $T(n)$. We emulate the algorithm on an $\sqrt{n} \times \sqrt{n}$ sensor mesh, where each sensor node has a range r.

Step I: Divide the mesh into n/r^2 blocks of size $r \times r$. Let B_{ij} represent the block in the i^{th} row and j^{th} column, where $1 \leq i, j \leq \sqrt{n}/r$. Node $N_{(i.r+\lceil r/2 \rceil),(j.r+\lceil r/2 \rceil)}$ is chosen to be the cluster head for sensor nodes in $B_{i,j}$ and is denoted by $S_{i,j}$. Initially all sensor nodes are in power state SO.

Step II: Data is aggregated from all sensor nodes in a block to the cluster head. Each block is scheduled to be either *active* or *inactive*. The power state of the sensor nodes in inactive blocks is SO. In each active block, the cluster head is in state RC for r^2 time slots. The $r^2 - 1$ sensor nodes transmit sequentially. To ensure collision free transmission, the active blocks must be interleaved by two inactive blocks as illustrated in Figure 4. This ensures that their is no interference from neighboring blocks at any time.

Step III: All sensor nodes except the cluster heads are put in state SO. Sensor nodes $S_{i,j}$ form a $\sqrt{n}/r \times \sqrt{n}/r$ mesh with r being the distance between adjacent cluster heads. Each cluster head represents r^2 data elements. The clusterhead mesh is similar to the sensor mesh with $r = 1$ as a sensor can only reach 4 adjacent blocks. A collision free emulation of an algorithm designed for MCC, can be ensured by the communication schedule defined earlier (in proof of Lemma 1) at the penalty of increase in time by a factor of 24.

The algorithm satisfies Property 1, thus on MCC of size n/r^2, where each processing element stores r^2, it takes time $O(T(n/r^2).r^2)$. Using Lemma 1, we can conclude that the time complexity of the simulation on the wireless mesh is given by $O(24.T(n/r^2).r^2)$.

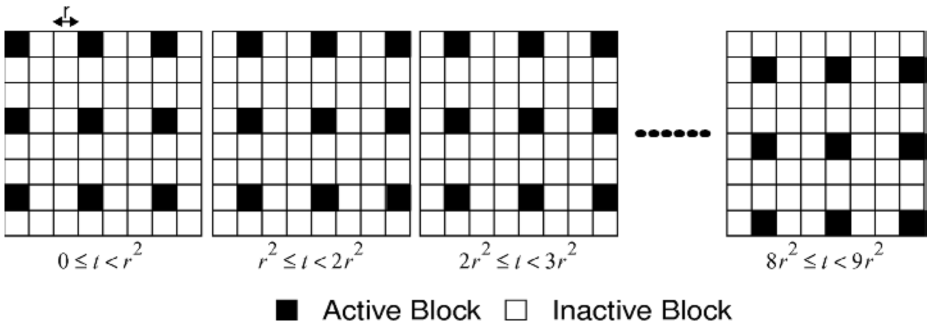
Step IV: The cluster heads transmit the result to the sensor nodes using reverse scheduling of Step I.

Analysis: In step II, the aggregation completes in time $9r^2$. Step III takes time $O(T(n/r^2).r^2)$. Analysis of Step IV is analogous to step II. Thus, time complexity of the algorithm implemented on a wireless mesh of size n is given by $O(r^2 + T(n/r^2).r^2)$. □

Remark 2. Theorem 1 gives an upper bound on the time complexity of MCC algorithm implemented on a wireless mesh. It assumes that the cluster head represents a data set of size of $O(r^2)$ at all time steps. This is the worst case scenario. For several algorithms, the size of the data can be reduced. Consider the following two scenarios (1.) Only a fraction of sensor nodes per cluster contain useful data.(2.) Data aggregation can take place at the cluster head. Let us assume that only $1 \leq l \leq r^2$ sensor nodes per cluster collaborate. The data can be transmitted to the cluster head in time $O(l)$. Next the cluster head aggregates the data. For example, only retains the partial sum in an algorithm for summing. Then the data set represented by a cluster head is reduced to size $1 \leq k \leq l$. Thus, time taken by the algorithm is $O(l + T(n/r^2).k)$ where $1 \leq k \leq l \leq r^2$.

Remark 3. In our proof of Theorem 1, we considered data from all sensor nodes in a block to be collected at the cluster head. This implies, that the the cluster head must have a memory of size $O(r^2)$. The data aggregation at the cluster head was assumed to keep the proof simple. The memory constraint can be relaxed as follows. Each communication step at the cluster head is split into two steps. In the first step the cluster head transmits in its cluster, the id of the sensor whose data is required. All sensor nodes receive the message. In the next step, only the sensor that matches the id remains awake and transmits/receives the data from the cluster head. Every single communication step at the cluster head now requires three more communication steps instead of one. This increases the overall execution time of the algorithm only by a constant factor.

The above theorem is useful as it helps in evaluating the existing MCC algorithms for time performance when implemented on a wireless mesh and thus, in selection of the optimal candidate. We discuss implementations of algorithms for ranking and sorting and in a wireless, mesh-like sensor arrays.



```
for t =0 to $9r^2-1$ {
  for all 0 < i, j < ($n^{1/2}$/ r+1); 0 < k, l, < $r^2$ +1
    if (((i-1) mod 3= t / $3r^2$) & ((j-1) mod 3= t/ $r^2$ mod 3) & (k= (t mod $r^2$/r + 1)& (l= t mod r +1))
      { State (Node$_{i.r+k,\ j.r+1}$) = TR;
      State (S$_{i,\ j}$) = RC ;  }
    else{
      State (Node$_{i.r+k,\ j.r+1}$) = SO;
      State (S$_{i,\ j}$) = SO ;  }
      }
```

Fig. 4. Power schedule for Step II

3.1 Ranking

The problem of ranking is described as follows. Given a set of n sensor node, determine the rank of a sensor node with id i, where $1 \le i \le n$. Ranking is a useful operation in several sensor applications. For example consider a scenario, where k highest power sensor nodes are required to be monitoring the field at any time. A sensor node may periodically poll the network to determine its rank to decide whether to monitor the field or go to sleep. Alternatively consider a scenario, where sensor nodes transmit in order of their rank. A sensor node must find its rank to determine its transmission slot. Ranking is also an important kernel for several filtering and image processing applications.

We consider the sensor field of size $\sqrt{n} \times \sqrt{n}$, where each sensor node has fixed transmission range r. Without loss of generality, let us assume sensor node $N_{1,1}$ requires to determine its rank.

Step I: Divide the mesh into blocks of size $r \times r$ and choose a cluster head $S_{i,j}$ for each block $B_{i,j}$. All sensor nodes except $N_{1,1}$ and $S_{1,1}$ are inactive. $N_{1,1}$ transmits its data to $S_{1,1}$.

Step II: In the $(\sqrt{n}/r)$ time steps, the value is transmitted to all the cluster heads in the row 1. In the next $\sqrt{n}/r$ time steps, the value is transmitted to all the cluster heads.

Step III: To avoid interference each block is scheduled to be active and inactive as illustrated in Figure 4 (discussed earlier in proof of Theorem 1). Each block is active for r^2 time steps. In the first time step the cluster head transmits the value to the cluster. All sensor nodes that have value larger than value transmitted transmit a response to the cluster head in their scheduled time slot. This can be accomplished in $r^2 - 1$ time steps without any collision. The cluster heads count the responses received. They compare the value to their own and increment the counter if required.

Step IV: The reverse schedule of step II is followed. Whenever, a cluster head receives a value, it adds its own counter to the value and transmits it to the next cluster head. This is accomplished in time $O(\sqrt{n}/r)$. $S_{1,1}$ transmits result to $N_{1,1}$.

Analysis: Each sensor node transmits and receives at most twice. Thus, energy of this algorithm is $O(n.r^2)$ and time taken is $O(r^2 + \sqrt{n}/r)$. The result is both time and energy optimal if the transmission range on the sensor nodes is fixed to r. The proof is trivial. Each sensor node must transmit its value at least once and each transmission costs energy r^2. Thus $E(n) = \Omega(n.r^2)$. The network diameter $\sqrt{n}$ must be traversed for data from $N_{1,1}$ to reach $N_{\sqrt{n},\sqrt{n}}$. Since the range of a sensor node is r, this requires at least $\sqrt{n}/r$ sequential operations. Thus, time complexity is $\Omega(\sqrt{n}/r)$ for $r < \sqrt{n}$. For $r \geq \sqrt{n}$, all sensor nodes can hear all transmissions. This implies only one sensor node may transmit at any time resulting in an overall time complexity of $\Omega(r^2)$.

3.2 Sorting

The problem of sorting n numbers is one of the most widely analyzed problem owing to its theoretical importance and use in a wide range of applications. Pixel sorting is required in several image compression and coding algorithms [15]. A sensor network can also be considered as a distributed data management system [2]. Sorting is an important kernel in a large number of data management and data mining applications. Sorting is useful when all sensors must determine their relative rank based on measured data or remaining battery power for filtering or network management applications. algorithms. The sorting problem is defined as follows. We are given a set of sensor nodes of n, where each sensor node contains a data element. Our goal is to redistribute the data among the sensors such that at the end of the sorting algorithm, for $1 \leq i \leq n$, sensor node with id i contains the data element with rank i.

Nassimi et al. [13] developed a parallel algorithm for bitonic sort that sorts the numbers on an $\sqrt{n} \times \sqrt{n}$ MCC in time $O(\sqrt{n})$. An implementation of the above algorithm on a wireless sensor mesh was discussed in [3]. The time complexity of the algorithm was demonstrated to be $O(r^2 + r\sqrt{n})$. Analysis using our model shows that the energy complexity of this algorithm is $O(n.r^2 + r.n\sqrt{n})$.

We observe that the algorithm bitonic sort satisfies the property of Problem size linearity (Property 1). Thus, a simple implementation of the algorithm bitonic sort can be achieved by using the clustering approach proposed in this paper. From Theorem 1, we conclude that using the clustering approach (discussed in proof of Theorem 1), bitonic sort can be implemented on a wireless mesh of size $\sqrt{n} \times \sqrt{n}$ in time $O(r^2 + \sqrt{n}/r.r^2) = O(r^2 + \sqrt{n}.r)$. Further analysis shows that the energy complexity of the algorithm is $O(n.r^2 + r.n\sqrt{n})$. Note the time performance and energy dissipation of our algorithm is same as the implementation discussed in [3].

The algorithm is time and energy optimal. Consider the scenario where $r < \sqrt{n}$. Let us assume that for all $1 \leq i, j, \sqrt{n}$, data element at position (i, j) in the mesh will be moved to position $((i + \sqrt{n}) mod \sqrt{n}, j)$ in the sorted distribution. Thus, each element must travel a distance of $\sqrt{n}$ to reach its final position (position after sorting is complete). This involves $\sqrt{n}/r$ hops. At any time only one sensor node out of adjacent r^2 (in a block of r^2) sensor nodes can transmit due to interference. Thus total time taken for data elements to reach their sorted position is $\Omega(r^2 + \sqrt{n}.r)$. Moreover, each data element makes $\sqrt{n}/r$ hops. This implies that the energy complexity of the algorithm is $\Omega((n.\sqrt{n}/r).r^2)$. Next we examine the scenario where $r \geq \sqrt{n}$. All sensor nodes can listen to each other. All sensors must transmit the value at least one. Thus, execution time is $\Omega(n)$ and energy dissipation is $\Omega(n.r^2)$.

4 Conclusions

In this paper, we illustrated a methodology for emulating parallel algorithms designed for mesh-connected computers onto wireless sensor meshes, and analyzed them as a function of the transmission range r of the sensor nodes. For a fixed transmission range r, the algorithms for ranking, and sorting are time and energy optimal.

The results obtained in this work are simple, and yet they demonstrate how analysis from prior research can be leveraged to design energy (or time) optimal algorithms for distributed sensor networks by exploiting the network configurability and density. The network model proposed in this work is an initial step towards understanding performance in these networks. The problem becomes more challenging when additional network parameters are considered to be variant such as number of communication channels, variable range control for transmission, fine tuned direction control at transmitter and receiver, among others. Moreover, a sensor network can be hierarchical in nature with some nodes being more powerful (more memory, faster computation, etc.) than others. Analysis of such networks require more sophisticated models.

In this paper, we assume that the sensor nodes organize themselves in a perfect mesh. It would be of interest to simulate a real scenario, and investigate the performance impact, when the sensor nodes are randomly distributed. The analysis will also provide some insight into the network design problem, as to with what density should the sensor nodes be deployed.

References

1. R. S. Bhuvaneswaran, L. J. Bordim, J. Cui, and K. Nakano, “Fundamental Protocols for Wireless Sensor Networks,” International Parallel and Distributed Processing Symposium (IPDPS), Workshop on Advances in Parallel and Distributed Computatinal Models, April 2001
2. P. Bonnet, J. E. Gehrke, and P. Seshadri, “Towards Sensor Database Systems,” International Conference on Mobile Data Management (MDM), January 2001
3. J. K. Bordim, K. Nakano, and H. Shen, “Sorting on Single-Channel Wireless Sensor Networks,” International Symposium on Parallel Architectures, Algorithms, and Networks (ISPAN), May 2002
4. N. Bulusu, J. Heidemann, and D. Estrin, “GPS-less Low Cost Outdoor Localization For Very Small Devices,” IEEE Personal Communications, October 2000
5. J. Elson and D. Estrin, “Time Synchronization in Wireless Sensor Networks,” International Parallel and Distributed Processing Symposium (IPDPS), Workshop on Parallel and Distributed Computing Issues for Wireless and Mobile Systems, April 2001
6. T. Hayashi, K. Nakano, and S. Olariu, “Randomized Initialization Protocols for Packet Radio Networks,” Discrete Mathematics and Theoretical Computer Science, SIAM Press (2000) 221–235
7. Joseph Ja Ja, “An Introduction to Parallel Algorithms,” Addison Wesley Publishing Company (1992)
8. V. Kumar, A. Grama, A. Gupta, and G. Karypis, “Introduction to Parallel Computing: Design and Analysis of Algorithms,” The Benjamin/Cummings Publishing Company (1994)
9. The μAMPS Project, http://www-mtl.mit.edu/research/icsystems/uamps/
10. R. Miller and Q. F. Stout, “Parallel Algorithms for Regular Architectures: Meshes and Pyramids,” The MIT Press (1996)
11. K. Nakano and S. Olariu, “Energy-Efficient Initialization Protocols for Radio Networks with no Collision Detection,” IEEE Transactions on Parallel and Distributed Systems, Vol. 11 (2000) 851–863
12. K. Nakano, S. Olariu, and J. L. Schwing, “Broadcast-Efficient Protocols for Mobile Radio Networks,” IEEE Transactions on Parallel and Distributed Systems, Vol. 10 (1999) 1276–1289
13. D. Nassimi and S. Sahni, “Bitonic Sort on Mesh-Connected Computer,” IEEE Transactions on Computers, Vol. c-27 January (1979)
14. The PACMAN Project, http://pacman.usc.edu
15. K. Peng and J. Kieffer, “Embedded Image Compression Based on Wavelet Pixel Classification and Sorting,” International Conference on Acoustics, Speech, and Signal Processing (ICASSP), May 2002
16. The PicoRadio Project, http://bwrc.eecs.berkeley.edu/Research/Pico_Radio/Default.htm
17. T. S. Rappaport, “Wireless Communication,” Prentice-Hall (1996)
18. The Smart Dust Project, http://robotics.eecs.berkeley.edu/~pister/SmartDust/
19. M. Singh and V. K. Prasanna, “System-Level Energy Tradeoffs for Collaborative Computation in Wireless Networks,” International Conference on Communications (ICC), Workshop on Integrated Management of Power Aware Communications, Computing and NeTworking, May 2002